HW2 Report

Hsuan-Ting Lin

B11901132

1 P1: Conditional Diffusion Models

1.1 Implementation Details and Difficulties of DDPM

I used the UNet structure from this repository [1], which contains the basic structure of a UNet while also having the ability to handle context embeddings. This is useful for classifier-free guidance.

The DDPM implementation adds Gaussian noise to images during the forward diffusion process and predicts the noise using the conditional UNet model with timestep and the 20 class embeddings, representing 10 digits and 2 datasets.

A major difficulty I encountered was during inference, where the score could not meet the baseline, which was unexpected given the relatively simple task. I had to set the conditional guidance very high to allow both datasets barely pass the strong baseline. With different conditional guidance and training iterations, one dataset often scored high while the other dropped significantly (to around 30).

1.2 100 Images for MNIST-M and SHVN

Figure 1: 10 images of each digit for both MNIST-M (left) and SVHN (right)

1.3 Timestep Visualization

The six timesteps I chose to visualize the denoising process are t = 0, 150, 300, 450, 600 and 749, where the total timestep is 750.

Figure 2: Visualization of six denoising timesteps of the first "0" of MNIST-M (left) and SVHN (right)

2 P2: DDIM

2.1 Experiments With Eta

In the figure below, the face images from the leftmost column to the rightmost column are generated from noise files 00.pt to 03.pt. From the first row to the last row are experiments with eta values of 0, 0.25, 0.5, 0.75, and 1.

We can see in the figure that for small eta values, like 0.25 and 0.5, the generated face images bear similarities to those with eta = 0. As eta exceeds these values, the generated images barely resemble the original ones anymore, with the eyes being in the same place at most.

Figure 3: Face images of noise 00.pt to 03.pt with different eta

2.2 Spherical and Linear Interpolation

Figure 4: Visualization of spherical interpolation of face images from noise 00.pt and 01.pt

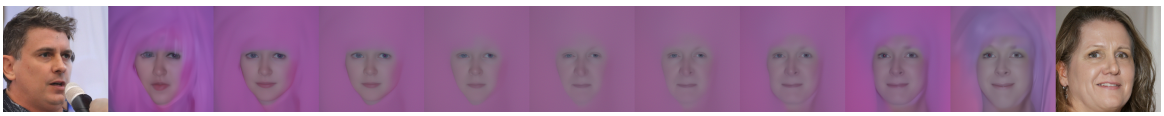


Figure 5: Visualization of linear interpolation of face images from noise 00.pt and 01.pt

In linear interpolation, the generated faces in between are unrealistic and has weird colors. This shows that direct interpolation in the curved latent space may be unfeasble and does not represent valid face images.

3 P3: Personalization

3.1 CLIP Zero-Shot Classification

CLIP performs zero-shot classification by first encoding both the image and a set of potential class labels into a shared embedding space. It then computes the cosine similarity between the image embedding and each class label embedding. Finally, the class label with the highest similarity score is selected as the predicted label. [2]

The accuracy of CLIP zero-shot classification on val dataset is 58.56%, some classification examples are shown below.

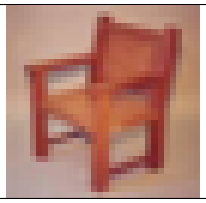
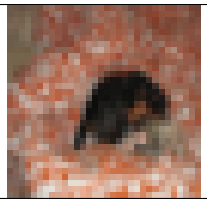
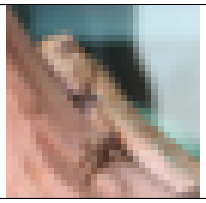
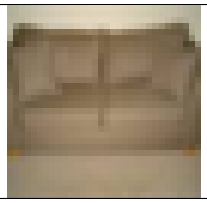
Images					
True Label	chair	possum	lizard	couch	crab
Predicted	chair	possum	lizard	couch	crab
Confidence	0.993652	0.089233	0.075928	0.996094	0.094727

Table 1: Examples of correct classification

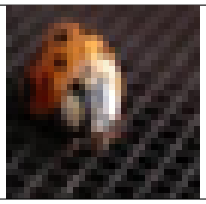
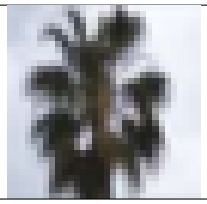
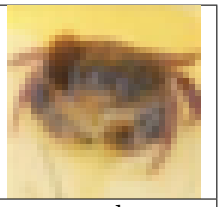
Images					
True Label	dinosaur	skunk	beetle	pine tree	crab
Predicted	elephant	worm	mushroom	palm tree	bee
Confidence	0.945801	0.079712	0.081482	0.968750	0.934082

Table 2: Examples of incorrect classification

3.2 Multiple Concept Image Generation

3.2.1 Multiple Concept Generation Result

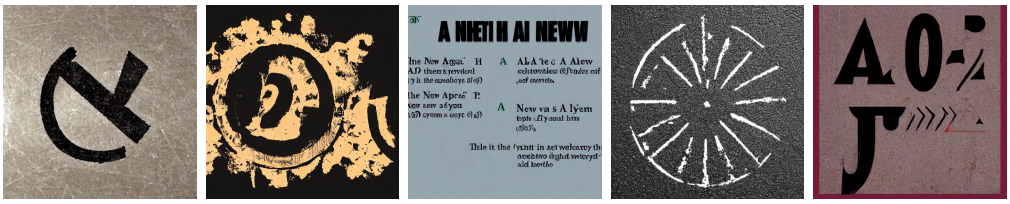


Figure 6: Multi-concept inference from my embeddings

Using the prompt "A <new1> in the style of <new2>.", the generated images are shown above. The images does not show any of the original concepts at all, instead, the images shows some kind of symbol or text in a simple, plain background.

This shows that incorporating multiple personalized concepts may need further fine-tuning or other combination approaches, since the model appears to be treating the placeholder syntax literally rather than understanding it as a template for concept substitution.

3.2.2 Survey on Paper Working on Multiple Concept Personalization

The paper 'Multi-Concept Customization of Text-to-Image Diffusion' by Kumari et al.[3] introduces an efficient method for personalizing text-to-image diffusion models with multiple concepts. Their method, joint training, simultaneously train multiple concepts using different token for each concept and updating only the cross-attention layers of the model. The shared cross-attention layers learn to map different concept tokens to their corresponding visual features, enabling the model to process multiple concepts simultaneously during inference. When given a prompt containing multiple V* tokens, the attention mechanism weighs and combines the learned features appropriately, allowing the diffusion model to generate images that coherently incorporate multiple personalized concepts.

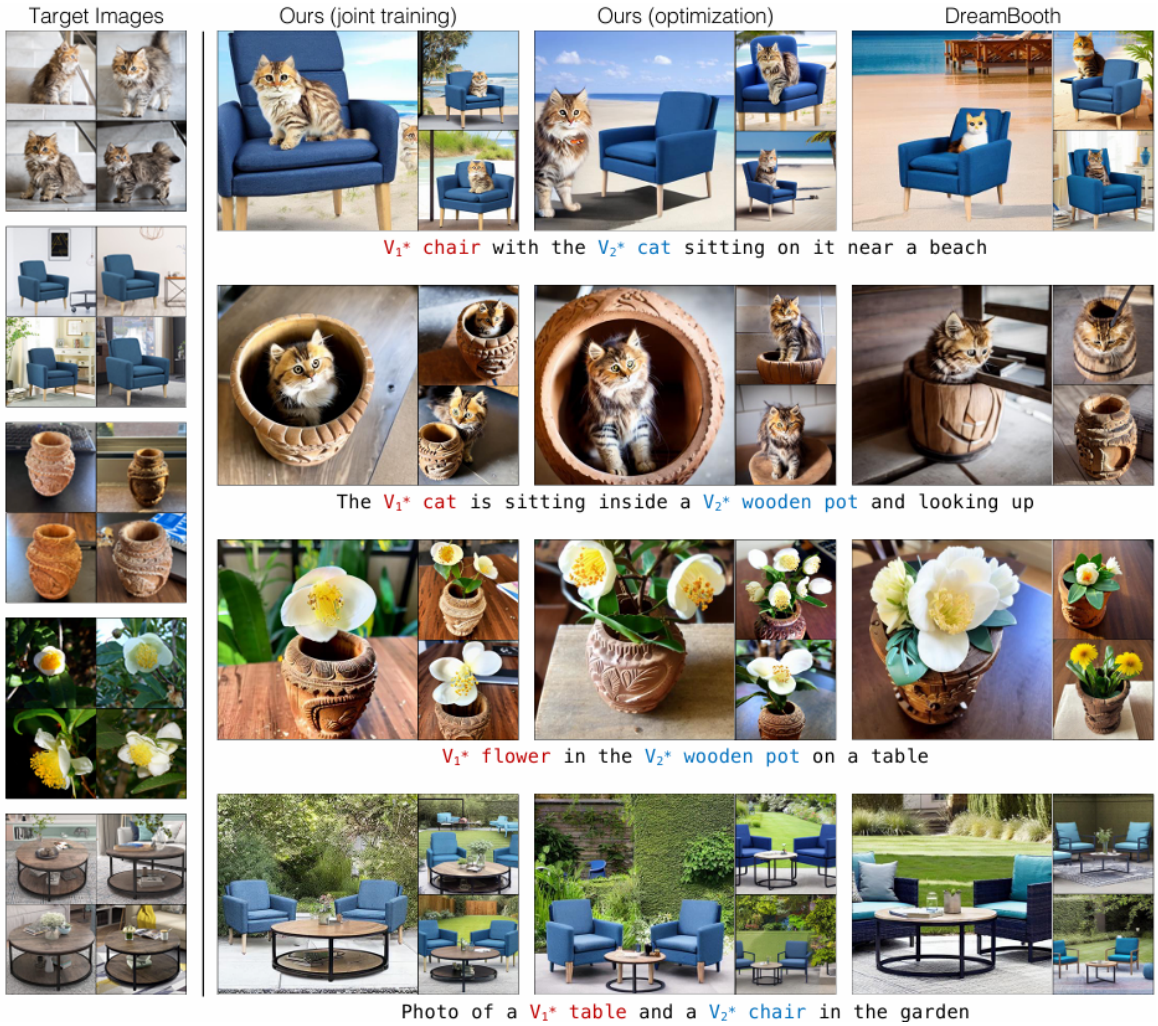


Figure 7: Multi-concept fine-tuning results from the paper

The advantages of this method are notable; it achieves superior efficiency by updating only 3% of model parameters while requiring minimal storage (75 MB per concept) and faster training time (6 minutes) compared to previous approaches like DreamBooth.

Acknowledgments

Some of the code in this project was generated with assistance from `ChatGPT` [4] and `Claude` [5]. Their contributions include code for model training, model evaluation, and general Python scripting.

References

[1] T. Pearce, "Conditional diffusion on mnist," https://github.com/TeaPearce/Conditional_Diffusion_MNIST, 2022, accessed: 2024-10-23.

[2] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," 2021. [Online]. Available: https://arxiv.org/abs/2103.00020

[3] N. Kumari, B. Zhang, R. Zhang, E. Shechtman, and J.-Y. Zhu, "Multi-concept customization of text-to-image diffusion," 2023. [Online]. Available: https://arxiv.org/abs/2212.04488

[4] OpenAI, "Chatgpt: Gpt-4 model," `https://chat.openai.com`, 2024, `https://chat.openai.com`.

[5] Anthropic, "Claude: A large language model by anthropic," `https://www.anthropic.com`, 2024, `https://www.anthropic.com`.