LoRA

What is LoRA?

Textual Inversion was a technique to compress "appearances difficult to explain in text" into a single word, but it does not have the power to make the model draw something it doesn't know from scratch.

When you wanted to "make the model draw something it originally couldn't!", conventionally you needed to fine-tune the entire model. However, training costs are quite high.

So, **LoRA (Low-Rank Adaptation)**, which was originally used in LLMs, came to be used.

LoRA uses a method where instead of rewriting the weights of the model itself, only the "difference" is saved externally as small additional data. You can add new styles and characters to the base model as if loading an expansion pack later.

---

text2image Applying LoRA

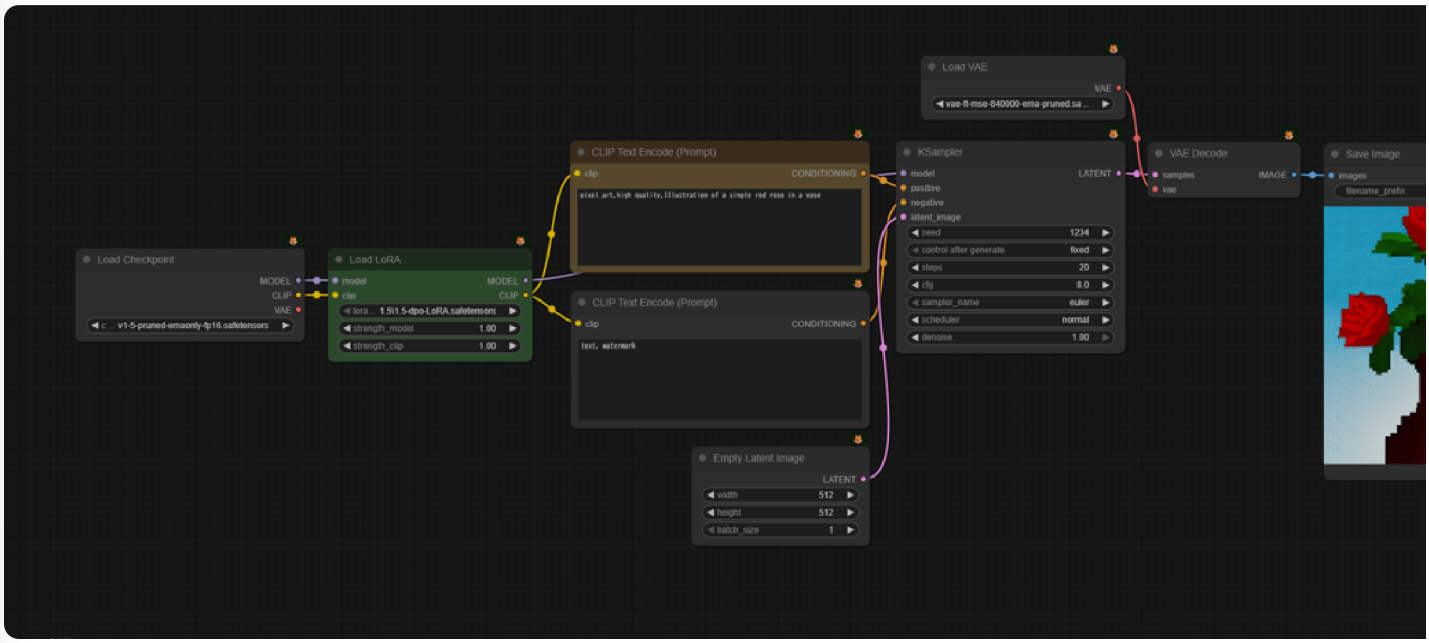
Downloading LoRA

This time, as an example, let's use a LoRA that makes it look like pixel art.

- 8bitdiffuser 64x

```
└── 📁 models/
    └── 📁 loras/
        └── PX64NOCAP_epoch_10.safetensors
```

workflow



SD1.5_lora.json

- 🟩 Add a `Load LoRA` node.
  - Connect it so that it is sandwiched between `Load Checkpoint` and `CLIP Text Encode` / `KSampler`.
  - Pass both **MODEL** and **CLIP** through `Load LoRA`.
- `strength_model` / `strength_clip` : The application strength of LoRA. Basically `1.0`, but lower it if it works too strongly.
- 🟧 Trigger Word
  - Just applying LoRA adds the ability to draw pixel art to the base model internally.
  - However, to ensure that ability is brought out, you need to include the word the author used during training in the prompt.
  - This is called a trigger word. In this LoRA, `pixel_art` is the trigger word.

Change in Design Philosophy of Image Generation AI

In **Stable Diffusion 1.5** and **SDXL**, when applying LoRA, it was common to target both the diffusion model (the core of image generation) and the text encoder (which interprets prompts) for learning.

However, in models after **Flux.1**, large language models like T5 and Qwen have been adopted for the text encoder. These are like small ChatGPTs and already have general-purpose language understanding capabilities, so re-training them for image generation is inefficient, and may even degrade performance.

Therefore, in the latest models, the design where the text encoder is fixed and only the diffusion model itself is trained has become mainstream.

LoRA Follows Suit

LoRA also follows this.

Up to SDXL, both the diffusion model and the text encoder were trained, but in models after Flux.1, LoRA training and application are also for the diffusion model only.

Changes in ComfyUI Workflow

You could use the `Load LoRA` node, but it's not very elegant to connect a node to a CLIP that isn't being used. So, the `LoraLoaderModelOnly` node is provided instead. As the name suggests, it is a node that applies LoRA only to the MODEL (diffusion model).

Flux.1_lora.json

In new models, LoRA is applied like this. Please remember this.
