Custom Exercise: Low Rank Adaptation of Flux-Dev

Geppner, Philipp ; Time Needed: approx. 50 hours

February 2025

1 Introduction

Generative diffusion models have recently gained attention for their capacity to transform textual descriptions into high-quality images [1]. Adapting these large models to incorporate new visual concepts without resorting to full retraining poses a significant challenge [5]. This exercise introduces the use of Low Rank Adaptation (LoRA) as a memory-efficient method to fine-tune a pre-trained diffusion model for the purpose of integrating a specific visual concept - in this case the properties of a specific person, namely the author [4]. The problem addressed is how to inject a novel entity into the latent space while preserving image fidelity and prompt adherence. The aim is to explore the impact of varying key hyperparameters, such as adapter rank, captioning detail, and training image resolution, on the quality and consistency of the generated images.

2 Theoretical Background

2.1 Low Rank Adaptation (LoRa)

Low Rank Adaption, originally introduced by Hu et al. [4] et al for Large Language Models, is a memory efficient way to fine-tune larger, transformer based deep learning models. LoRa integrates trainable rank decomposition matrices into each attention layer to adapt the (frozen) pretrained weights of a model. This allows for easy sharing and experimentation of the resulting "adapters", due to the much smaller footprint compared to a finetuned model - they are often only a few megabyte in size. The community around diffusion models has quickly adapted the LoRa approach to facilitate the introduction of out-of-training styles, concepts and entities.

2.2 Generative Diffusion Models

Generative diffusion models work by gradually corrupting data through a forward process that incrementally adds Gaussian noise, transforming complex data distributions into a simple prior [3, 6]. A neural network is then trained to reverse this process by estimating the score function and iteratively denoising the noisy data—this reverse process is often formulated as solving a reverse-time stochastic differential equation (SDE) or its deterministic ordinary differential equation (ODE) counterpart [1]. This two-step mechanism offers a stable training objective and high-quality sample generation, despite requiring a more computationally intensive iterative sampling procedure compared to other generative models.

3 Experimental Setup

A general note: All images, samples and code that are not explicitly shown can be found in the provided project files. I tried to reduce the length of the documentation a bit this way.

3.1 Hardware

Everything for this exercise was done on my local machine. The main specifications are:

1. **RAM**: 32GB DDR4 3600MHz
2. **CPU**: Ryzen 9 5950X 16 Core 32 Thread
3. **GPU**: NVIDIA RTX 3090 24GB VRAM

Although the base flux-dev model barely fits on the 24GB Vram, there are a few quantization and cpu-offloading options available now. For example, **flux-gym** provides an option for 12, 16 and 20 GB VRAM training.

3.2 Available Tools

Although there are already a few tools available for easier LoRa training, all of them are early-stage software and therefore quite flaky and unreliable. For the exercise I tried out two tools **ai-toolkit** [1] and **flux-gym** [2]. For inference I used the base **comfy-ui** [3] workflow for flux-dev. Figure 1 shows part of the easy to use UI of flux gym.

All of them require a functioning virtual python environment with torch and quite a few required auxilliary packages. If the training is done on local hardware, as is the case here, a functioning cuda environment is also needed to train on the GPU. In addition to the software itself, the models for the UNet, VAE and Text Encoder need to be downloaded from huggingface. Fluxgym generates a training script behind the scenes, containing all the settings needed for training.

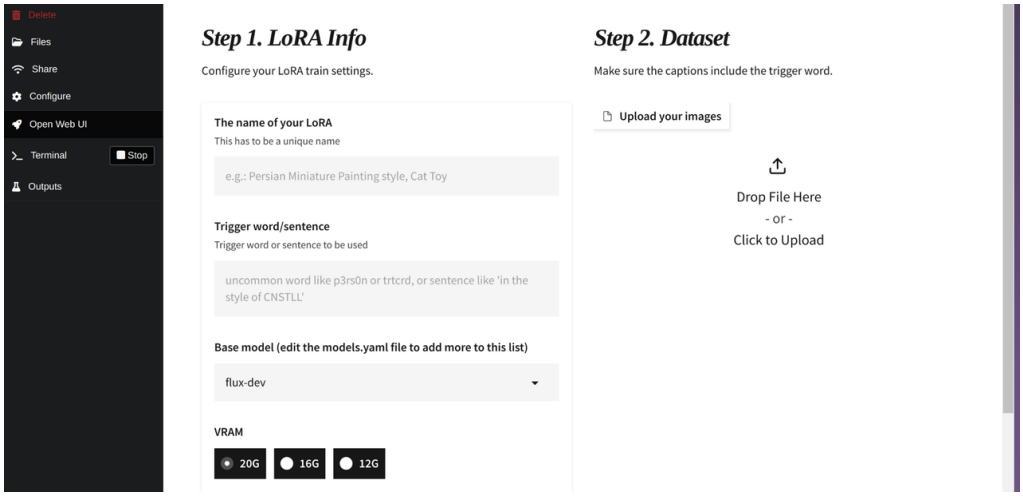

Figure 1: Shows part of fluxgyms WebUI. Lets one setup training parameters, such as number of training steps, Rank of the LoRa matrix and training images.

3.3 Data

The dataset used for the personal lora, introducing the concept of *me, Philipp Geppner*, into the latent space is quite minimal and consists of mostly selfies made from different angles and some minimal outfit variations.

[1] https://github.com/ostris/ai-toolkit
[2] https://github.com/cocktailpeanut/fluxgym
[3] https://github.com/comfyanonymous/ComfyUI

3.4 Test Sample Prompts

To evaluate the model checkpoints during and after training, three different test prompts are used to generate validation samples. The first one is just *<phigep>* to check what the model does with high degrees of freedom. The second one depicts a very specific but still believable and probable scene, *<phigep>* sitting on a park bench wearing pink crocs. The third test prompt is supposed to test adherence in case of a more challenging and improbable scene, with it being *<phigep> playing chess in space*.

3.5 Model

All LoRa adapters are trained for the Flux.1 Dev model, the mid-tier version of the Flux *rectified flow transformer* model family, introduced by black forest labs[4]. As of the time of this exercise, the flux models are recognized as state-of-the-art image generators. It has 12B parameters and is supposedly based on a hybrid architecture of multimodal and parallel diffusion transformer blocks. It is trained using *flow matching*, that learns a time-dependent velocity (or drift) field which continuously transforms a simple "base" distribution into a more complex "target" distribution. By matching the probability flow between these two distributions at each time-step, it is supposed to achieves stable training and efficient generation with fewer evaluations than other diffusion methods.

3.6 LoRas trained

To at least get an idea of the influence of the various training parameters. The following settings are used to train different LoRas:

Parameter	**Values**
Training Image Resolution (1:1)	512, 1024
Adapter Rank	4, 16, 32
Captioning	"Just Keyword", "Basic Position and Background (Florence-2)", "Detailed Object and Background (GPT-4o)"
Training Epoch Eval	2, 4, 6, 8, 12, 14, 16

Table 1: Hyperparameters and their corresponding values, Training Epoch Eval shows at which intervals the LoRas where evaluated with test prompts. Each Epoch takes 200 steps of training.

3.7 Evaluation Metrics

This section describes the various metrics implemented to assess the quality and consistency of the generated images. The evaluation framework combines both objective and subjective measures, ensuring a comprehensive analysis.

3.7.1 CLIP Score

The **CLIP score** leverages the pre-trained CLIP model (`openai/clip-vit-base-patch16`) to measure the semantic alignment between an image and its textual description. In the implementation, images are preprocessed (converted from BGR to RGB, normalized, and reshaped) and the input prompts are modified by replacing the custom token `<phigep>` with a descriptive phrase ("young male person with brown hair"). The CLIP score provides an objective metric that quantifies how well the generated image corresponds to the prompt, without the inclusion of the specific new entity (me). This metric was added to complement the LLM-based evaluations by offering a direct, model-based assessment of image–text alignment. [2]

[4] https://blackforestlabs.ai/announcing-black-forest-labs/

3.7.2 LLM-Based Evaluation Metrics

Several evaluation functions interact with a large language model (LLM) to assess specific aspects of the generated images. These include:

- **Person Detection:** Evaluates whether the image depicts a person or multiple people, returning a score of 0 (not a person) or 1 (a person).
- **Same Person Consistency:** Compares a base image with a generated image to determine if they depict the exact same individual, based on facial features, hair color, hair length, eye color, and other visual attributes.
- **Prompt Adherence:** Measures how closely the generated image follows the given prompt. This evaluation takes into account both the accurate depiction of the designated person (as indicated by the custom token `<phigep>`) and the overall scene description. **This is the most vital LLM metric, as it combines general scene adherence with the adherence to our new concept.** Exemplary Code is shown below on how the metric is implemented.
- **Image Quality:** Assesses the overall visual quality of the generated image. The LLM is also provided with a Canny Edge version of the image to try and spot any inconsistencies in shape.
- **Artifacts:** Assesses if any larger visual artifacts occur due to the generation process.

Each of these LLM-based evaluations is performed by sending structured prompts—often including both the original image and its Canny edge version (encoded as base64 JPEGs)—to the LLM. The model then returns a score between 0 and 1 for each criterion. For brevity of this exercise, not all results are visualised in this short summary. Most of the evaluation is done regarding the prompt adherence and image quality. Afterall, the most important factor of evaluating this finetuning method, is whether a completely new concept can be taught to the model.

3.7.3 Example Code: Prompt Adherence

The code snippet in listing 3.7.3 demonstrates the functions used to evaluate how closely a generated image adheres to a given prompt. it also shows one of the functions used to evaluate image quality, in this case the tournament version used for the image resolution experiment.

4 Experiments: Entity LoRa

General Note: Due to time and compute restrictions, the experiments are not as quantitatively rigorous as I would have liked. However, I made sure to keep all other values fixed and compare the influence of a single "variable" per experiment.

4.1 Qualitative Experiment: Demonstrating LoRa Finetuning Effectiveness

Here we simply compare the results on the same prompt and generation seed of the base model with that of a LoRa finetuned model. Figure 2 shows that the base model has no idea who the entity is, LoRa finetuned model does.

4.2 Experiment: Effect of Captioning Detail-Level

To evaluate the influence of the image captioning detail, three contrasting captioning strategies are conceived. This experiment shows the influence of what and how much of the portrayed entity

Listing 1: Prompt Adherence and Image Quality Evaluation Code

```python
def eval_prompt_adherence_llm(base_image, generated_image, generation_prompt):
    completion = client.beta.chat.completions.parse(
        model=MODEL_NAME,
        messages=[
            {
                "role": "user",
                "content": [
                    *construct_user_text_message(
                        f"The first image contains a real image of <phigep> (custom token for the person). "
                        f"The second is a generated one based on the following prompt:\nPROMPT:{generation_prompt}\n"
                        "Rate the adherence to the prompt and return a single score between 0 and 1. "
                        "A score of 1 indicates perfect adherence to the prompt."
                    ),
                    *construct_message_for_image(base_image),
                    *construct_message_for_image(generated_image),
                ],
            }
        ],
        response_format=EvalScore,
    )
    return completion.choices[0].message.parsed
def eval_image_quality_tournament(image1, image2):
    completion=client.beta.chat.completions.parse(
        model=MODEL_NAME,
        messages=[
            {
                "role": "user",
                "content": [
                    *construct_user_text_message(f"Evaluate both attached Images based on Image Fidelity, Image Resolution overall Quality and Contrast. Give it a score from 0 to 1, where 0 means you totally prefer the first image and 1 means the second image is of way higher quality."),
                    *construct_message_for_image(image1),
                    *construct_message_for_image(image2),
                ],
            }
        ],
            response_format=EvalScore,
    )
    return completion.choices[0].message.parsed
```

Figure 2: Shows the effect of LoRa finetuning on the example of learning the concept of a specific person. (Philipp Geppner). As we can see, the non-finetuned model has no valid concept of the "<phigep>" token.

is linked to the concept in latent space. The first strategy was to simply caption each image with just the trigger word - in this case "<phigep>".

The second strategy was a simple and short image captioning, done with Florence-2 - a vision capable foundation model with under 1B parameters.

The third and most extensive image captioning was done with GPT-4o, capturing both minute details of the person, the expression, as well as the background and mood.

Keyword Only Captioning The LoRa with simple trigger word captioning learns, due to the consistency of the persons outfit across most images, even the black, graphical tshirt as part of the concept <phigep>. This makes sense if the outfit itself is part of the concept to be learned. It also overfits in the sense, that scenes, expressions and lightning are all consistent with the low variability in the input images.

Florence-2 The Florence-2 Captioned LoRa on the other hand has learned the concept of the person, but linked the black tshirt and background to the additional captioning tokens as opposed to the trigger word. This more detailed approach makes sense in general, because normally the persons abstract concept is not intricately linked to their clothing.

Extensive GPT-4o Captioning The third approach, where even the details of the person itself are described by additional tokens, results in an inability to learn or link the right information to the trigger word <phigep>.

For this comparison, the LoRas Rank is kept at 4 and the resolution at 512 x 512 pixel.

The results in figure 3 could indicate, that the captioning detail itself, is a good parameter to tune. The right level of detail depends on what the LoRa *should* capture. This experiment seems to imply a subtractive approach to captioning: Apart from the trigger token, mention everything that is *not* supposed to be part of the learned concept. The very descriptive captioning with GPT-4o results in the model struggling to subscribe meaningful visual characteristics to the trigger token. The keyword only captioning on the other hand appears to subscribe scene and person to the trigger token, angles, background, and even lighting conditions appear similar to the input samples.

4.3 Experiment: Effect of LoRa Matrix Rank

To test for the influence of the adapter size on various metric, the florence captioned lora was trained as a rank 4, rank 16 and rank 32 matrix. Figure 4 shows a selection of the LLM-based metrics over the training course for rank 4 and rank 16.

Both models show gradual improvements in prompt adherence and limb quality throughout training. Image Quality remains stable, probably due to the leniency of the LLM judge, as mentioned below for the input resolution experiment. Same Person consistency fluctuates more, especially in Rank 4, indicating less stability in character identity. In general, rank four seems to

Figure 3: Shows the effect of captioning details on the learned concept on **generated images** during the first 1000 steps of training, in 200 step increments. The first row is GPT-4o captioning, which describes the concept and scene both in detail. The second row is the simpler but still informative description by Florence-2. The last row has only the trigger token <phigep> as captioning.

have struggled with the very open-ended trigger-only prompt. Evaluation of concept adherence ("same Person"), as well as prompt adherence align well for the trigger-only prompt, which makes sense as it should be identical in this case. For the park bench prompt, performance remained stable and showed a slight steady increase of the training steps, only the limb quality was judged to be lower at the beginning of training for rank 4.

The chess in space prompt, which really challenges probable known outputs, both for the concept of Philipp Geppner, as well as the unlikely scene, rank 4 and rank 16 have quite contrasting training runs. It also shows a clear sign of overfitting on the concept, as the concept adherence ("Same Person") gets better and is initially aligned with general prompt adherence.However, they later on diverge with a stronger focus on the concept and loss in general prompt adherence. This again reinforces the importance of not just a diverse dataset, but also diverse testing prompts to detect signs of concept overfitting and loss of general prompt adherence.

Even though the LLM metric is again too "lenient", upon visual inspection it is noted that image quality and resolution seem to change with the rank of the adapter matrix. To analyze this further an average FFT representation for r4 and r32 LoRa output images was calculated respectively. As it appears to be incremental, the differences between r4 and r32 are visualised in figure 5. The images generated by rank 32 adapter seem to on average have sharper edges and directional lines as signified by the stronger line patterns in frequency domain.At the same time, r4 hase in general higher magnitude for higher frequencies, suggesting that the r4 images contain more high-frequency noise or artifacts, leading to a less structured appearance. In contrast, the r32 images exhibit clearer directional features, likely due to better preservation of fine details and textures. This might indicate that higher-rank adapters improve frequency coherence, reducing diffusion while enhancing edge sharpness and structure.

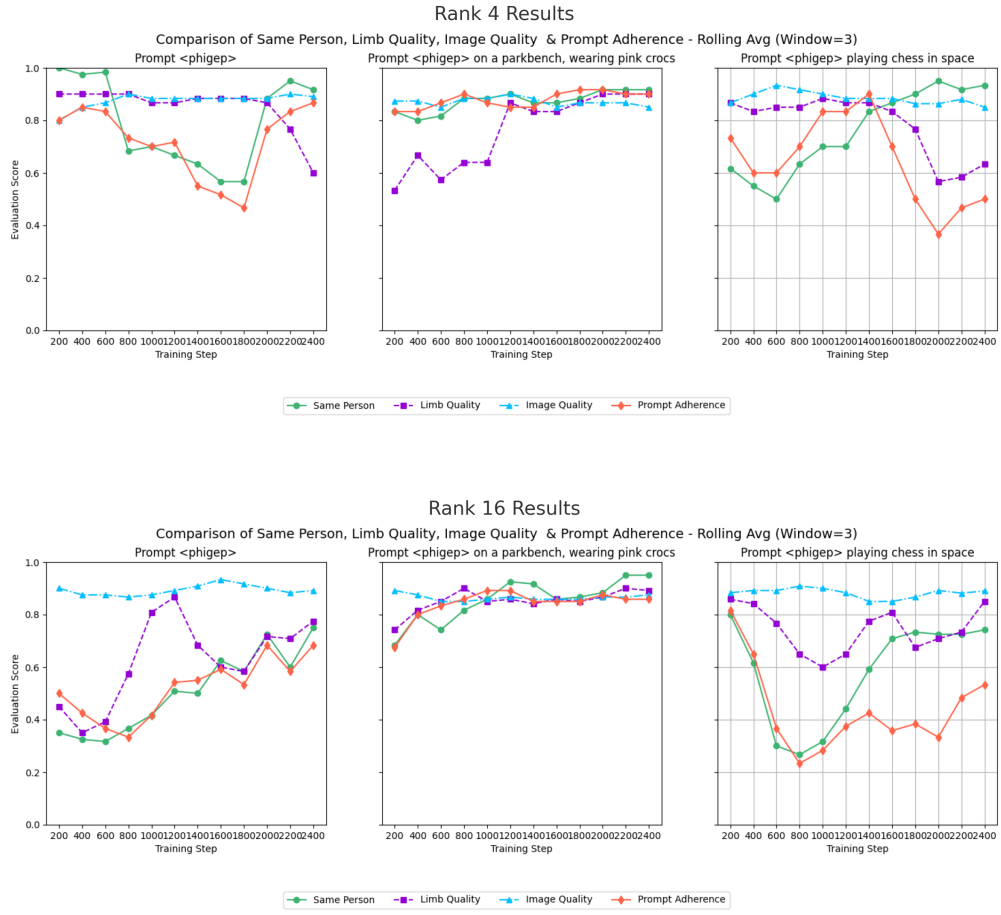

Figure 4: Shows the difference between the training runs of rank 4 LoRa and rank 16 LoRa for the LLM based metrics. The Image Quality metric is again seemingly not critical enough and stays at a high level throughout.

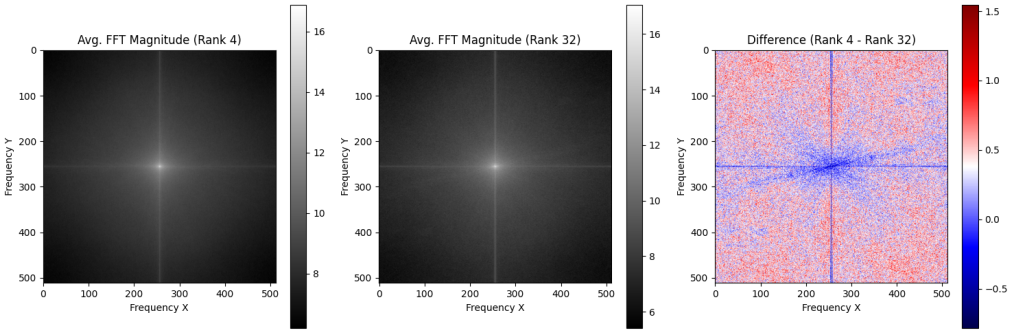

Figure 5: Shows the difference between the mean representation of rank 4 LoRa output and rank 32 LoRa output. It appears as if, edges get sharper on both the horizontal and vertical axis. It also appears that a higher rank has less high frequency content and the opposite for "lower" frequency content.

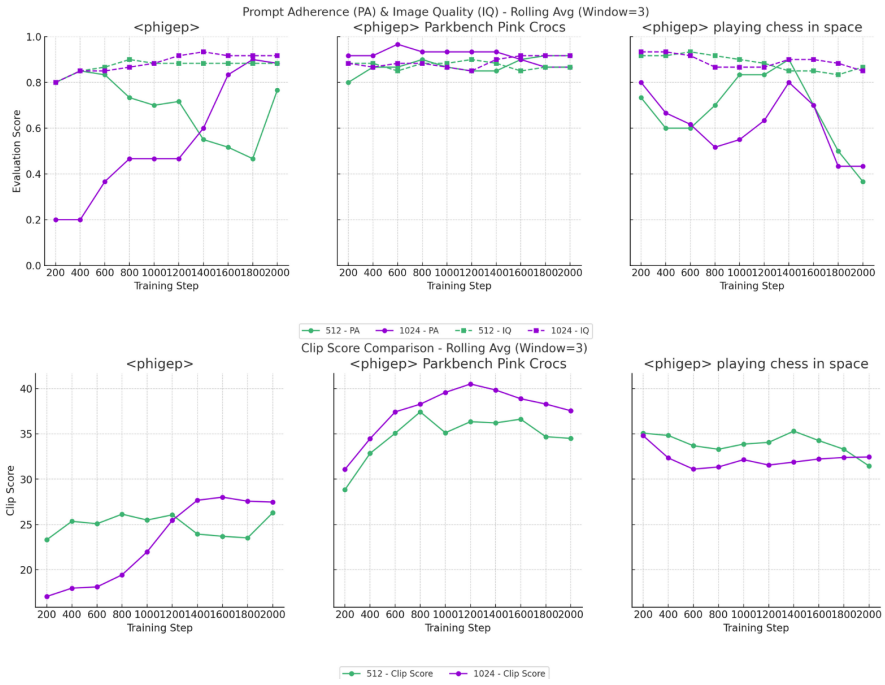

Figure 6: Comparison of Prompt Adherence (PA) & Image Quality (IQ) with Rolling Avg (Window=3) for different prompts and input image size during training. Dotted Lines represent 512x512 images, uninterrupted lines show metrics for the 1024x1024 input. The second row shows the CLIP score, which in general aligns well with the PA score, but fluctuates less and shows a clearer change over time.

4.4 Experiment: Effect of Training Image Resolution

While the setting for image resolution changes the input images, the output at inference always stays at 512x512. In theory, the detail quality that is learned by the Adapter, should increase with the resolution of the examples. An LLM-as-a-Judge Approach is used to rate prompt adherence (PA) and (perceived) Image Quality (IQ). Additionally a popular metric, CLIP-Score is used to check consistency with the prompt adherence LLM-Approach. Figure 6 shows the scores for each prompt. For visualisation purposes only, the results were smoothed with a rolling average of windowsize 3. The CLIP Score and PA score generally agree quite well. The image quality score seems to not differentiate well or is too lenient. For both the simple trigger word prompt and the plausible prompt in the park, CLIP score and PA score showed that the 1024x1024 LoRa model was able to better adhere to prompt instructions. For the implausible chess in space prompt, the other model had a slight edge on adherence.

Of all the modifications, the increase in input image resolution from 512x512 to 1024x1024 had the biggest qualitative increase in output quality. The prompt based image quality score does not seem to capture this well, likely due to the fact, that it already scores the lower resolution images so high and does not evaluate them in contrast. For this reason, another custom metric was implemented that randomly takes two images, one from each of the two LoRas and asks an LLM, which one it prefers based on resolution, image quality and fidelity. Figure 7 shows a clear preference for images that were generated using the larger input images. This tournament style metric seems to align quite well with the preferences of the author.

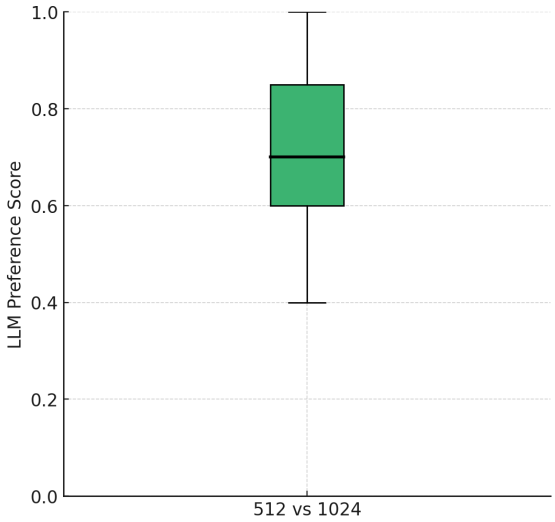

Figure 7: LLM Arena Tournament style evaluation of image fidelity and quality. A LLM judge has to decide between images from the 512x512 and an image from the 1024x1024 LoRa. If it prefers the first the score is closer to zero, if it prefers the latter, the score is closer to one. It shows a clear preference for the LoRa using 1024x1024 pixel images as input.

4.5 Magnitudal Artifacts

The longer the training, the more likely were noticeable distortions in body proportions. Overfitting on the basis of LoRas also seems to manifest in different, sometimes quite comical concept "magnitude" effects. For example, hair seems to get longer, heads seem to get bigger and even the *number of people depicting the same learned concept* seems to get higher. Figure 8 shows such effects.

It bight be, that specific low rank matrix positions corresponding to <phigep> get very high in value and the magnitude seems to affect the notion of *"more"* in different ways.

5 Summary and Discussion

Apart from generally showing that a LoRa adapter is able to finetune a model and teach it very specific concepts, such as a concrete persons looks, 3 additional experiments were conducted to evaluate the influence of certain hyperparameters. In this chapter the results are again summarized and further interpreted.

LoRa: Does it Work? Yes, a small adapter trained on under 20 images of a concept is able to teach a diffusion model, in our case FLUX.1-Dev a specific concept. It allowed for a (more or less) visually faithful generation, the LLM scores for concept adherence ("Same Person Score") and total prompt adherence were generally relatively high for the high performing check points. This can be seen in figure 4 and 6. However, further human evaluation would be needed to check whether the ("good") generated images are similar enough in their depiction of the author to fool someone. Also test sample diversity during the training is important, as the differences across prompts on the given metrics (again in the figures mentioned before) are sometimes higher and/or even in opposition when trying to determine which checkpoint model performed the best.

Overfitting on a concept seems to happen during longer training runs as well. An interesting aspect that could be further explored is the way overfitting manifests in generated images. Apart from the expected loss in generation diversity, some magnitude effects appear to

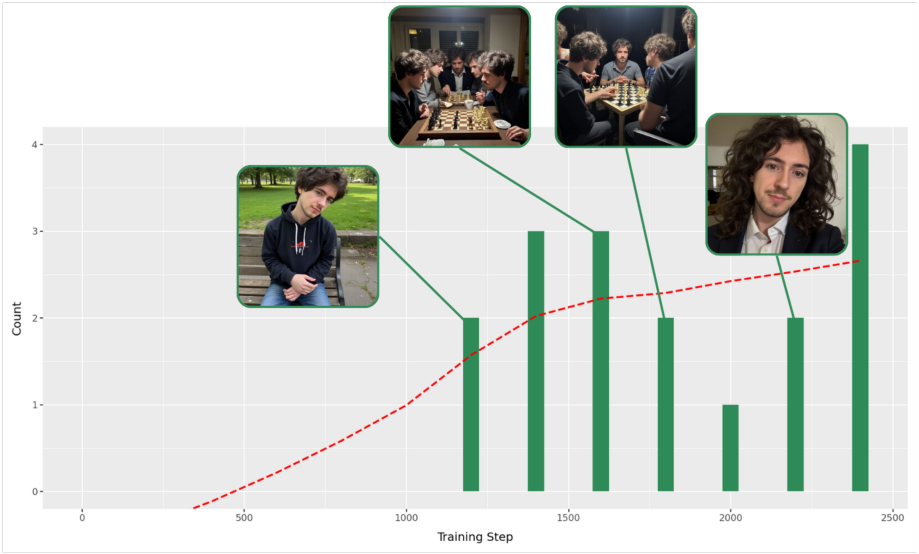

Figure 8: Shows the number of observed artifacts that distort the concept in some kind of increase of an attribute. This includes the number of identical concepts, head size and also hair length. Examples are shown too.

overemphasize the learned concept. In the case of the person concept, this lead to things such as bigger heads, uncalled for cloning and also longer hair.

Captioning Detail Matters The experiment suggests that a subtractive captioning approach—where the caption includes the trigger token plus explicit descriptions of what should not be associated with the concept—yields better results. While using only the trigger token lead to overfitting on incidental details and overly detailed captions (GPT-4o) confused the model and hindered representation learning, the intermediate Florence-2 captions allowed the model to separate core attributes from background elements. This indicates that by clearly specifying what is not part of the concept, the model can better learn the intended visual characteristics linked to the trigger.

Training Image Resolution Training with 1024×1024 images improved the detail quality learned by the Adapter, even though outputs are still 512×512. Both the CLIP score and an LLM-based prompt adherence score show better results for the higher resolution model on two out of three prompts. A custom pairwise comparison further confirms a clear preference for images generated with the larger training images, although the lower resolution model slightly outperforms on one implausible prompt. This aligns with recommendations, as the FLUX model itself was also trained with the larger input size. However, this of course increases compute load and training time as well.

Adapter Rank Influence Training with different LoRa matrix ranks reveals notable differences in output quality and stability. Both rank 4 and rank 16 models gradually improved in prompt adherence and limb quality, yet rank 4 showed more fluctuation in character consistency and struggled with open-ended, trigger-only prompts. For challenging cases like the chess in space prompt, both models initially aligned in concept and prompt adherence but later diverged, indicating potential overfitting on the concept in rank 4. The effect on the LLM metrics over the couse of training can be seen in figure 4. Additionally, frequency analysis, as shown in figure 5 suggests that higher-rank adapters (e.g., rank 32) yield outputs with sharper edges and clearer directional features, while rank 4 outputs tend to have more high-frequency noise, confirming that higher rank improves the preservation of fine details and overall structure.

Overall the LoRa finetuning approach represents an effective and lightweight method, that even works on smaller VRAM GPUs. The experiments in this report showed that a hyper parameter optimization is effective and important to find the best model checkpoints. The experiments also had quite a few limiations in terms of computational resources and time. The LLM metrics can likely - especially the image quality one - be improved by further prompt engineering and extensive testing.

References

[1] Hanqun Cao et al. *A Survey on Generative Diffusion Model*. 2023. arXiv: 2209.02646 [cs.AI]. URL: https://arxiv.org/abs/2209.02646.

[2] Jack Hessel et al. *CLIPScore: A Reference-free Evaluation Metric for Image Captioning*. 2022. arXiv: 2104.08718 [cs.CV]. URL: https://arxiv.org/abs/2104.08718.

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. *Denoising Diffusion Probabilistic Models*. 2020. arXiv: 2006.11239 [cs.LG]. URL: https://arxiv.org/abs/2006.11239.

[4] Edward J Hu et al. "LoRA: Low-Rank Adaptation of Large Language Models". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=nZeVKeeFYf9.

[5] Nataniel Ruiz et al. *DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation*. 2023. arXiv: 2208.12242 [cs.CV]. URL: https://arxiv.org/abs/2208.12242.

[6] Yang Song et al. "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=PxTIG12RRHS.

A Testing the LLM Metrics

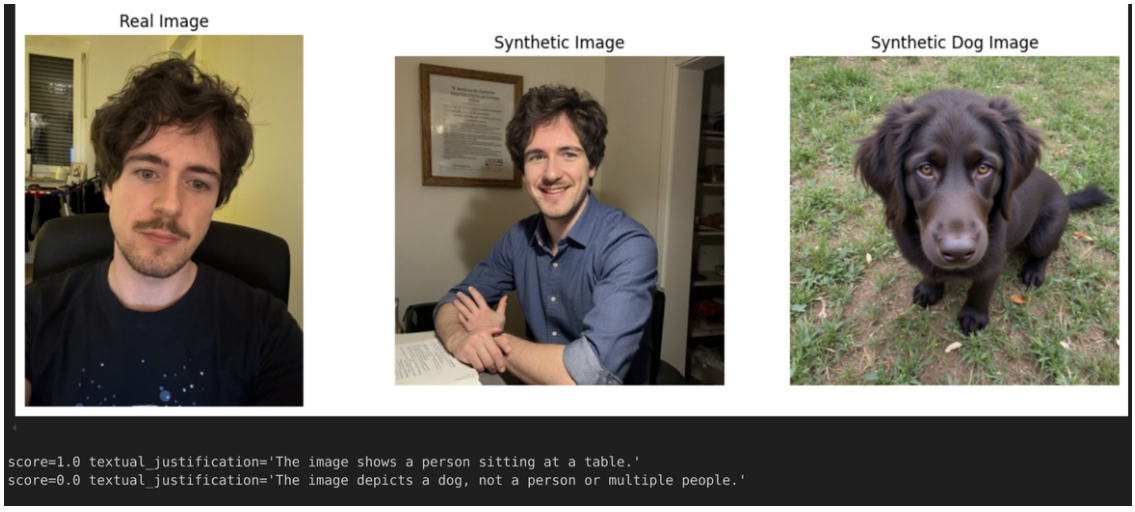

Figure 9: A representation of the validation of the binary LLM metric that judges whether the image contains a person (1) or not (0).

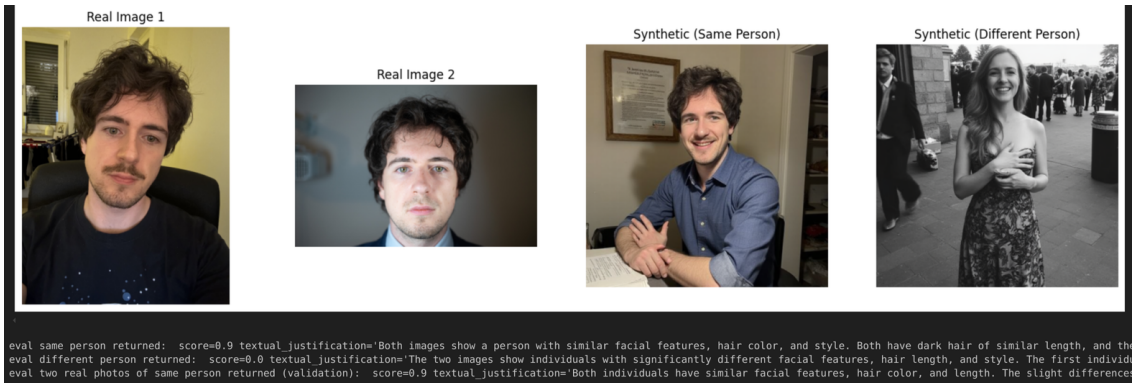

Figure 10: A representation of the validation of the LLM metric that estimates if a synthetic image contains the same person as the reference real image.

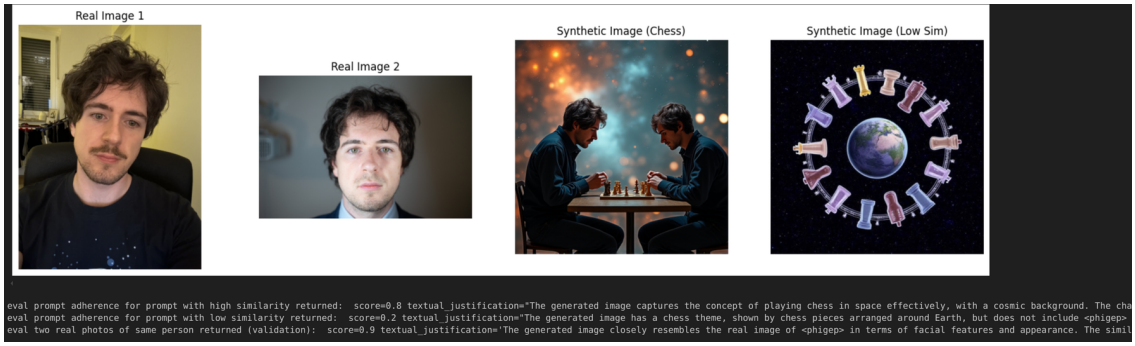

Figure 11: A representation of the validation of the LLM metric that tests for overall prompt adherence, both for the person concept and the additional prompt context.

B Training Script Example

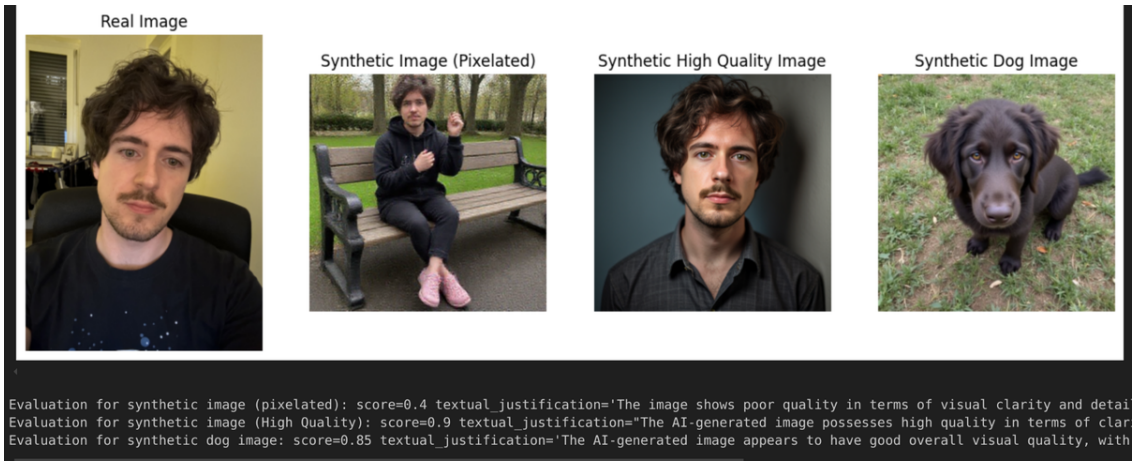

Figure 12: A representation of the validation of the LLM metric that tests for image quality with a direct LLM eval.

Listing 2: Command for Training with Accelerate, used by fluxgym in the background.

```
accelerate launch \
  --mixed_precision bf16 \
  --num_cpu_threads_per_process 1 \
  sd-scripts/flux_train_network.py \
  --pretrained_model_name_or_path "/home/phigep/pinokio/api/fluxgym.git/models/unet/flux1-dev.sft" \
  --clip_l "/home/phigep/pinokio/api/fluxgym.git/models/clip/clip_l.safetensors" \
  --t5xxl "/home/phigep/pinokio/api/fluxgym.git/models/clip/t5xxl_fp16.safetensors" \
  --ae "/home/phigep/pinokio/api/fluxgym.git/models/vae/ae.sft" \
  --cache_latents_to_disk \
  --save_model_as safetensors \
  --sdpa --persistent_data_loader_workers \
  --max_data_loader_n_workers 2 \
  --seed 42 \
  --gradient_checkpointing \
  --mixed_precision bf16 \
  --save_precision bf16 \
  --network_module networks.lora_flux \
  --network_dim 4 \
  --optimizer_type adamw8bit \
  --sample_prompts="/home/phigep/pinokio/api/fluxgym.git/outputs/personal-lora-fluxdev-florence2-1024/sample_prompts.txt" \
  --sample_every_n_steps="200" \
  --learning_rate 8e-4 \
  --cache_text_encoder_outputs \
  --cache_text_encoder_outputs_to_disk \
  --fp8_base \
  --highvram \
  --max_train_epochs 16 \
  --save_every_n_epochs 2 \
  --dataset_config "/home/phigep/pinokio/api/fluxgym.git/outputs/personal-lora-fluxdev-florence2-1024/dataset.toml" \
  --output_dir "/home/phigep/pinokio/api/fluxgym.git/outputs/personal-lora-fluxdev-florence2-1024" \
  --output_name personal-lora-fluxdev-florence2-1024 \
  --timestep_sampling shift \
  --discrete_flow_shift 3.1582 \
  --model_prediction_type raw \
  --guidance_scale 1 \
  --loss_type l2
```