PANDA: Patched Averaging Noise Diffusion Algorithm

Chih-Chun,Chen 311505014 Yu-Che,Chang 311513058

January 12, 2023

Abstract

The purpose of our project is to improve the Diffusion Model for image generation by splitting the image into patches and training two models independently on the patches. The Anime Face Dataset (64x64) was used for training and the models were trained using the Adam optimizer with a batch size of 32 and initial learning rate of 0.0001 for 40 epochs. The experiment results showed that the generated anime faces were visually appealing and the method had the potential to be a valuable tool in the field of image generation. Additionally, the method resulted in lower VRAM usage and not too longer training time compared to the original Diffusion Model. Overall, the patch-based approach demonstrated promising results and has potential for further improvement.

1 Motivation

After we finished homework3, we found that the Diffusion Model is very powerful, and the generated images is with high quality, but at the same time, we are also thinking about whether there is still place for improvement of the diffusion model. Although the diffusion model is powerful, in the sample and training needs to consume a lot of VRAM and time, so in this project we tried to split the picture into several patches for training, to see if such shortcomings can be improved.

2 Method

2.1 Training

First we spilt our image into 4 patches, then we train two models independently($Model_L$ and $Model_R$) on same dataset, but we train $Model_L$ use left three patches, train $Model_R$ use right three patches. As Figure 1 shown.

2.2 Sampling

Our sample process is show in the Figure 2. First we spilt noised image x_t in time step t into 4 patches, and send left three patches (red, blue and green) to $Model_L$ and send right three patches (blue, green and yellow) to $Model_R$, then we combine 2 predict noise by taking average for overlapping patches. x_{t-1} is obtained by subtracting concatenation of these parts.

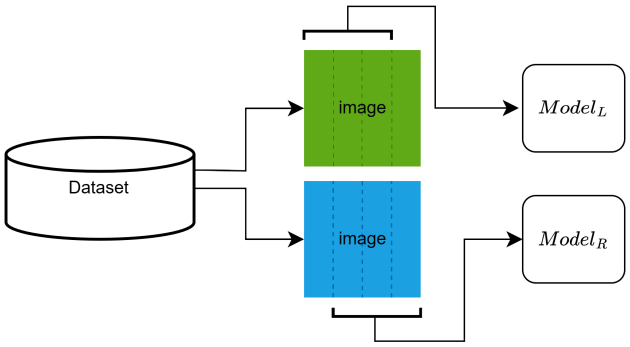

Figure 1: Training process.

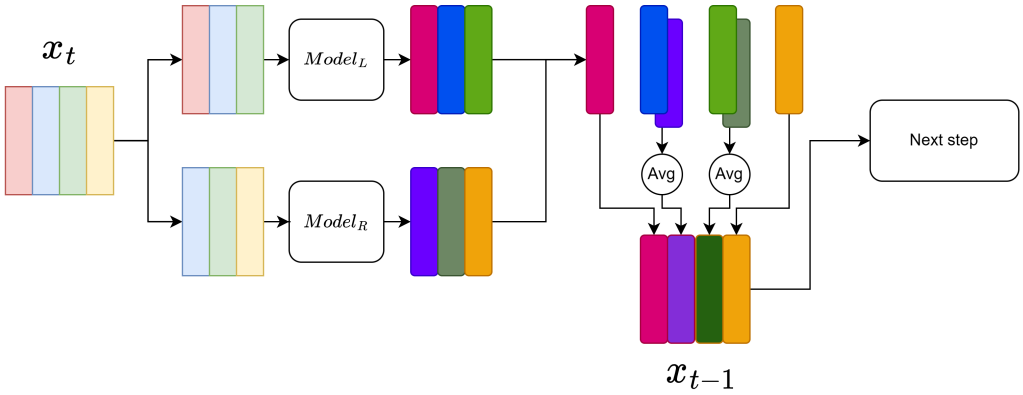

Figure 2: Sampling process.

3 Experiment

3.1 Original Design and Tried Things

Our initial experimental design was actually to spilt a picture into many patches, then add position encoding and send it to a model for training, but later we found that adding position encoding is not easy for model to learn, no matter how sample, it will only generate a certain corner. Afterwards, we splilt it into four patches and used two models to train. Although the training time is longer, it saves quarter of VRAM used, and the experiment results also show that the effect is also good.

3.2 Dataset

We have chosen to use the Anime Face Dataset[1] (resize to 64x64) as the training dataset for our model. This dataset is particularly well-suited for this purpose because it contains a large number of 64x64 anime face images - 63,632 in total. These images are diverse and provide a rich source of data for the model to learn from. We believe that by using this dataset, our model will be able to effectively learn the unique characteristics of anime faces and improve its ability to recognize and classify them. Additionally, using a large and diverse dataset will allow the model to generalize its knowledge and potentially perform well on a variety of different anime face images. Overall, we are confident that

[1]The link of the dataset: https://www.kaggle.com/datasets/splcher/animefacedataset

Figure 3: Result of our generation.

the Anime Face Dataset (64x64) will be a valuable resource for training our model, leading to more visually appealing results.

3.3 Experiment Setting

All of our experiment use same setting to train. Batch size is 32. And initial learning rate is set as 0.0001. Train for 40 epochs. Using Adam optimizer and MSE loss. For diffusion parameter T and β, we follow vanilla DDPM [HJA20] paper. Our GPU is 1080Ti with 11GB VRAM.

3.4 Experiment Result

The Figure 3 demonstrate the diffusion result of our method. we can see that, the generated anime face is not bad at all. In this case, our method has effectively taken the desired features of an anime face and applied them to generate the new images, resulting in a convincing and visually appealing output. Overall, we are quite pleased with the results of our method and believe it has the potential to be a valuable tool in the field of image generation.

Although we used two models to train, we can see the picture we generated, which is not obvious that this is a combination of two pictures. This is because when in the sample process, we combine the noise predicted of two models by each step, averaging the overlapped part, so we can have such an effect.

The Table 1 shows the detail result of our experiment, we can see that our method PANDA lower the require of VRAM, and FID is not bad at all, which is sampled 10000 images to calculate.The training time has not been extended too much. The reason is that the training pictures have become smaller. In the future, we may consider training more models at the same time to generate larger pictures.

Model	Training VRAM (GB)	Sample VRAM (GB)	Training Time (hr)	FID
Ours	**8.3**	**10.1**	16	28.355
DDPM	10.4	12	**14.6**	**23.567**

Table 1: Experiment result. Bold font means better one.

4 Conclusion

In this project, we purpose a method to train and sample images using two models. This method result in lower consumption of VRAM in once training, and allow we to get high quality images even with the lack of equipment. In addition, we hope that using two models can increase our flexibility, allowing us to better adapt to different application. We believe that this training and sampling method will bring benefits and new insights to our research in the future.

5 Discussion and Future Work

Emphasize again that our original purpose is to generate higher resolution image. But in conventional DDPM, feeding whole image into model causes large VRAM consumption. So we turn to patched method to avoid this issue, even though it need more time to train. However, multi-model method is not our initial expectation. Training one model for each patch is very efficiency less, it also means number of model increasing when we need to split larger image into more patches.

In next stage, we want to try using one model base on current take-average-noise method.

As the Figure 4 show, in each sampled time step, sample 3 continuous patches, concatenate left 2 and right 2 patches as 2 input of model. Then adjust loss function as

$$\mathcal{L} = \rho\mathcal{L}_{left} + \rho\mathcal{L}_{right} + \gamma\mathcal{L}_{overlap}$$

where $\rho = \frac{1}{2}(1-\gamma)$ and γ are hyperparameters. We hope $\mathcal{L}_{overlap}$ this term will force model to output similar predict of overlapping patches.

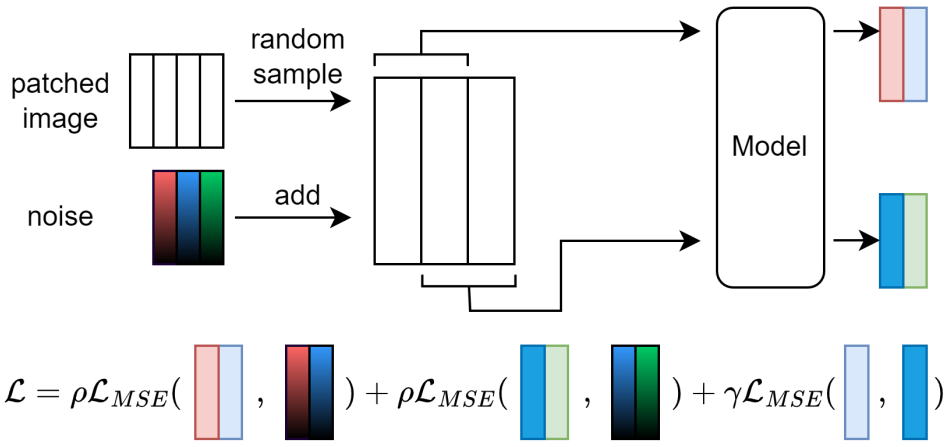

Figure 4: Future work architecture.

References

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.