Character Style Transfer with Low-Rank Adaptation of Latent Diffusion Models

Kevin Chu (85406312)*
Department of Electrical and Computer Engineering
The University of British Columbia
Vancouver, BC V6T 1Z4
cchu19@student.ubc.ca

Abstract

Since the fast development of latent diffusion models, text prompts have become a crucial factor for image synthesis. However, generating complex images is often accompanied by complex text prompts, which makes it difficult for people to intuitively understand. Such a sophisticated relationship between text prompts and images hinders the widespread adoption of image-generating applications.

This project aims to use the low-rank adaptation (LoRA) method to train customized adapters based on pre-trained models to solve tasks in a specific category. We take the character style transfer as the topic, to explore whether the LoRA method and its applications have the potential to solve the hard prompts problem. The results show these lightweight adapters make the character more consistent with the target styles under the same prompt conditions.

This project provides experimental test results, comparing between different styles, while we can understand the potential of LoRA technology in solving the hard prompt problem and its performance in character style transfer.

1 Introduction and Background

The rise of diffusion models has made image-generative AI grow at an incredible speed, these state-of-the-art technologies gradually change the way how people work. Among these tasks, image style transfer has been an important topic in the digital industry. Traditional image style transfer methods, such as montage techniques and manual digital painting, require significant manpower and resources. This project chose the image transfer of a video game character as the topic, not only to discuss the potential of LoRA technology in solving the hard prompt issue but also to present this topic in practical tasks and scenarios. If the problem of hard prompt is successfully solved, it will undoubtedly make image style transfer more popular than nowadays, while more people will be willing to use it for production. The combination of generative AI and image style transfer will have a significant impact on various industries, including entertainment and art. By enhancing product quality and reducing costs, this technology offers substantial market value and potential.

2 Related Work

The related works of this project are divided into two sections with the related algorithms and the related applications. We explain the related techniques and models for our task while listing out the possible applications for image style transfer.

*https://github.com/FalKon1256/UBC-EECE-570

2.1 Related Algorithms

2.1.1 Denoising Diffusion Probabilistic Models (DDPMs)

The denoising diffusion probabilistic models were proposed by Ho et al. [1], elaborating on a new concept of generative models for high-quality image synthesis from denoising a random signal. This approach involves initially adding a Gaussian distributed noise to image data and later training the model to learn how to denoise the resulting image by predicting the noise at each step. By learning the patterns from the images with different levels of noise, the Kullback–Leibler divergence between the predicted and ground truth image can be reached to a minimum, which helps the model generate closer results towards the provided image. The backward process of removing noise guided by the Markov chain is the foundation theory of diffusion models.

2.1.2 Latent Diffusion Models (LDMs)

Latent Diffusion Models, introduced by Rombach et al. [2], extend the principles of DDPMs, applying the training operation in the latent space of pre-trained autoencoders to enhance computational resources rather than in the pixel space. In addition, by introducing cross-attention layers into the model structure, the image outputs can be controlled by inputs of text or bounding boxes. This significantly enhanced the efficiency and fidelity of diffusion models, making them to be capable of completing complex tasks with reduced computational consumption.

2.1.3 Low-Rank Adaptation (LoRA)

Low-rank adaptation was first proposed by Hu et al. [3], offering a new concept of fine-tuning large pre-trained models without retraining all model parameters. This method freezes the model weights and inserts rank decomposition matrices with lesser trainable weights into each layer in the transformer. The performance through LoRA training can reach or surpass traditional fine-tuning methods, keeping the training process resource-efficient.

Style transfer through LoRA training was conducted in different researches. Liao et al. [4] performed one-shot transfer learning through LoRA to transfer Chinese calligraphy art styles to other characters and symbols, such as English letters and digits. Shrestha et al. [5] managed to transfer any given image into the style of Calvin and Hobbes comics through LoRA training on the Stable Diffusion model. The LoRA fine-tuning technique is effective and versatile for various style transfer tasks across specific topics and objects.

2.2 Related Applications

Image style transfer technology has a wide range of applications and can greatly accelerate project progress, especially in art-related tasks. This technology has potential and commercial value in any industry, such as advertising, publicity, and art production. In this section, we use film and game production as examples to demonstrate the significant benefits that image style transfer technology offers from the perspective of industries.

2.2.1 Film Production

Film production is highly related to fine art, especially remakes often face the challenge of style transfer. Traditional production processes, such as draft production, actor selection, and scene construction, require substantial financial, human, and time resources. This has forced some projects, which have the potential to become great works, to compromise under these constraints. For example, if a film production team invests heavily in art drafts, they may have to choose actors or create lower-quality scenes within a limited budget.

If image style transfer technology is supported by generative AI, it can solve many current difficulties in remake movies. The production team can use the original illustrations and text descriptions as training materials to quickly obtain abundant style-transferred images. These images not only serve as important references for characters and scenes but also help the team save a huge amount of budget while the tasks can still be completed accurately in a short time. This allows the team to put resources into key aspects of the production, such as stunning scene construction and cast selection.

As image style transfer technology becomes more widespread with generative AI, it will not only optimize the production process of remake movies but will also be expected to bring about a revolution in the entire film industry. In a future where generative AI technology matures, even small-scale production teams with limited budgets can produce excellent works, and Hollywood-level film production will no longer be the privilege of large teams or companies.

2.2.2 Game Production

Game production involves a lot of artwork, which is highly related to the programming team, the modeling team, and even the publicity team. Art tasks have a significant impact on the entire product production process. From the early art concept draft production and character design to the later stage of modeling and programming, consensus and consistency must be maintained. If the art concept or character design is not properly resolved in the early stage, it may compress the timelines for other tasks, leading to delays in product release, quality issues, or cancellation of the entire development project. However, it is challenging to produce a large number of high-quality drafts of different styles in a short period. The process of decision towards the art concept is lengthy and costly, but it is an essential part of game production.

Image style transfer technology can solve this problem. Art workers only need to take their artworks as training materials for the model, assisting the model to quickly and accurately generate a large number of high-quality images of different styles of the same building, character, or scene. These images can also provide inspiration for improvements to the artwork.

As every aspect of game production is closely connected, image style transfer technology will significantly impact on game production. In the future, as generative AI matures, smaller teams will no longer be restricted by tight financial budgets in art creation and will be able to produce more games with outstanding graphics and unique styles. Large companies will be able to reduce their cost of art creation, scene production, and character creation, while they can focus on enhancing the gameplay, which is the core element of game creation.

3 Experiment Setup and Results

The training for LoRA adapters is faster and requires fewer datasets, which means the quality of chosen datasets impacts the training and results significantly. Firstly, we discuss the dataset, including data collection and pre-processing. The training implantation and testing are then explained upon the utilization of the dataset in detail.

3.1 Dataset

The dataset contains 20 selected images (in JPG data form) and the corresponding captions (in TXT data form) of "Geralt of Rivia", who is a well-known character from the series "The Witcher". These images were mostly sampled from the game "The Witcher 3: Wild Hunt". The creation of the dataset includes data collecting and pre-processing.

3.1.1 Data Collection

All images were from the website "Wallhaven" at `https://wallhaven.cc`, which is one of the greatest sources of high-quality images. 35 images were first selected with different features of "Geralt of Rivia", e.g. his face, only upper body, full body, different poses, different angles, etc. Since the final dataset for LoRA training should include enough information about the character with limited data, the 20 images were carefully picked, which contain 7 close-ups, 10 upper-body, and 3 full-body images as the final dataset.

3.1.2 Data Pre-processing

All images from the dataset were resized, cropped, and transformed into the JPG format at Birme (`https://www.birme.net`), where these tasks were conducted in batches. Images were resized to 512 x 512 pixels dimension, which is consistent with the size of the training images for Stable Diffusion v1-5 (`https://huggingface.co/runwayml/stable-diffusion-v1-5`), the base model that we use in this project. Unrelated elements, such as figures and complicated objects, were cropped

geralt of rivia, 1boy, 1 sword on back, looking at viewer, realistic, chainmail, pauldrons, ash, white background, serious face, upper body, unsheathing, holding sword

geralt of rivia, 1boy, tree, looking at viewer, chainmail, bare tree, outdoors, half body, belt, gauntlets, standing, pauldrons, fire, smoke, serious face, 2 swords on back, realistic

Figure 1: This figure shows samples of the dataset images and captions.

off to have "Geralt of Rivia" be the main part of each image. As a low number of images for LoRA training, we conduct this process to alleviate the impact of overfitting during the training.

All captions were first auto-generated by "WD14-Tagger" (`https://github.com/picobyte/stable-diffusion-webui-wd14-tagger`), a labeling extension for Automatic1111's Web UI (`https://github.com/AUTOMATIC1111/stable-diffusion-webui`). To enhance the quality of each caption, we manually added additional captions and deleted inappropriate generated captions for all 20 images with the dataset tag manager "Booru" (`https://github.com/starik222/BooruDatasetTagManager`). We set the trigger word for our LoRA to be "geralt of rivia" and added this word to each training caption, which means it is expected to have a strong effect on LoRA when entering the trigger word as part of the prompt. With all 20 images and captions created (see Fig. 1), we put them all into the same folder for training.

3.2 Training

The main idea of LoRA training is to freeze the base model's weights and insert trainable layers, which are matrices for rank decomposition, in each transformer block. Stable Diffusion v1-5 was chosen as the base model and not all weights were trained with the LoRA technique. All weights can be divided and stored as matrices with far fewer weights to train in total. We provide detailed descriptions of our environment settings, implantation, and results.

3.2.1 Environment

This training was run on the hardware system with CPU as AMD Ryzen 7 7800X3D (8-core processor) and GPU as NVIDIA GeForce RTX 4070 (12g VRAM), the training process takes around 40 minutes to complete. Important used libraries include PyTorch v2.2.2 (CUDA v12.1), Diffusers package (`https://huggingface.co/docs/diffusers/en/index`), Kohya_ss package (`https://github.com/bmaltais/kohya_ss`), Parameter-Efficient Fine-Tuning (PEFT), and xFormers.

The Diffusers package provides various functions to create pipelines for our training and testing, while the Kohya_ss package and PEFT provide crucial configurations and scripts for LoRA training. In addition, xFormers optimizes the attention blocks to alleviate GPU memory consumption and increase the inference and training speed.

3.2.2 Implantation

We selected and used the existing training framework of standard LoRA training from the Kohya_ss library, this script file is called "train_network.py", and we customized the parameters for this project's training process. We list the main training parameters as follows:

pretrained_model_name_or_path: We selected "Stable Diffusion v1-5" as the base model that we want the LoRA adapter to apply to.

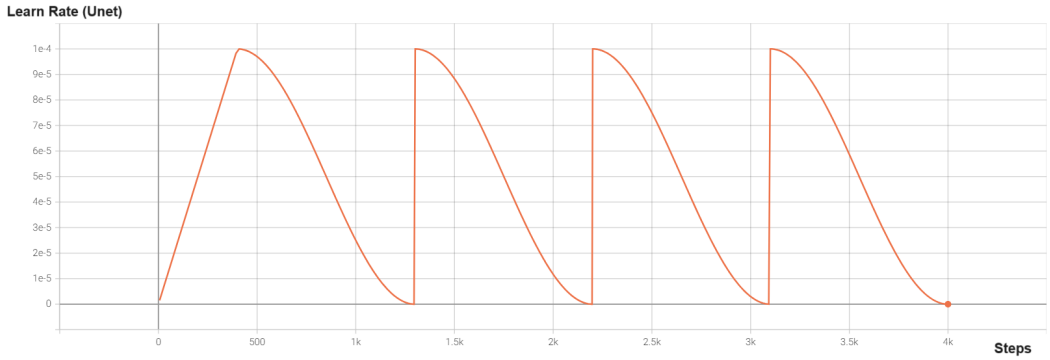

Figure 2: This figure shows the warm-up stage and learning rate cycles of Unet.

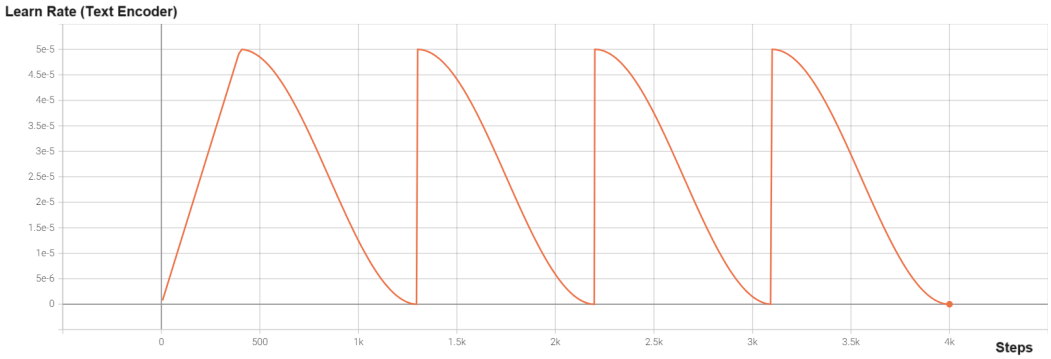

Figure 3: This figure shows the warm-up stage and learning rate cycles of text encoder.

mixed_precision: We selected "fp16" to implement half-precision data types during the training, this can reduce the memory usage and increase the performance.

resolution: We selected "512 x 512" since the base model Stable Diffusion v1.5 was trained with this size of images. We keep the consistency with our dataset.

seed: We set the seed number to reduce the randomness, improving the reproducibility of a certain image result. However, randomness still exists to some extent because of random cropping.

cache_latents: We turned on "cache latents" to compress the images into smaller latents stored in VRAM as cache. This increases the speed of the training.

unet_lr: We set as default value 0.0001 for LoRA training with the U-Net learning rate on the inference process (denoising and feature extraction).

text_encoder_lr: We set a default value of 0.00005 for LoRA training with the text encoder learning rate. The text encoder is the CLIP model (CLIP ViT-L/14) in Stable Diffusion v1-5.

loss_type: We selected "l2" to activate the L2 loss (MSE loss) mode for the Huber loss. The model can be less sensitive to small variances but very responsive to reducing large errors under the L2-loss behavior. This is crucial for noise reduction.

lr_scheduler, lr_scheduler_num_cycles, lr_warmup_steps: These parameters are for adjusting the learning rate to prevent overfitting and underfitting cases during the process. We selected the scheduler "cosine_with_restarts" for cyclical change of the learning rate with 10% of the total steps as warm-up steps (400 steps) and 4 periods of cycles during the training (see Fig. 2 and 3).

train_batch_size: We set it to 2 for processing 2 images in the same batch. A high value of this parameter can result in significant consumption of GPU memory.

max_train_steps: We set it to 4000 steps, determining the total epoch. This is because the training script reads the dataset folder name and takes the prefix "100" as the steps for each image. Since having 20 images and a batch size of 2, we had 1000 steps per epoch (100 x 20 / 2 = 1000). Since the maximum number of steps is 4000, we end up with 4 training epochs in total.

save_every_n_epochs: We set it to 1, saving middle LoRA checkpoints for each epoch. This allowed us to compare them in the testing.

network_dim: As this is the dimension of the LoRA network, we selected a higher value to let the LoRA adapter learn as many features as possible. The main reason is that we filtered the images and unrelated elements were fewer in our dataset.

network_alpha: "Alpha" is a parameter for the regularization term, and also a penalty term. We set it to a high value as we only provide 20 images, intending to prevent overfitting and expecting the decision boundary to have lesser curvatures. This is because smaller weights tend to fix high variance when increasing alpha.

optimizer_type: We selected "AdamW8bit" as the optimizer, as it decouples weight decay from the gradient updates and applies directly to the weights. It also reduces the bit width from 32-bit float points to 8-bit integers, resulting in better stability and faster training.

huber_c: We set this value to "0.1" since we use the Huber loss function for our training. The value acts as an error threshold for switching the behavior of mean squared error (MSE) when the error is small and mean absolute error (MAE) when the error is large. This pattern helps reduce the sensitivity to outliers in data that might influence the total loss significantly by shifting the loss function from quadratic to linear in large error values.

huber_schedule: We set this to "snr" so that the threshold is adjusted dynamically based on a signal-to-noise ratio during training. This scheduling can enhance the learning process and speed up the convergence.

3.2.3 Results

Our training result (see Fig. 4) shows the LoRA training gradually changed the weights, leading to slow but stable updates with more effective learning and a subsequent loss decrease. It is important to observe the first 400 to 500 steps, while the loss is increasing and not stable. The main reason is that we applied a warm-up process for the learning rate during the first 400 steps (10% of total steps). The parameters are sensitive to the starting received gradients, and starting with a smaller learning rate can make a positive impact on further training. The parameters should learn slowly and adapt without overwhelming the patterns that already exist in the base model. The loss dropped from the maximum value of 0.1582 at around the 500 step to 0.0896 at the 4000 step, which comes to the end of the training. The threshold of Huber loss was set to 0.1, while we see similar behavior from the loss curve before and after this threshold, which means outliers in data did not affect too much during our training. This shows our training process is relatively stable and worked well, while the Huber loss reduced gradually for 90% of the training process.

3.3 Testing

Four testing cases were conducted in this project, including applying a single LoRA adapter, multiple LoRA adapters, a LoRA adapter with pre-trained models, and controlling inference steps. The hyperparameters of image width and height (512 x 512), guidance scale (7.0), and clip layer skipping (2) were kept the same for all test cases. The inference steps were set to 50 except for test 3.3.3 case 3 and test 3.3.4 (set to 100). All positive prompts, negative prompts, and seed values were the same within each test case. All LoRA checkpoint adapters (lora-1, lora-2, lora-3, lora-4 for 1, 2, 3, 4 epochs trained) were used in all tests except only lora-4 was used in test 3.3.4. The LoRA weights were set to 0, 0.25, 0.5, 0.75, and 1.0 for each test except it was fixed to 1.0 in test 3.3.4.

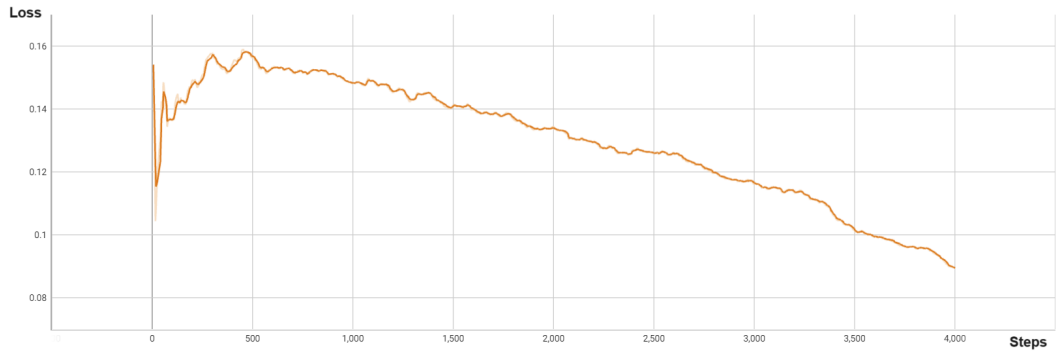

Figure 4: This figure shows the average loss during the training stage.

3.3.1 Single LoRA adapter

We compared the generated images before and after applying the LoRA-trained adapters to the base model Stable Diffusion v1-5. We set the positive prompt to be "geralt of rivia" with no negative prompt. The seed value was 1 and the inference steps were 50 for all image outputs.

The test result (see Fig. 5) shows that the LoRA adapter was adding more of the features of Geralt to the character in the image, making it closer to the true image of Geralt, such as the hairstyle, two swords on his back, the unique necklace, face scars, etc. However, overfitting can be observed when the LoRA weight is near 1.0, which shifts the original image to be very similar to one of the images in our dataset. We conclude that a better weight of LoRA should be controlled between 0.25 to 0.75, in order to keep the original elements and add important features for Geralt.

3.3.2 Multiple LoRA adapters

We compared the generated images of a pixel-style LoRA merging with the LoRA adapters we trained, then we applied each merged adapter to the base model Stable Diffusion v1.5. The pixel-style LoRA adapter was from CivitAI (`https://civitai.com/models/44960/mpixel,filename:pixel_f2.safetensors`), which can generate decent pixel-style images with this LoRA adapter. The mixing ratio was 1:1 for our adapters and the pixel-style adapter. We set the positive prompts to be "geralt of rivia, pixel" with no negative prompt. The seed value was 924608315 and the inference steps were 50 for all image outputs.

We can observe more correct features when LoRA weights are increasing (see Fig. 6). In addition, some significant features did not appear, such as the swords or the necklace, while overfitting is not evident at LoRA weight of 1.0. This implies that merging multiple LoRA adapters can dilute some features that should be important. Changing the ratio of the mixing and observing what is missing can be a good indicator to modify the dataset for future training. Moreover, mixing multiple LoRA adapters is also a method to reduce the visual effects of overfitting.

3.3.3 LoRA adapter with pre-trained models

We apply LoRA adapters on three pre-trained checkpoint models to show the ability of image style transfer that LoRA has. The three styles were anime, comic, and realistic style, all three pre-trained models were based on Stabel Diffusion v1.5. The anime-style (`https://civitai.com/models/4468/counterfeit-v30?modelVersionId=57618`, file name: Counterfeit-V3.0_fix_fp16.safetensors), comic-style (`https://civitai.com/models/35960/flat-2d-animerge?modelVersionId=266360`, file name: flat2DAnimerge_v45Sharp.safetensors), and realistic-style (`https://civitai.com/models/4201/realistic-vision-v60-b1?modelVersionId=130072`, file name: realisticVisionV60B1_v51VAE.safetensors) checkpoint model can all be found on CivitAI.

For the anime style test, we set the positive prompts to be "1boy, geralt of rivia" with no negative prompt. The seed value was 629389646 and the inference steps were 50 for the outputs. The testing result (see Fig. 7) has shown ideal LoRA weights were between 0.5 and 1.0 in anime style.

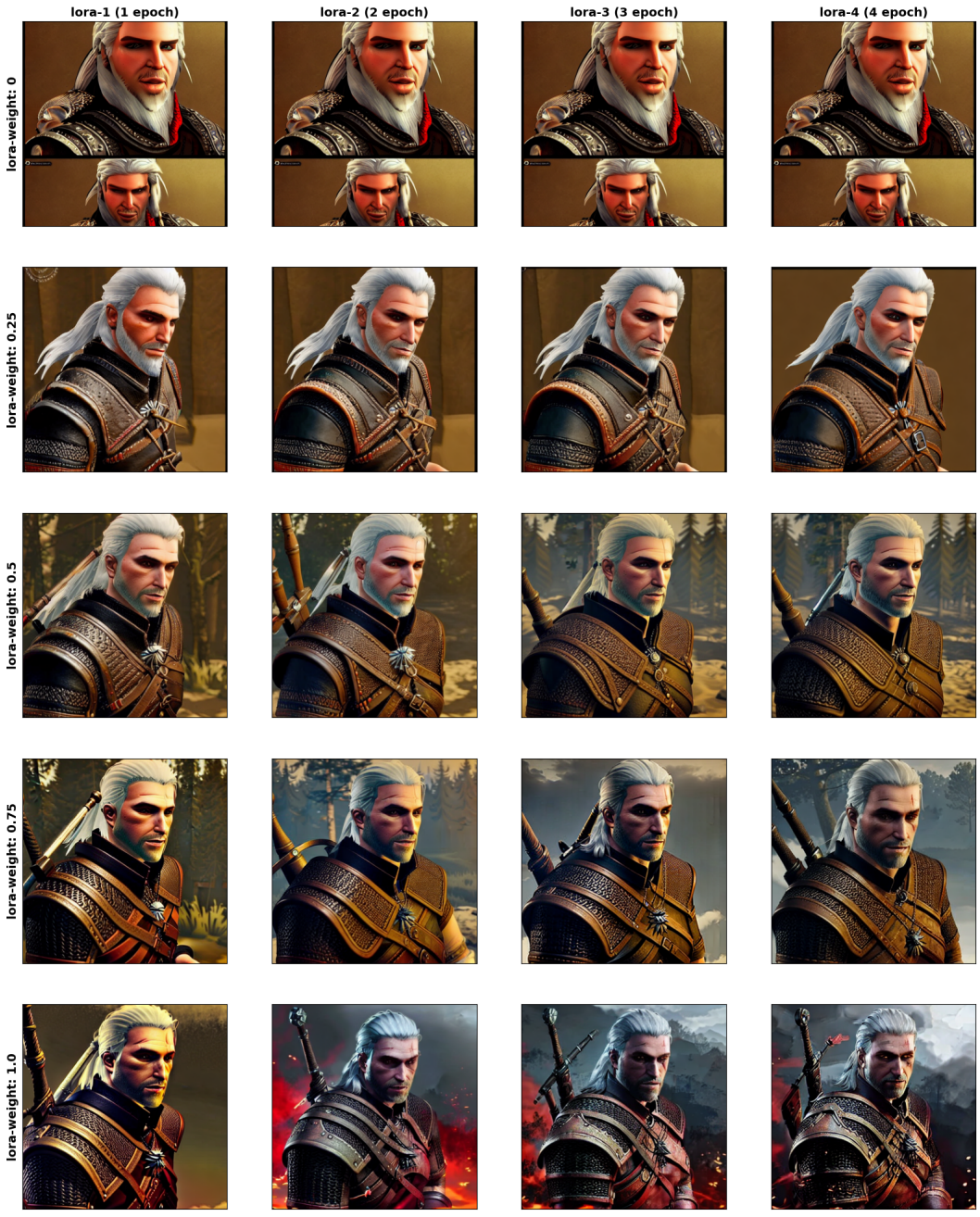

Figure 5: This figure shows the test results of applying our trained LoRA adapters to the Stable Diffusion v1-5 model.

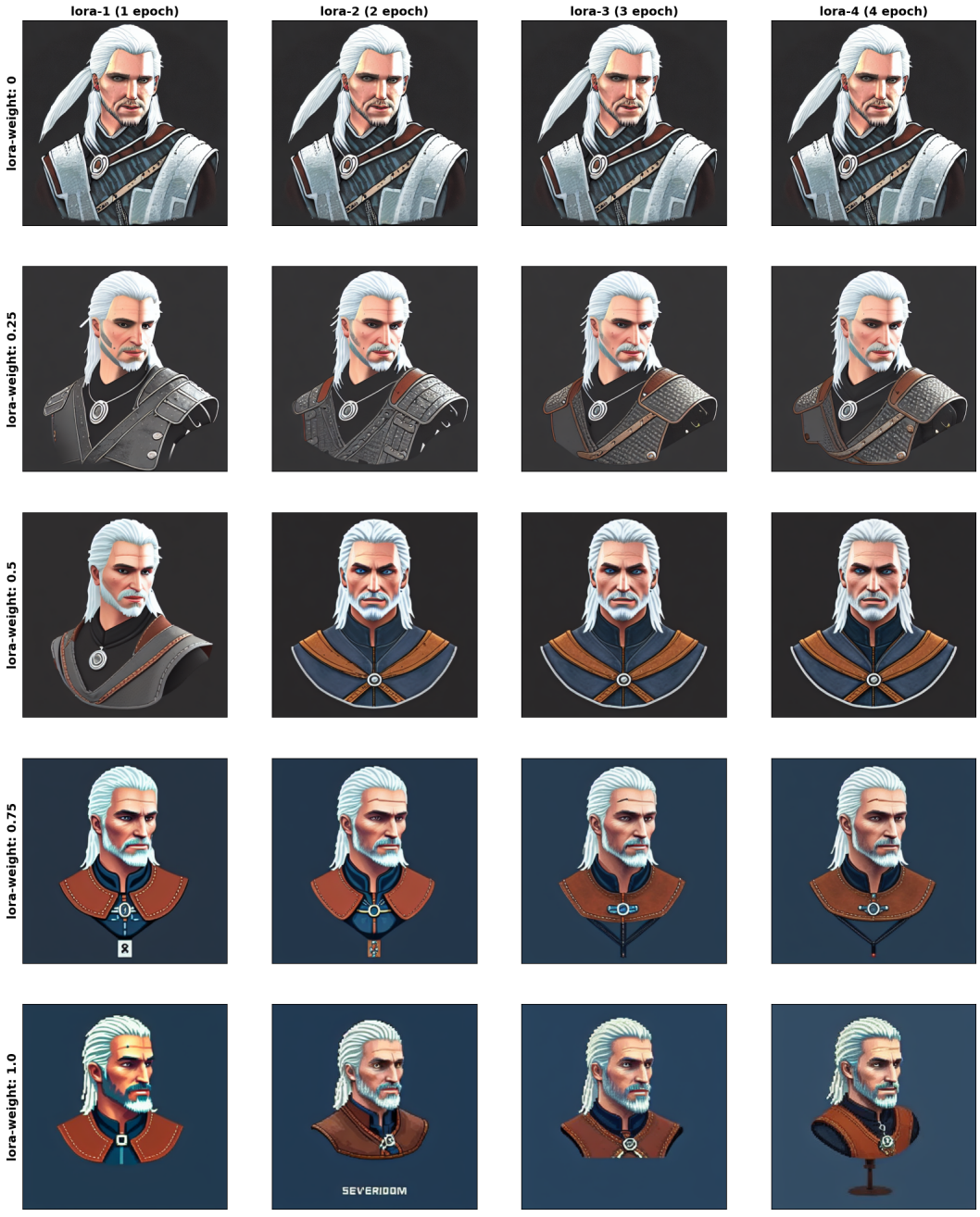

Figure 6: This figure shows the test results of merging our trained LoRA adapters with a pixel-style LoRA adapter and applying it to the Stable Diffusion v1-5 model.

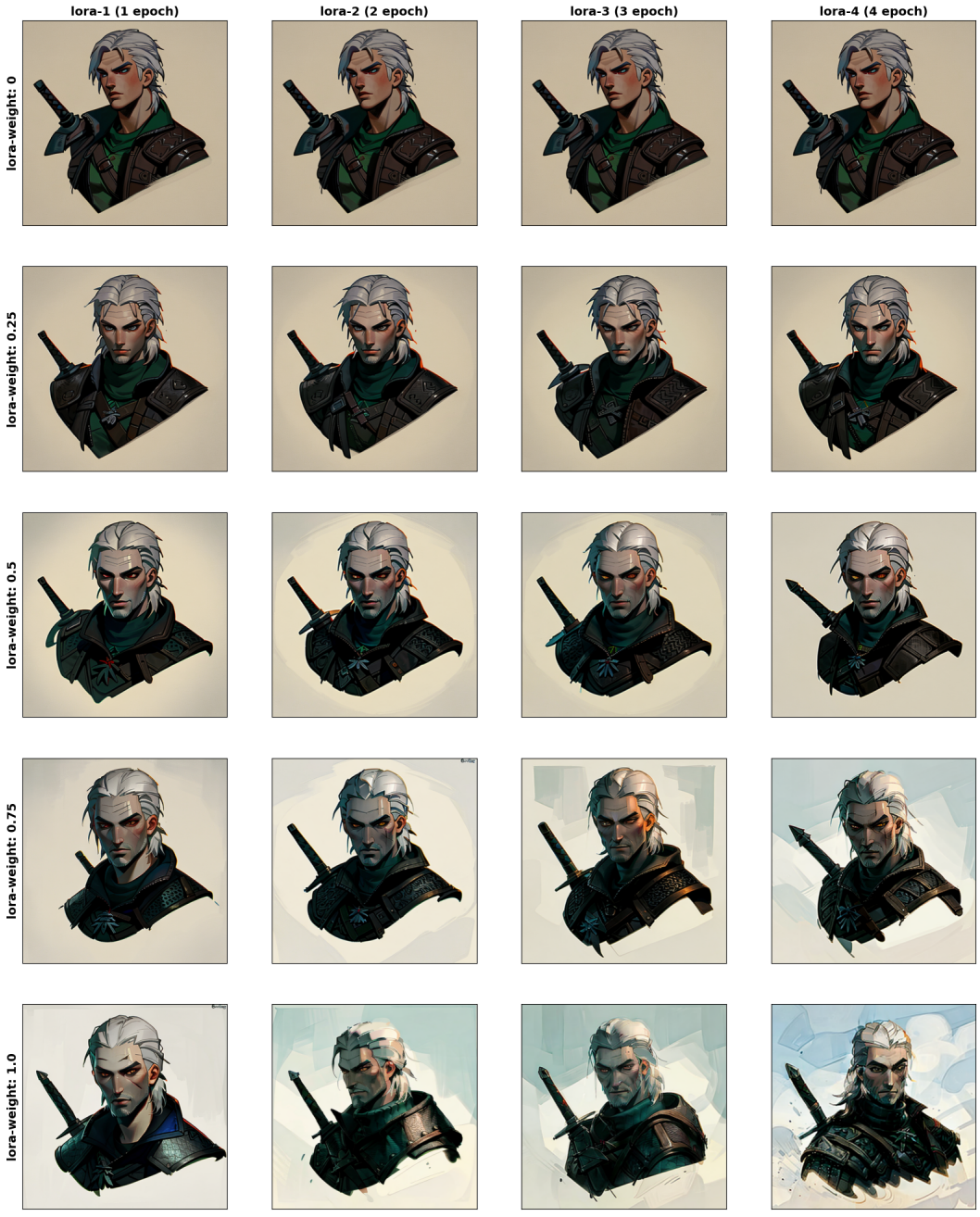

Figure 7: This figure shows the test results of applying our trained LoRA adapters to an anime-style pre-trained model.

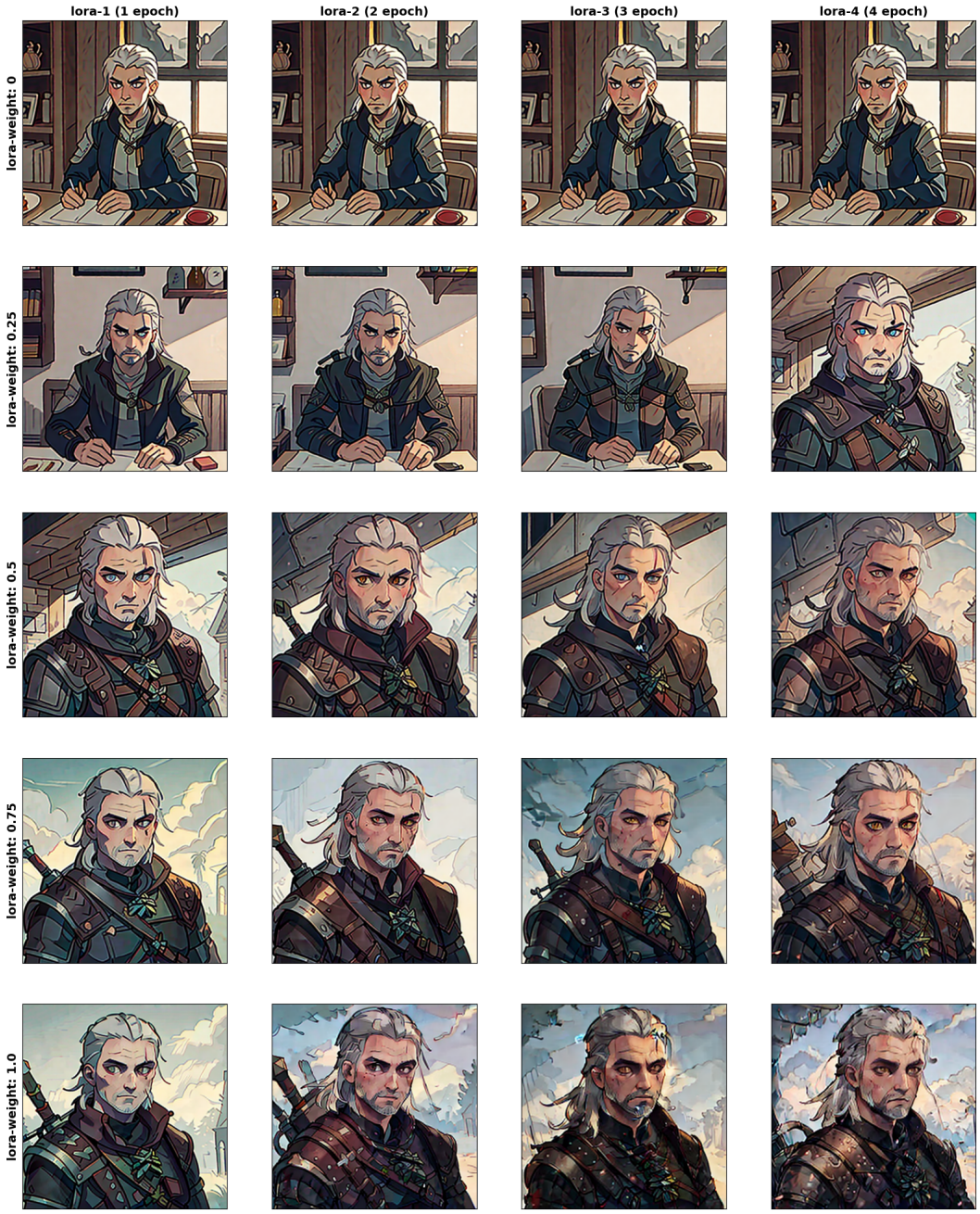

Figure 8: This figure shows the test results of applying our trained LoRA adapters to a comic-style pre-trained model.

For the comic style test, the positive prompts were "geralt of rivia, 1boy, masterpiece, best quality" while the negative prompts were "lowres, bad anatomy, bad hands, text, error, missing fingers, extra digit, fewer digits, cropped, worst quality, low quality, normal quality, jpeg artifacts, signature, watermark, username, blurry", which we added some standard prompts for quality enhancement in this case. The seed value was 3330428146 and the inference steps were 50 for the outputs. The testing result (see Fig. 8) has shown ideal LoRA weights were between 0.5 and 1.0 in comic style.

For the realistic style test, the added prompts were complicated as we followed the author's document of the pre-trained model. The positive prompts were "1boy, geralt of rivia, full body, RAW photo, subject, 8k uhd, dslr, soft lighting, high quality, film grain, Fujifilm XT3", while the negative prompts

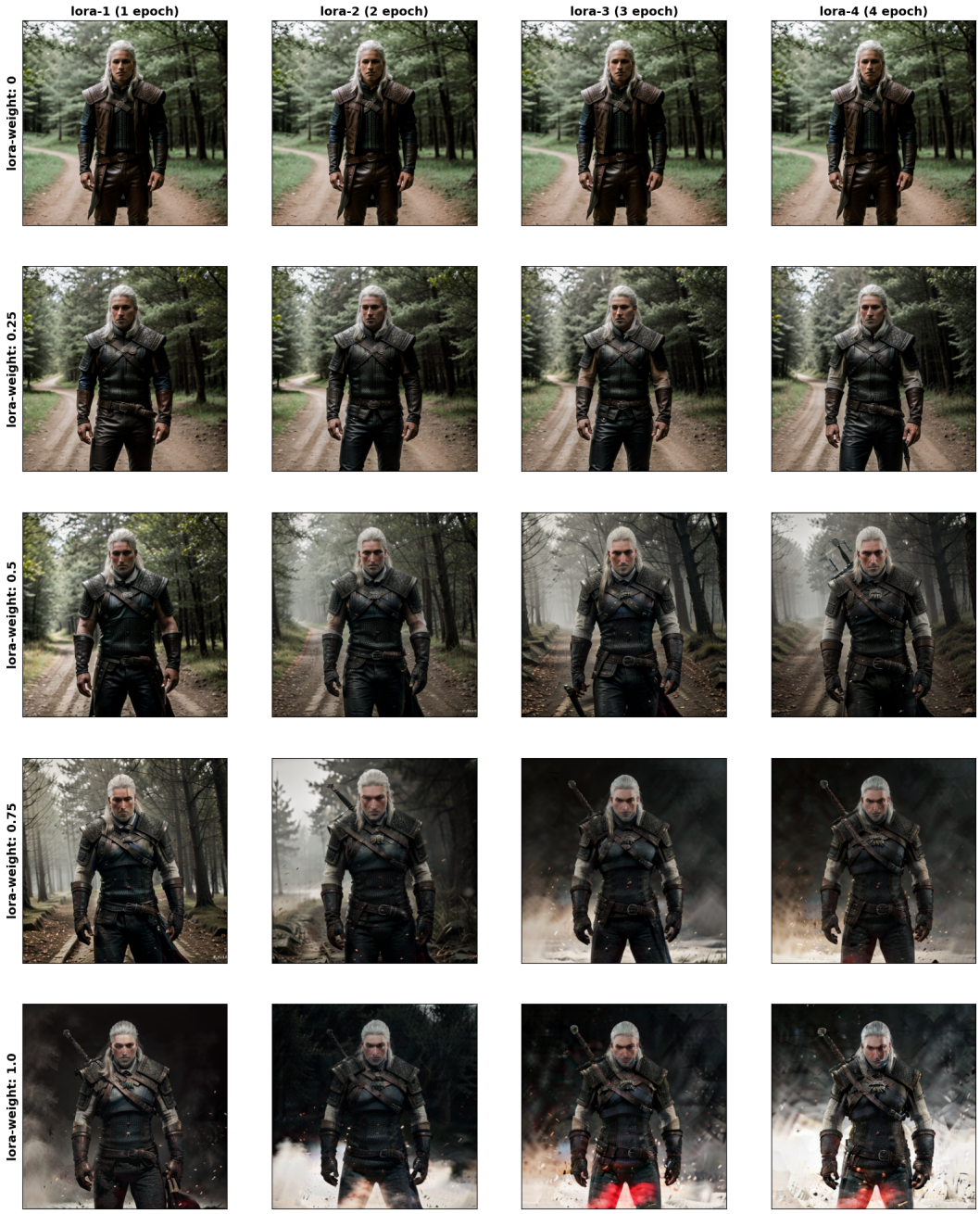

Figure 9: This figure shows the test results of applying our trained LoRA adapters to a realistic-style pre-trained model.

were "deformed iris, deformed pupils, semi-realistic, cgi, 3d, render, sketch, cartoon, drawing, anime, mutated hands and fingers, deformed, distorted, disfigured, poorly drawn, bad anatomy, wrong anatomy, extra limb, missing limb, floating limbs, disconnected limbs, mutation, mutated, ugly, disgusting, amputation". The seed value was 3599032598 and the inference steps were 100 for the outputs. The inference steps were set to 100 for high-quality requirements of realistic-style images. The testing result (see Fig. 9) has shown ideal LoRA weights were between 0.5 and 0.75 in realistic style.

For all three cases, the method of applying LoRA to checkpoint models can significantly transfer the style for Geralt. We observe that the visual effects between the three styles can be very different.

Figure 10: This figure shows the test results of changing the inference steps by applying the lora-4 adapter (weight = 1.0) to a realistic-style pre-trained model.

Generally, better output images are generated when LoRA weights are between 0.5 to 0.75. In addition, when the LoRA weight is 1.0, the anime and comic style has less distortion and overfitting in the images, but the realistic style is badly affected by such a high value of LoRA weight. It is clear to observe distortion starting at LoRA weight of 0.5 and overfitting at LoRA weight of 0.75 to 1.0. Realistic-style images also require more inference steps to process the complicated pixel layout of the image. This implies the LoRA weights should be optimized for different categories of styles, as the large difference between the two styles varies the optimized weight input for LoRA adapters.

3.3.4 Inference steps control

We found out how many inference steps are crucial when generating complicated images, while low inference steps cause generated images to have massive noise. In this test, we selected lora-4 (trained with complete 4 epochs) which was the adapter trained with the highest epochs and set the LoRA weight to its maximum value 1 to challenge the minimum inference steps we need to generate a complicated realistic-style image without noise residue. We reused the settings in the realistic-style test, while only changing the inference steps to 30, 40, 50, 60, and 70 steps.

From the test result (see Fig. 10), we learn the inference steps should be set between 50 and 60 for optimization, as high inference steps mean more GPU computing resources and time are required.

It is an important topic to decide the trade-off for tasks related to generating images as the limitation of the GPU. We test the realistic-style images since they require more inference steps for their high quality than other styles in anime and comic style, in which the pixel layout is relatively simple. However, the final decision for images of character style transfer still depends on what kind of visual effect is the style pursuing.

4 Challenges, Limitations and Future Directions

From this project, we learned that LoRA technology can solve the hard prompt problem and can be applied to the task of character style transfer. In this section, we explore the current challenges and limitations of the LoRA training method, which include intellectual property and privacy issues, limitations of dataset pre-processing, and overfitting issues. In response to these issues, possible future directions are proposed as an end of this project.

4.1 Intellectual Property and Privacy Issues

Challenges and Limitations: Nowadays, generative AI technology has been questioned over intellectual property rights and privacy issues. Taking generated pictures as an example, the concerns are not only about whether the images may cause infringement on copyrights but also the privacy leakage from the training data. In addition, whether generated images are protected by copyrights will be a controversial topic in the future.

Future Directions: Taking diffusion models as an example, if the algorithm of the scheduler can be adjusted to add watermark patterns that are imperceptible by the human eye without reducing the image quality, this will provide an effective way for determining generated images. By aligning this approach with clearly defined legal boundaries of copyright infringement in the future, the risk of infringement on creators' rights can be reduced, while the AI tools that meet these standards will be trusted. In terms of privacy, combining federated learning and relevant regulations to train and update

models by end-to-end weight exchanging instead of directly providing datasets can be a feasible solution to privacy concerns.

4.2 Dataset Pre-processing limitations

Challenges and Limitations: In data pre-processing, the LoRA training technique requires high-quality images and accurate captions as training datasets. However, defining the captions precisely may be challenging for some images, which may result in more costs of financial, time, and labor for the dataset preparation and pre-processing.

Future Directions: Using generative AI models or tools to solve this problem is a promising future direction. For example, some models can complete the task of converting text to high-quality images nicely, providing a large number of high-quality training images for the dataset. In terms of caption generating, although such tool exists, they still need to be improved for precision. Further development on image-to-text models can be a solution while making these models into automated captioning tools. This can be one of the promising directions to address the problems of dataset preparation for LoRA training.

4.3 Overfitting Issues

Challenges and Limitations: The LoRA adapters are trained for specific tasks. Although they do not require a large number of datasets, their versatility is lower compared to directly trained large models. The LoRA training may encounter overfitting easily with a small and singular dataset provided.

Future Directions: Adding a larger number of images that are not directly relevant to the original dataset for regularization can reduce the possibility of overfitting and improve the versatility of the LoRA adapter. However, determining the proper number of regularization images and the elements that should be included or avoided is a challenge for this future direction.

Currently, LoRA is still a cutting-edge technique for model fine-tuning, while various methods have been derived from LoRA, such as LyCORIS (LoHa/LoCon) (Yeh et al. [6]). These latest training methods will become important indicators for the future development of image-generative AI.

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[2] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[3] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[4] Qisheng Liao, Gus Xia, and Zhinuo Wang. Calliffusion: Chinese calligraphy generation and style transfer with diffusion modeling. *arXiv preprint arXiv:2305.19124*, 2023.

[5] Sloke Shrestha, Asvin Venkataramanan, et al. Style transfer to calvin and hobbes comics using stable diffusion. *arXiv preprint arXiv:2312.03993*, 2023.

[6] Shih-Ying Yeh, Yu-Guan Hsieh, Zhidong Gao, Bernard BW Yang, Giyeong Oh, and Yanmin Gong. Navigating text-to-image customization: From lycoris fine-tuning to model evaluation. In *The Twelfth International Conference on Learning Representations*, 2023.