

Figure 1. Breaking the upper bound on achievable quality with less aggressive downsampling. Since diffusion models often exploit inductive biases for spatial data, we do not need the heavy spatial downsampling of related generative models in latent space but still downsampled the dimensionality of the data into a suitable size for training. We use the DDIM [1] and DDPM [2] as our baseline models. See Sec. 4 for more details. The spatial downsampling factor is 7. Reconstruction EDs [1, 2] and PSNR are calculated on language-level [1, 2] see also Tab. 3.

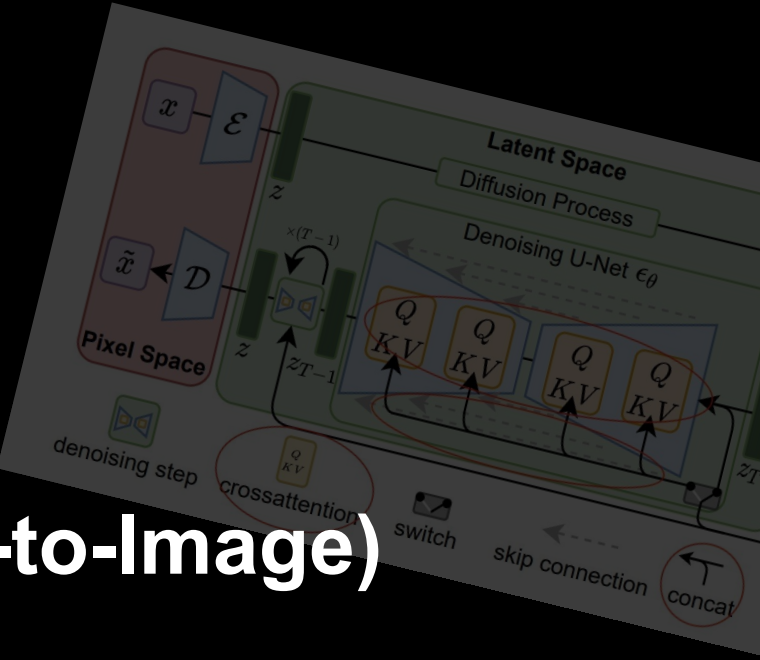
Figure 2. Results in image synthesis [2, 3] and beyond [4, 5, 6, 7]. We define the state-of-the-art in class-conditional image synthesis [1, 2] and super-resolution [1]. Moreover, even unconditional DMs can readily be applied to tasks such as inpainting and colorization [1] or stroke-based models [8, 9]. Being likelihood-based models they do not exhibit mode-collapse and training instabilities as GANs [10, 11]. By exploring parameter sharing, they can exploit mode-collapses of natural images without the need for complex distributions of natural images [12] and, hence, exploring parameter sharing, they can exhibit mode-collapse and training instabilities as GANs [10, 11].

Figure 3. Denoising Diffusion Probabilistic Models (DDPM) [12]. DDPMs are a class of generative models that can be trained on a wide range of data distributions. They are trained on a wide range of data distributions. They are trained on a wide range of data distributions.

Figure 4. Denoising Diffusion Probabilistic Models (DDPM) [12]. DDPMs are a class of generative models that can be trained on a wide range of data distributions. They are trained on a wide range of data distributions.

Speakage (Speech-to-Image)

TEAM #5



Affan Bin Usman
ausman4@asu.edu

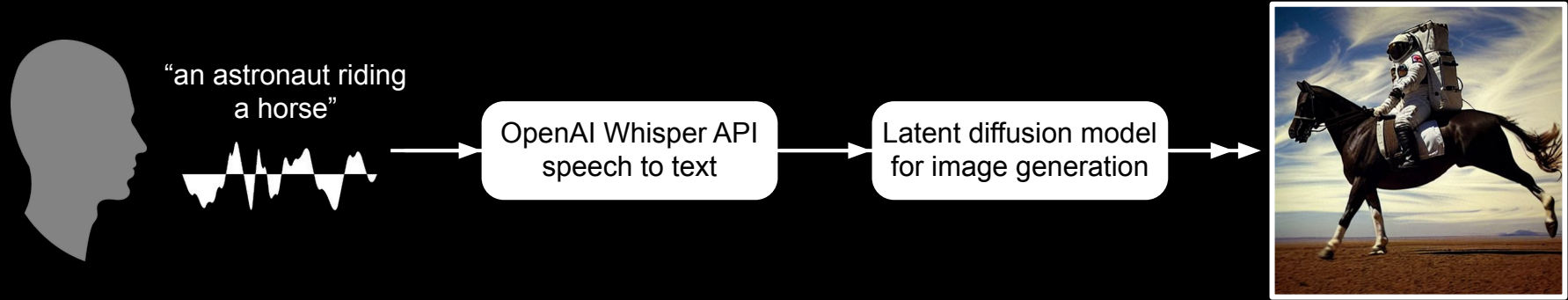
Avnish Singh
asing330@asu.edu

Sidhant Das
sdas116@asu.edu

Vihari Gandrakota
vgandrak@asu.edu



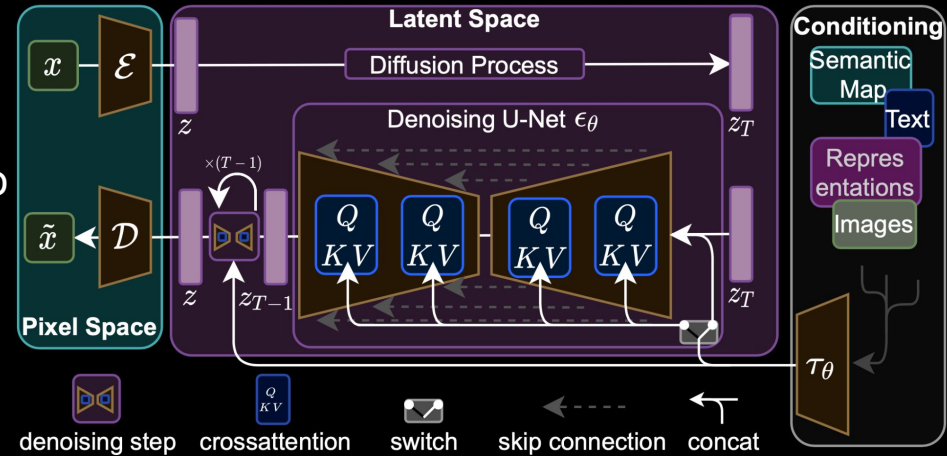
Problem Statement



- Traditional GAN techniques may run into “mode-collapse” during image generation
- We use diffusion based models to “generate an image from speech”
- Used transfer learning to train diffusion model on our custom dataset

Algorithms & Approaches

- Model - Pre Trained Latent Diffusion Model from Stable Diffusion
- Training data - Custom data set containing 30 images of Affan resized to 256x256x3 preserving aspect ratio
- Training hyperparameters:
 - Learning rate: 1.0e-06
 - Training Steps: 2000
 - Batch Size: 1
 - Optimizer: Adam



$$\mathbb{E}_{\mathbf{x}, \mathbf{c}, \epsilon, t} \left[w_t \left\| \hat{\mathbf{x}}_\theta(\alpha_t \mathbf{x} + \sigma_t \epsilon, \mathbf{c}) - \mathbf{x} \right\|_2^2 \right]$$

Loss function (Perception Loss)



Algorithms & Approaches

Approach:

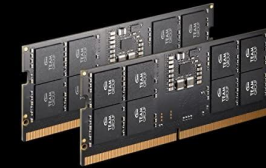
- This method takes a few images of a subject (Affan) and the corresponding class name (Person), and returns a fine-tuned/"personalized" text-to-image model that encodes a unique identifier that refers to the subject.
- Fine tuning the low-resolution text-to-image model with the input images by applying a class-specific prior to preservation loss
- Inference with a text prompt containing a unique identifier and the name of the class the subject belongs (Eg: Affan Person)



Implementation

- **Step 1**: System Requirements
 - RTX 3090 GPU
 - 24 GB RAM

```
requirements
omegaconf
einops
pytorch-lightning==1.6.5
test-tube
transformers
kornia
-e git+https://github.com/CompVis/taming-transformers.git
transformers
-e git+https://github.com/openai/CLIP.git@main#egg=clip
setuputils==59.5.0
pillow==9.0.1
torchmetrics==0.6.0
-e .
protobuf==3.20.1
gdown
-qq diffusers["training"]==0.3.0 transformers ttfy
-qq "ipywidgets>=7,<8"
huggingface_hub
ipywidgets==7.7.1
captionizer==1.0.1
```



- **Step 2**: Install dependencies
 - [Requirements.txt](#)

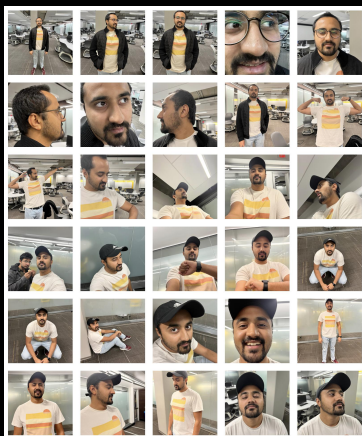
- **Step 3**: Obtain an open source weight file
 - [Pre trained weights](#)



Hugging Face

Implementation

- **Step 4:** Data preparation
 - [Custom dataset](#)
 - Token = Custom name (Affan)
 - Class = Person
- **Step 5:** Download token images
 - Custom (Affan's) pictures



```
Running DDIM Sampling with 50 timesteps
DDIM Sampler: 0% | 0/50 [00:00<, 7.1it/s]
DDIM Sampler: 2% | 1/50 [00:00-00:06, 7.87it/s]
DDIM Sampler: 4% | 2/50 [00:00-00:05, 8.05it/s]
DDIM Sampler: 6% | 3/50 [00:00-00:05, 8.12it/s]
DDIM Sampler: 8% | 4/50 [00:00-00:05, 8.16it/s]
DDIM Sampler: 10% | 5/50 [00:00-00:05, 8.18it/s]
DDIM Sampler: 12% | 6/50 [00:00-00:05, 8.19it/s]
DDIM Sampler: 14% | 7/50 [00:00-00:05, 8.19it/s]
DDIM Sampler: 16% | 8/50 [00:00-00:05, 8.20it/s]
DDIM Sampler: 18% | 9/50 [00:01-00:05, 8.19it/s]
DDIM Sampler: 20% | 10/50 [00:01-00:04, 8.19it/s]
DDIM Sampler: 22% | 11/50 [00:01-00:04, 8.20it/s]
DDIM Sampler: 24% | 12/50 [00:01-00:04, 8.19it/s]
DDIM Sampler: 26% | 13/50 [00:01-00:04, 8.20it/s]
DDIM Sampler: 28% | 14/50 [00:01-00:04, 8.20it/s]
DDIM Sampler: 30% | 15/50 [00:01-00:04, 8.21it/s]
DDIM Sampler: 32% | 16/50 [00:01-00:04, 8.21it/s]
DDIM Sampler: 34% | 17/50 [00:02-00:04, 8.20it/s]
DDIM Sampler: 36% | 18/50 [00:02-00:03, 8.20it/s]
DDIM Sampler: 38% | 19/50 [00:02-00:03, 8.19it/s]
DDIM Sampler: 40% | 20/50 [00:02-00:03, 8.19it/s]
DDIM Sampler: 42% | 21/50 [00:02-00:03, 8.19it/s]
DDIM Sampler: 44% | 22/50 [00:02-00:03, 8.19it/s]
DDIM Sampler: 46% | 23/50 [00:02-00:03, 8.19it/s]
DDIM Sampler: 48% | 24/50 [00:02-00:03, 8.20it/s]
DDIM Sampler: 50% | 25/50 [00:03-00:03, 8.20it/s]
DDIM Sampler: 52% | 26/50 [00:03-00:02, 8.20it/s]
DDIM Sampler: 54% | 27/50 [00:03-00:02, 8.19it/s]
DDIM Sampler: 56% | 28/50 [00:03-00:02, 8.20it/s]
DDIM Sampler: 58% | 29/50 [00:03-00:02, 8.19it/s]
DDIM Sampler: 60% | 30/50 [00:03-00:02, 8.19it/s]
DDIM Sampler: 62% | 31/50 [00:03-00:02, 8.18it/s]
DDIM Sampler: 64% | 32/50 [00:03-00:02, 8.18it/s]
DDIM Sampler: 66% | 33/50 [00:04-00:02, 8.19it/s]
DDIM Sampler: 68% | 34/50 [00:04-00:01, 8.20it/s]
DDIM Sampler: 70% | 35/50 [00:04-00:01, 8.20it/s]
DDIM Sampler: 72% | 36/50 [00:04-00:01, 8.20it/s]
DDIM Sampler: 74% | 37/50 [00:04-00:01, 8.21it/s]
DDIM Sampler: 76% | 38/50 [00:04-00:01, 8.20it/s]
DDIM Sampler: 78% | 39/50 [00:04-00:01, 8.20it/s]
DDIM Sampler: 80% | 40/50 [00:04-00:01, 8.19it/s]
DDIM Sampler: 82% | 41/50 [00:05-00:01, 8.18it/s]
DDIM Sampler: 84% | 42/50 [00:05-00:00, 8.18it/s]
DDIM Sampler: 86% | 43/50 [00:05-00:00, 8.18it/s]
```

- **Step 6:** Download class dataset
 - Person_ddim
- **Step 7:** [Run training](#)
 - Obtain [new weights](#) file
- **Step 8:** Prompt for new image generation

Results



"Affan person as a masterpiece portrait painting by John Singer Sargent in the style of Rembrandt"



"Affan person eating a pizza painting by John Singer Sargent in the style of Rembrandt"



"Affan person eating chocolate icecream hyperrealistic"

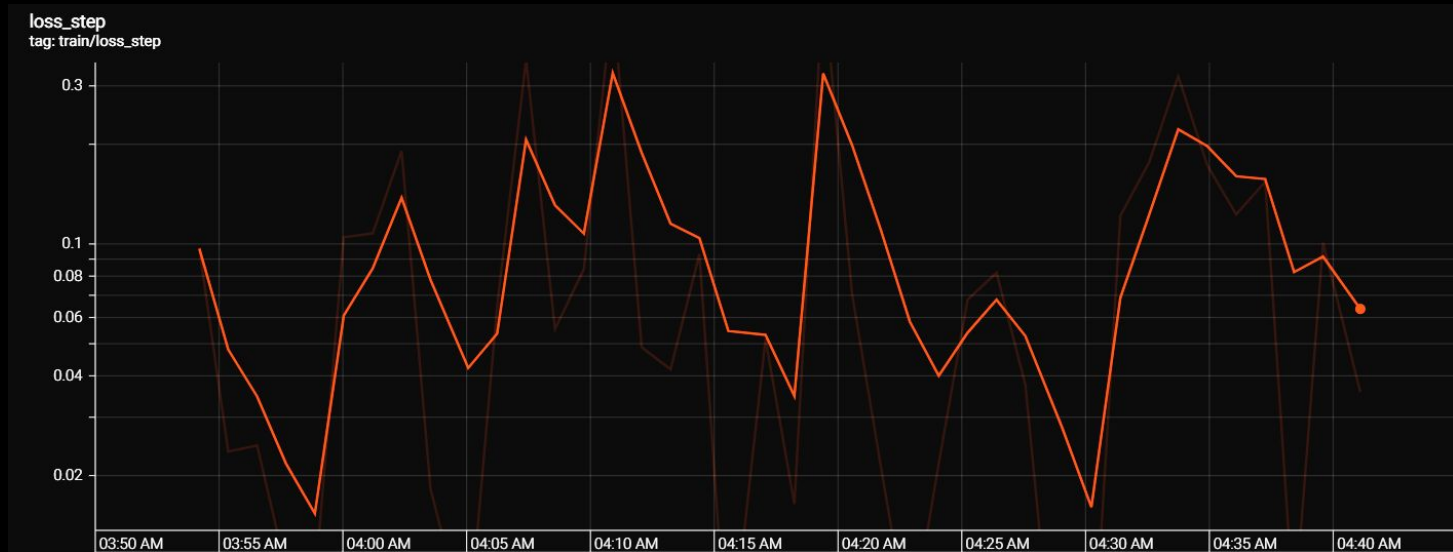


"Affan riding a motor bike on a highway hyper realistic"



"Affan skydiving from a plane"

Results



Training loss after 2000 steps is 0.03

Training the model for higher number of steps will saturate the loss at a value closer to zero.

The process took 46 minutes for 2000 steps



Applications

- Photography and painting Inspiration
- Computer Aided Design
- Fortnite emotes



Demonstration

- The following video demonstrates the whole process of generating an image from a speech prompt
 - Uses new weights
 - Generates image incorporating custom dataset

Video - [click here](#)



Questions?



THANK YOU!

