Neural Networks for Images - Exercise #4

Submission Date - 29/6/2023 (23:59)

Programming Task: Style manipulation and classifier-guided denoising diffusion

In this exercise you will manipulate the style of images generated by a pre-trained denoising diffusion model using a mechanism called *classifier guidance*.

To accomplish the task you will need to use two main components:

- A module that computes the style loss of an image (given a style reference image)
- A denoising diffusion-based image generator

Part 1:
To compute the style loss you may use the following pytorch implementation of the Neural Style Transfer algorithm (by Gatys et al., which was covered in class on the 6/6). To make sure you understand the algorithm and its implementation, you are expected to perform and include in your report the following experiments:

A. **Feature inversion**: modify the algorithm to invert features from different stages of the VGG-19 network (which is the pretrained feature extraction network used by the algorithm). More specifically, given a target image, record its feature map Φ at the output of one of the layers of VGG-19. Next initialize an input image to noise, and optimize the input until its feature map for that layer closely approximates Φ. Experiment with inversion of layers of different types and at different depths of VGG-19.
B. **Texture synthesis**: modify the algorithm to generate a texture inspired by the style input, and demonstrate how using Gram matrices from different network layers affects the characteristics of the generated texture.
C. Implement an **"MeanVar" style loss** that computes and compares the mean and the variance of each activation channel, in a given layer, instead of the Gram matrix of the layer. Compare the style transfer results obtained with this style loss to those obtained by the original implementation that uses the Gram matrices.

Having mastered the style losses in the first part of the assignment, we will next use them to induce a target style onto an image generated with denoising diffusion, using a denoising diffusion mechanism called *classifier-guidance* (not to be confused with classifier-free guidance!)
The idea of classifier-guided diffusion is that the result of each denoising step is shifted slightly in the direction of the gradient of the log-probability predicted by some pre-trained classifier. This steers the result of the diffusion towards the corresponding class. Here's the pseudocode of classifier guidance as it appeared in the paper "Diffusion Models Beat GANs on Image Synthesis":

Algorithm 1 Classifier guided diffusion sampling, given a diffusion model $(\mu_\theta(x_t), \Sigma_\theta(x_t))$, classifier $p_\phi(y|x_t)$, and gradient scale s.

Input: class label y, gradient scale s
$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$
for all t from T to 1 **do**
  $\mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$
  $x_{t-1} \leftarrow$ sample from $\mathcal{N}(\mu + s\Sigma\, \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$
end for
return x_0

In the second part of this assignment we will attempt to steer the result towards some target style by using the gradient of the style loss instead of the gradient of the classifier's logit.
Note: classifiers in general, and the style loss as well, assume that they are given clean images, rather than the noisy intermediate images of the denoising diffusion process. Thus, in order to compute the style loss, rather than feeding it with the noisy sample x_t, it would probably be better to use the estimate of the clean image obtained from the noisy sample instead. This estimate is also available (or can be easily computed) at each denoising diffusion step.

Part 2:
Start by understanding how to use the diffusers library from huggingface to deploy a pre-trained diffusion pipeline, and how to deconstruct such a pipeline into a somewhat more detailed form, such that it would become possible to add the guidance described above into the denoising diffusion image generation process.

The diffusers library makes it easy to experiment with different pretrained diffusion models, and much more. For simplicity, it would be easiest to start with a simple unconditional DDPM pipeline, such as the `google/ddpm-celebahq-256` pipeline (trained to generate CelebA-HQ images at 256x256 resolution), for example:

Unfortunately, it is unclear how well the above model will serve our purpose: the ability of this model to generate faces with an interesting artistic style is likely to be limited, since the model was only trained on photorealistic portraits.

To produce (hopefully) more impressive results you could then switch to a pre-trained *stable diffusion* model (e.g., one of the models here) that was trained on a much wider variety of images, including a wide variety of artworks of many styles. Note that stable diffusion is a

text-to-image model, so you can have it generate images conditioned by a text prompt. In fact, you can also easily specify the style in the prompt. For example, using the prompt “an oil painting of a cat in the style of Van Gogh” you might get images like the one below:

For the purposes of this assignment, however, you may provide prompts to describe only the content of the generated images, while their style should attempt to match that of the reference style image (style input), so you should refrain from specifying anything stylistic in the prompt (except for comparison purposes in task B below).

The stable diffusion framework is efficient, since it performs the denoising on a latent space representation of the image (64x64x4). But this also means that the intermediate noisy “images” or the clean samples inferred from them, are not actual images, so before feeding them into the style loss computation you need to first convert them to an actual image using the VAE decoder that is part of the stable diffusion framework.

The actual experiments that you are expected to perform in this part are:

A. Compare the results of diffusion obtained with style loss guidance to those obtained by the optimization-based Neural Style Transfer from Part 1, using both the Gram-based and the MeanVar-based style loss.
B. Compare your style loss guided results to those obtained by ordinary text-to-image stable diffusion, with the text-based style description added to the prompt.

As in previous exercises, in each of your experiments you will need to implement the changes to the network(s), rationalize the results and document your conclusions as to what happened and why. Your report should be submitted as a pdf file, as explained in the Submission Guidelines below.

We expect you to report and elaborate on every practical task in the report, using your own words and your own analysis of what you’ve done. Include everything that you think is crucial for us to understand your way of thinking.

For this exercise the submission will be done in **<u>pairs</u>.** Please submit a single zip file named “ex4_ID1_ID2.zip”. This file should contain your code, along with an “ex4.pdf” file which should contain the description of your experiments with figures and analysis. Furthermore, include in this compressed file a README with your names, IDs, and CSE usernames.
Please write readable code, with documentation where needed, as the code will also be checked manually.