CS 771 HW4

Arshad Kazi, Bhuyashi Deka, Sadman Sakib

December 2024

1 UNet Architecture

Q1) Can you describe the architecture of the encoder and decoder within the UNet?

UNet follows an encoder decoder architecture with skip connections to perform semantic segmentation (pixelwise classification of an image).

Encoder: The encoder is responsible for extracting deep features from the input image. It compresses the input image into a low dimensional space, learning the spatial representations of the image. The original UNet uses a series of convolutional layers followed by Max pooling layers to downsample the feature maps.

Decoder: The decoder's role is to reconstruct the image's spatial resolution and generate a segmentation mask from the learned encoded features. It utilizes Transpose Convolution (Deconvolution) layers, which are employed to upsample the encoded feature maps back to the original resolution of the input image. The transpose convolution operation performs learnable upsampling, effectively increasing the spatial dimensions of the feature maps. To improve the performance of the decoder and facilitate the reconstruction of fine-grained details, skip connections are used. These connections concatenate the feature maps from the encoder (before downsampling) with the upsampled feature maps in the decoder. This helps preserve high-resolution spatial information from earlier layers in the network, which is crucial for accurate pixel-level segmentation. In the final layer of the decoder, a 1x1 Convolution is typically used to reduce the number of channels to the desired output (the number of segmentation classes), producing the final pixel-wise classification map.

Q2) How does the current implementation inject time t and condition c (image labels) into the UNet?

Adding Time(t):
The timestep in the first encoded using Sinusoidal Embedding and then directly

added into the feature map as a bias just at each block of the encoder of the UNet. This makes sure that every layer of the UNet is aware of the time step during the denoising process.

Adding Condition(c):
The condition is embedded into an neural network using Attention and Cross Attention mechanism into each layer of the UNet. The feature map passes through self attention layer and then through an cross attention with the condition embeddings. This conditioning is used to guide the denoising process towards the desired output distribution.

Implementation:
The implementation includes concatenation of feature maps from encoder and the previous block of the decoder. These concatenated features are passed through a resnet block along with timestep embeddings. The features are then passed through a spatial transformer that performs the cross-attention of condition and feature maps. These feature maps are then upsampled using Transpose Convolution and passed to the next decoder block. Finally, a final convolution layer is added to convert the output into a single-channel output.

2 MNIST Training

Forward Diffusion:
The qsample function is used to do the forward diffusion process by generating the noisy image for each timestep. This is done by adding noise to the original image in the increasing trend. For each time step the noise is sampled from a gaussian distribution using the following equation

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\mathbf{z} \tag{1}$$

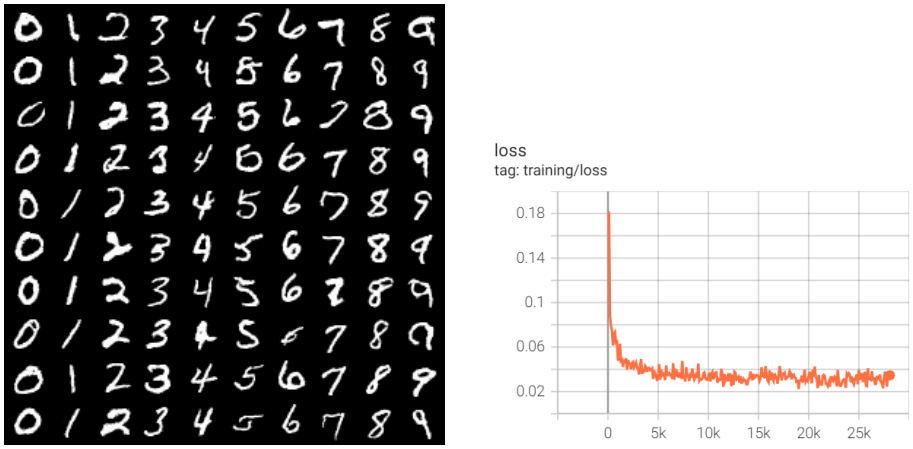
Loss Computation:
The denoising function learns to denoise the image at a given timestep t with a condition c. The function is trained to predict the noise that was added in the forward diffusion process. MSE loss is calculated between the predicted noise and the actual noise that was added to original image.

Sampling Process:
The sampling process starts with a noise image. The model then iteratively predicts the noise at each timestep that is then used to calculate the mean for the next iteration. The image at t-1 is generated by following equation

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t, t) \right) \tag{2}$$

$$x_{t-1} = \mu + \sigma \cdot \text{noise} \tag{3}$$

For the last step, no further noise is added as the data has almost reached the desired distribution.



(a) Sampled Images for MNIST

(b) Training Loss

Figure 1: MNIST outputs

Discussion of results: The trained model is able to generate handwritten digits with some artifacts specifically in '5'. The output was fairly in the acceptable range as the dataset is in pure black and white in color.

3 AFHQ Training

LDM performs the diffusion process in the latent space rather than the pixel space. The images are first encoded into the pixel space using a pre-trained autoencoder and the forward and the reverse diffusion process is carried out. During sampling the DDPM generates the latent space of the generated image which is then decoded to generate the final output image.

Discussion of results:

The results from LDM is partially able to construct the face of animals but could not construct the fine details of the image.

Figure 2: Sampled images from diffusion model

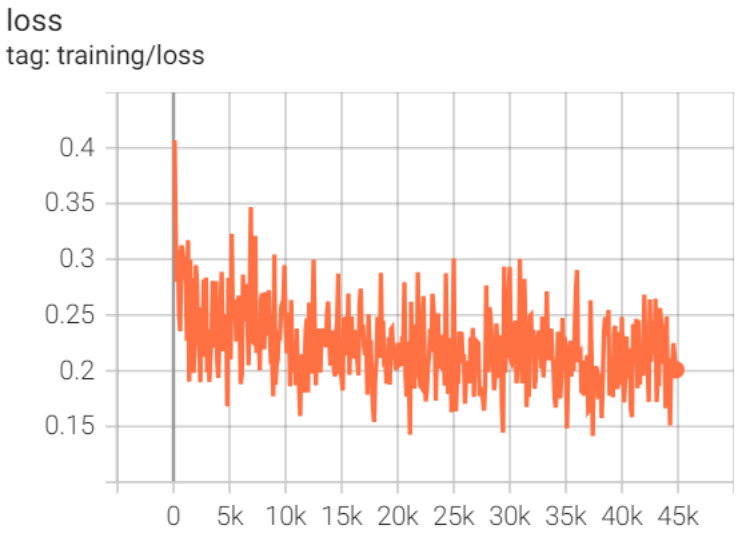


Figure 3: Training loss on AFHQ