

Part 8

Stable Diffusion 模型

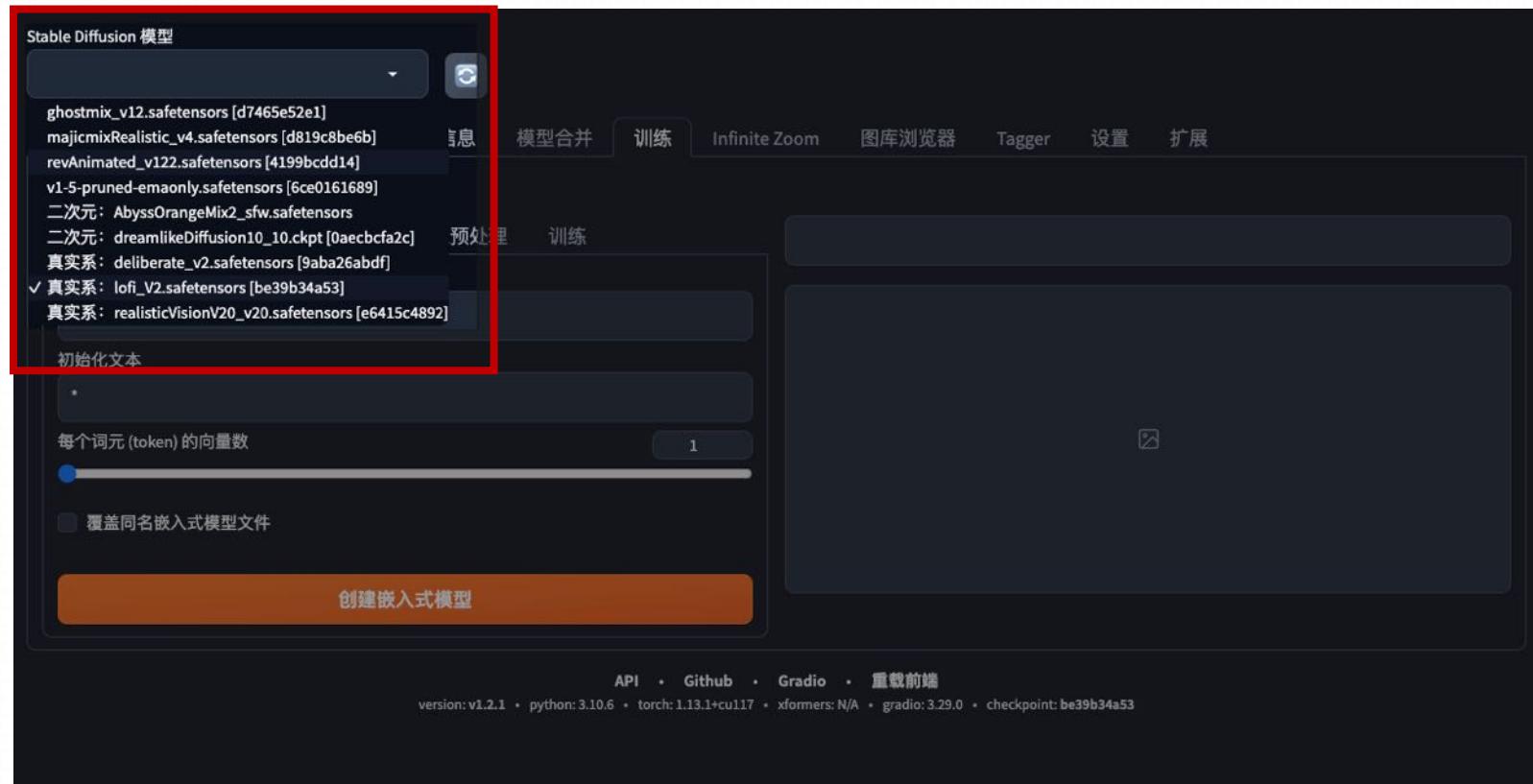


Stable Diffusion

1. 模型概念
2. CheckPoint模型
3. Embedding模型
4. LoRA模型
5. Hypernetwork模型
6. VAE模型
7. 训练自己的模型

1. 模型概念

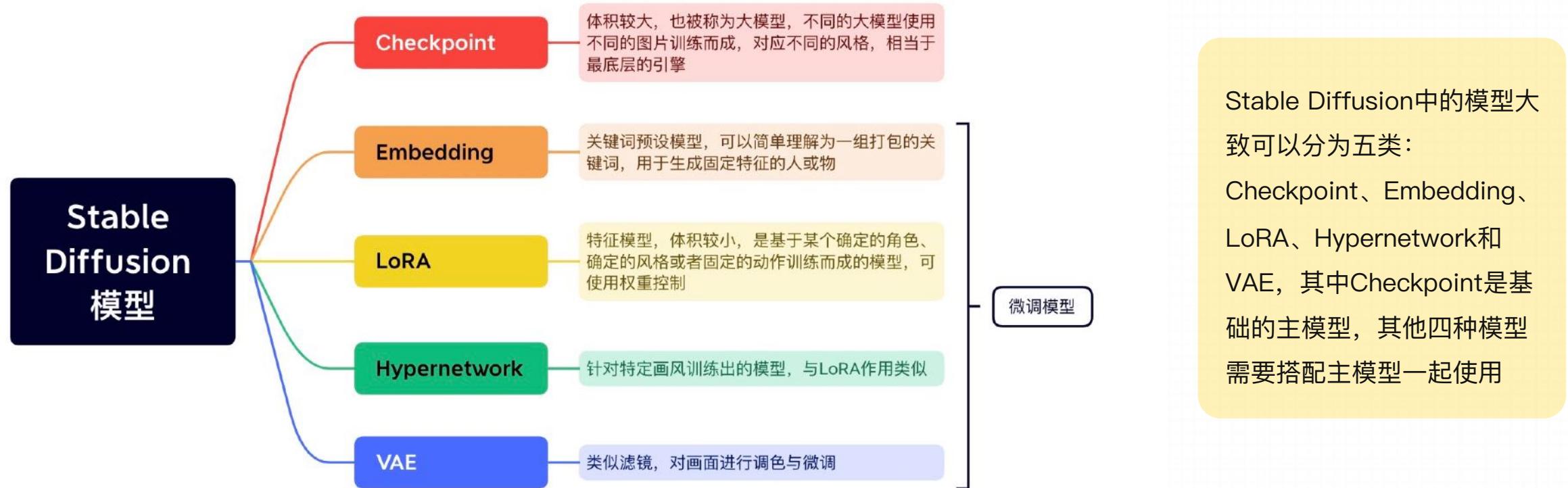
什么是Stable Diffusion中的模型



模型是AI生图的重要依据，算图用的模型可以安装很多个，WebUI在初次启动时，会自动下载原始版的Stable Diffusion v1.5模型。由于不同人对于AI生图有不同的需求，以写实风格为主的Stable Diffusion模型可能无法满足所有人。因此我们可以下载自己想要使用的模型来控制生成效果。

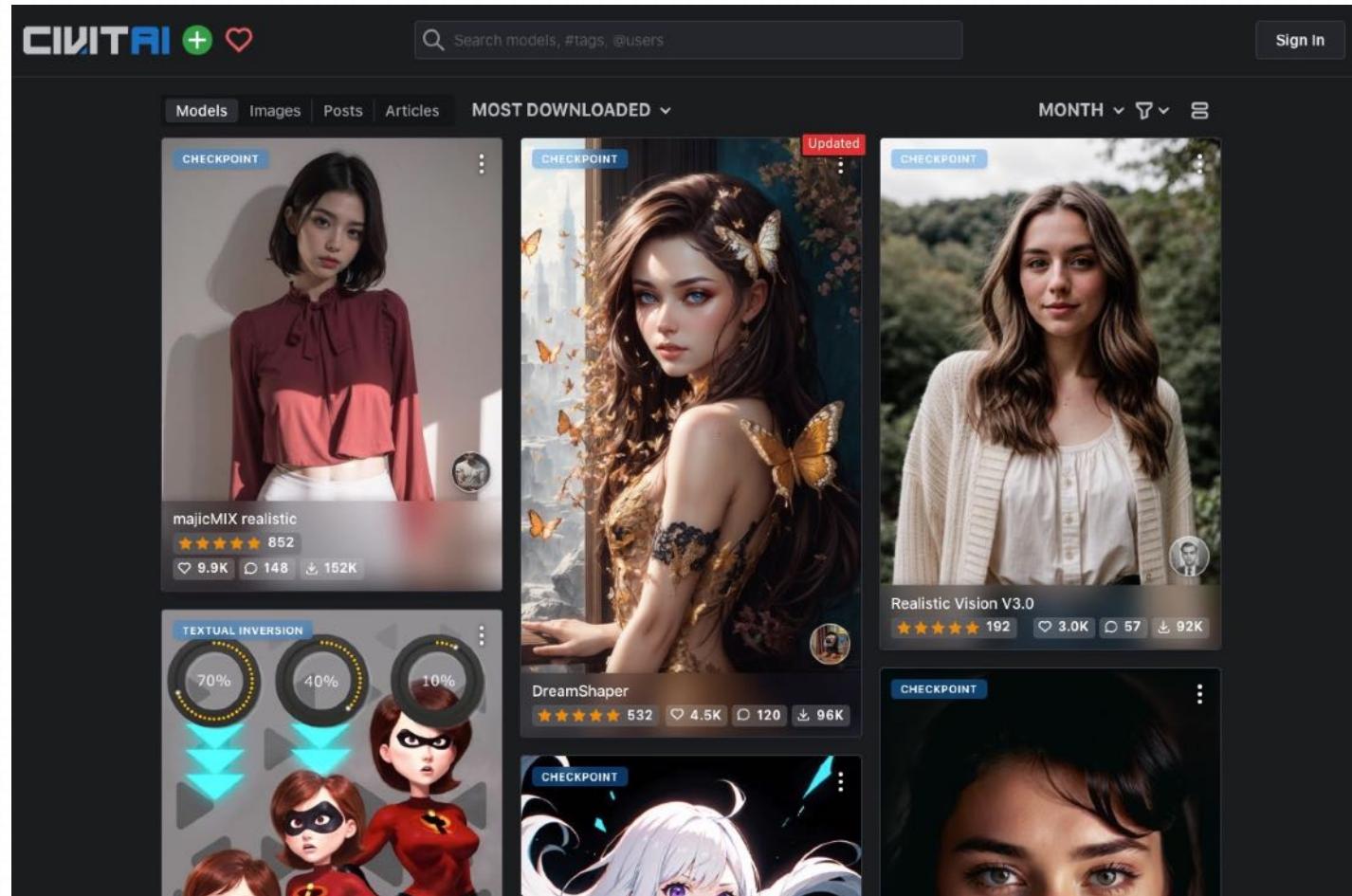
1. 模型概念

模型分类



1. 模型概念

模型下载--Civitai



Civitai

C站，一个AI绘画模型分享平台，特色是模型都有示范缩略图，用户也可以分享彼此使用的提示词，以及分享作品。

链接: <https://civitai.com/>

1. 模型概念

模型下载—Hugging Face

Hugging Face Search models, datasets, users...

Models 4,978 Filter by name new Full-text search Sort: Trending

- Tasks: Libraries, Datasets, Languages, Licenses, Other
- Multimodal: Feature Extraction, Text-to-Image, Image-to-Text, Text-to-Video, Visual Question Answering, Document Question Answering, Graph Machine Learning
- Computer Vision: Depth Estimation, Image Classification, Object Detection, Image Segmentation, Image-to-Image, Unconditional Image Generation, Video Classification, Zero-Shot Image Classification
- Natural Language Processing: Text Classification, Token Classification, Table Question Answering, Question Answering, Zero-Shot Classification, Translation, Summarization, Conversational, Text Generation, Text2Text Generation

Model	Type	Last Updated	Size	Stars
runwayml/stable-diffusion-v1-5	Text-to-Image	May 5	3.44M	8.5k
stabilityai/stable-diffusion-2-1	Text-to-Image	May 16	218k	2.79k
CompVis/stable-diffusion-v1-4	Text-to-Image	May 16	336k	5.7k
prompthero/openjourney	Text-to-Image	May 16	293k	2.74k
WarriorMama777/OrangeMixs	Text-to-Image	5 days ago	9.29k	3.17k
stabilityai/sd-vae-ft-mse-original	Text-to-Image	Dec 19, 2022		974
gsdf/Counterfeit-V3.0	Text-to-Image	Apr 29		309
SG161222/Realistic_Vision_V1.4	Text-to-Image	May 2	257k	248
dreamlike-art/dreamlike-photoreal-2.0	Text-to-Image	Mar 13	190k	1.4k

Hugging Face

一个允许用户共享AI学习模型和数据集的平台，可以说是人工智能界的Github，Stable Diffusion背后用到的很多AI工具，如Transformer、Tokenizers、Datasets都可以在平台上找到，网站上也有丰富的教学文档

链接：

https://huggingface.co/models?pipeline_tag=text-to-image&sort=trending

1. 模型概念

模型安装

路径：

Checkpoint模型：

\models\Stable-Diffusion

Embedding模型：

\embeddings

LoRA模型：

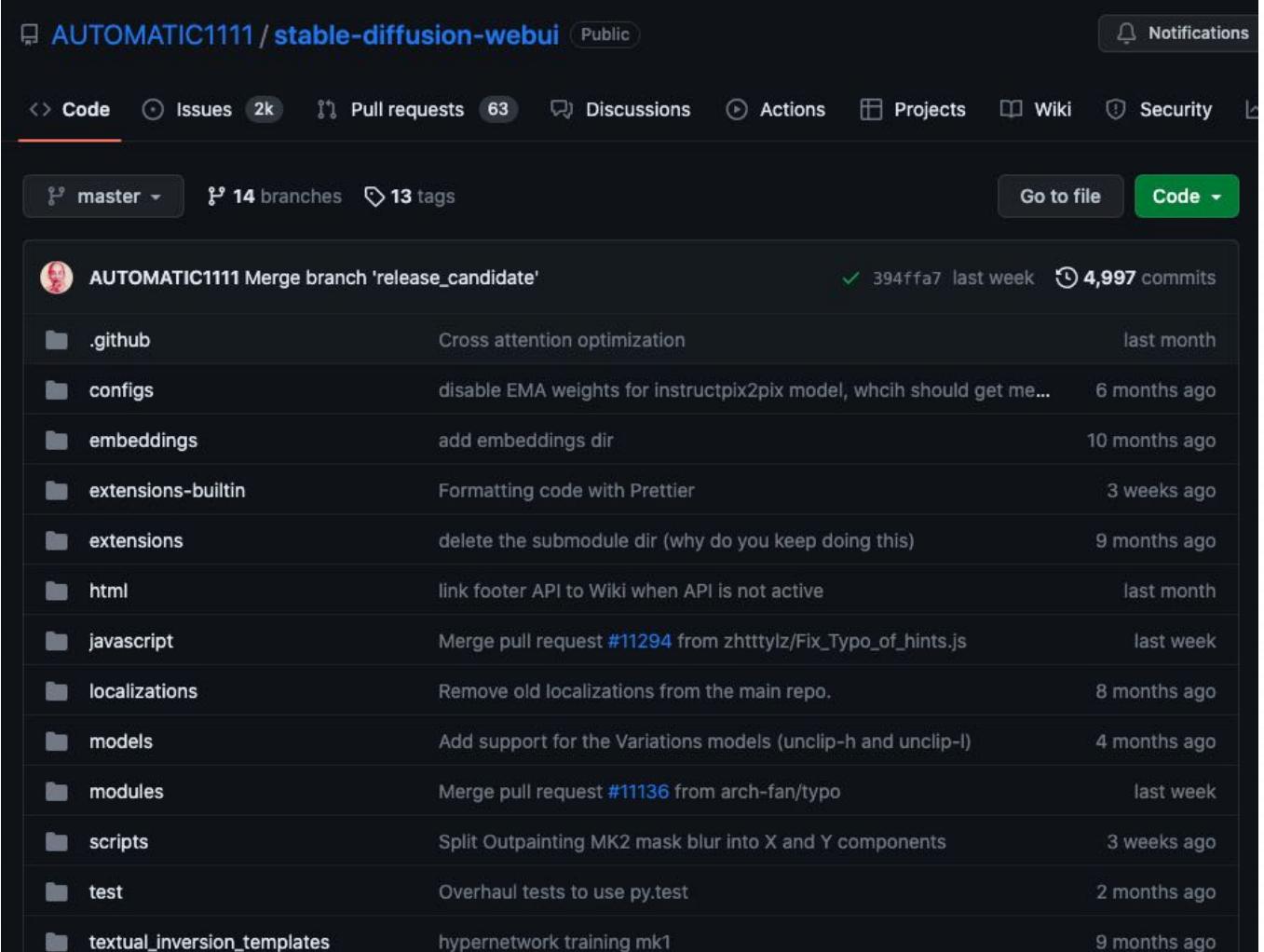
\models\lora

Hypernetwork模型：

\models\hypernetworks

VAE模型：

\models\VAE



The screenshot shows the GitHub repository page for `AUTOMATIC1111/stable-diffusion-webui`. The repository is public and has 14 branches and 13 tags. The master branch is selected. The commit history lists 4,997 commits from the last 9 months, with the most recent being a merge pull request from `zhttylz`. The commits are categorized by file changes, such as .github, configs, embeddings, extensions-builtin, extensions, html, javascript, localizations, models, modules, scripts, test, and textual_inversion_templates. Each commit includes a brief description and the date it was last updated.

Commit Details	Description	Last Updated
.github	Cross attention optimization	last month
configs	disable EMA weights for instructpix2pix model, whcih should get me...	6 months ago
embeddings	add embeddings dir	10 months ago
extensions-builtin	Formatting code with Prettier	3 weeks ago
extensions	delete the submodule dir (why do you keep doing this)	9 months ago
html	link footer API to Wiki when API is not active	last month
javascript	Merge pull request #11294 from zhttylz/Fix_Typo_of_hints.js	last week
localizations	Remove old localizations from the main repo.	8 months ago
models	Add support for the Variations models (unclip-h and unclip-l)	4 months ago
modules	Merge pull request #11136 from arch-fan/typo	last week
scripts	Split Outpainting MK2 mask blur into X and Y components	3 weeks ago
test	Overhaul tests to use py.test	2 months ago
textual_inversion_templates	hypernetwork training mk1	9 months ago

2. Checkpoint模型

模型概念

Checkpoint模型，是Stable Diffusion能够进行绘图的基础模型，被称为大模型、主模型、检查点模型等等

Checkpoint模型是通过Dreambooth训练方式得到的大模型，这些文件包含了模型参数和优化器状态等信息，是训练过程中定期保存的状态快照。一个大模型训练起来是很消耗算力的，checkpoint类似于在游戏中的存档，模型运算到一个关键点时会保存一个版本，便于回滚和继续计算

不同的checkpoint模型使用不同的图片训练而成，对应不同的风格，相当于最底层的引擎。对于使用者而言，可以将Checkpoint 文件理解为不同风格的画手，如油画、漫画、写实风等。通过选择对应的 Checkpoint 文件，可以将 Stable Diffusion 模型生成的结果转换为所选择的特定风格

路径：\models\Stable-Diffusion

Checkpoint模型比较大，通常在2–7GB，格式为**ckpt**和**safetensors**格式，还有ema、full ema、nonema、pruned等版本

2. Checkpoint模型

模型概念

相同的Prompt，不同checkpoint的效果：

Prompt: half body portrait of a young woman, black hair



Realistic Vision



lofi



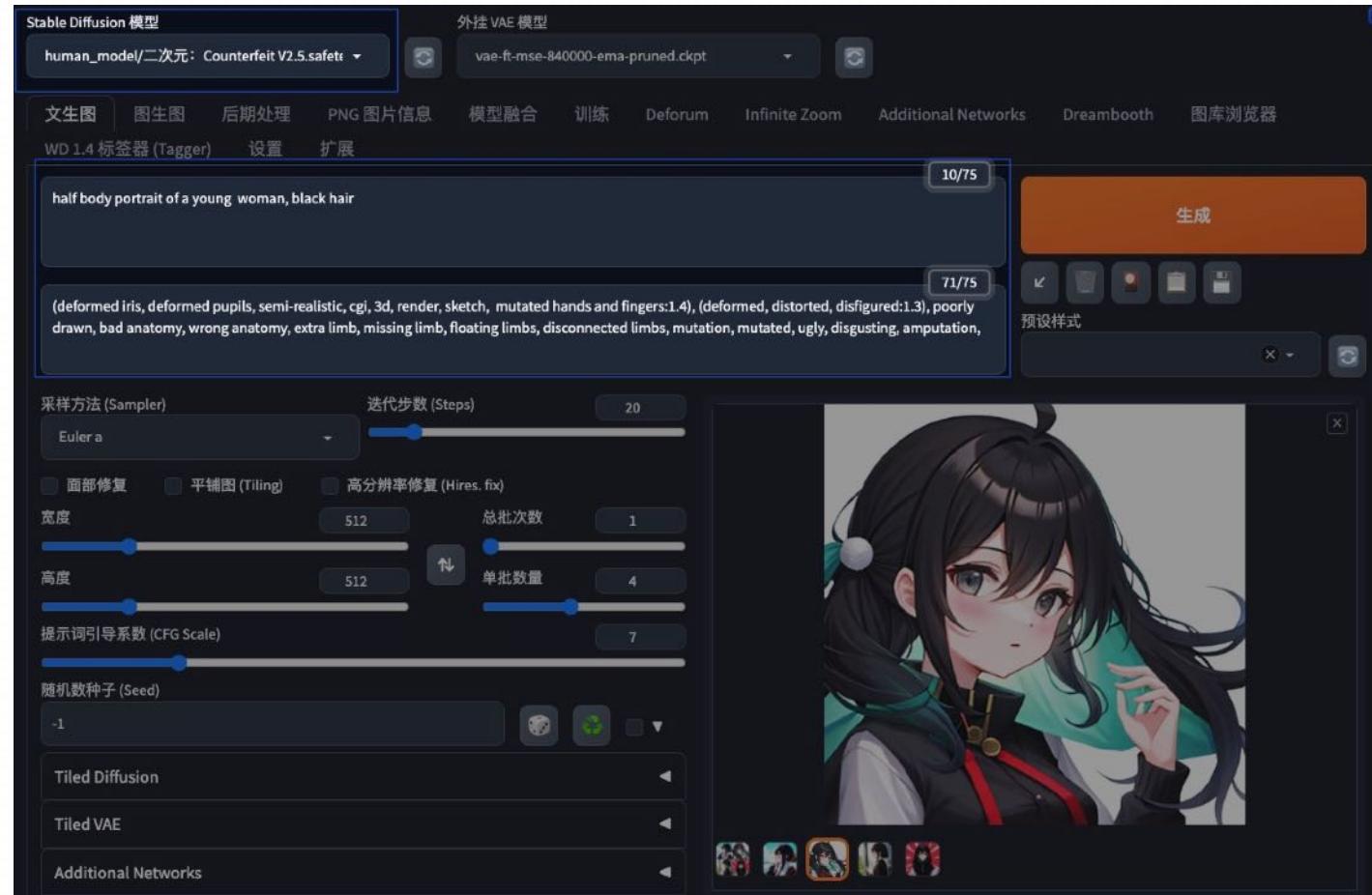
Anything



Counterfeit

2. Checkpoint模型

模型使用方法



在使用相应的checkpoint模型时，可参考模型作者给出的文档，c站完整的模型下面一般会给出文档，包括

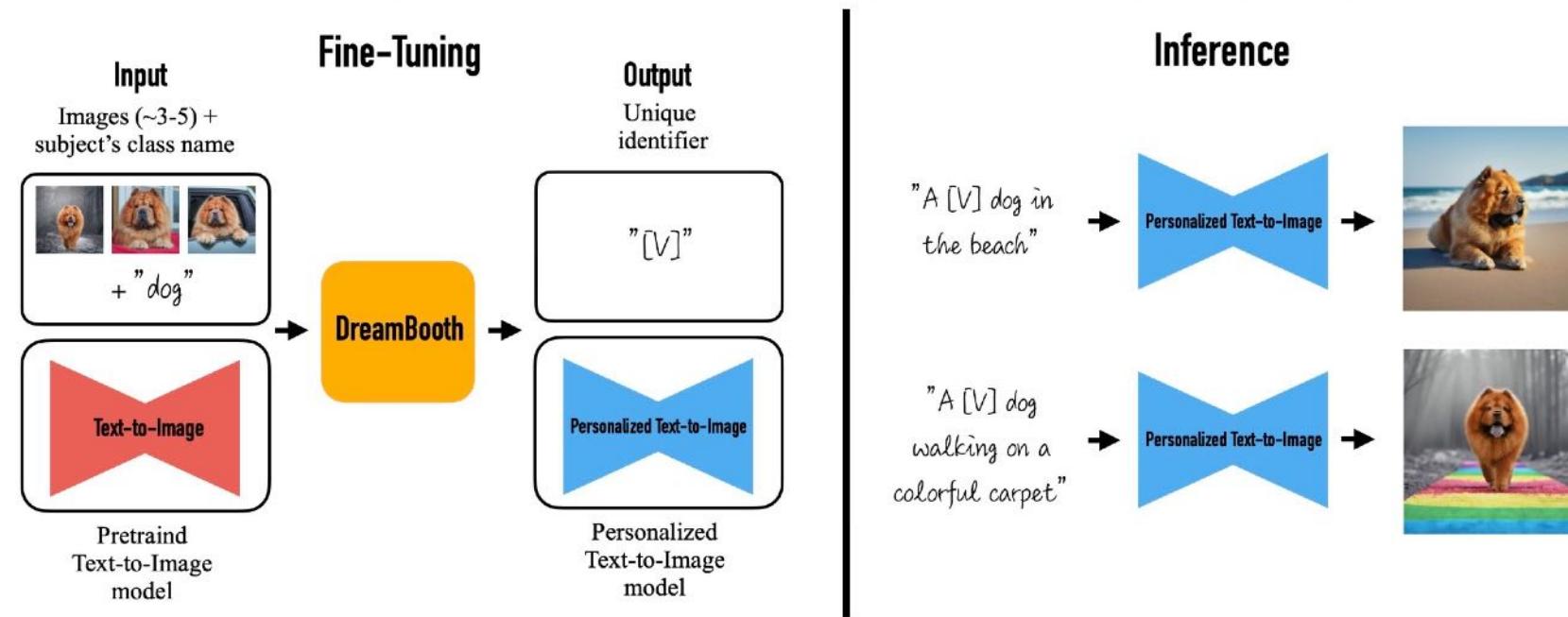
- 模型擅长的风格
- VAE的使用建议
- 正面提示词的结构建议
- 负面提示词的结构建议

2. Checkpoint模型

模型训练--Dreambooth

DreamBooth是谷歌推出的一个主题驱动的AI生成模型，它可以微调文本到图像扩散模型或新图像的结果。

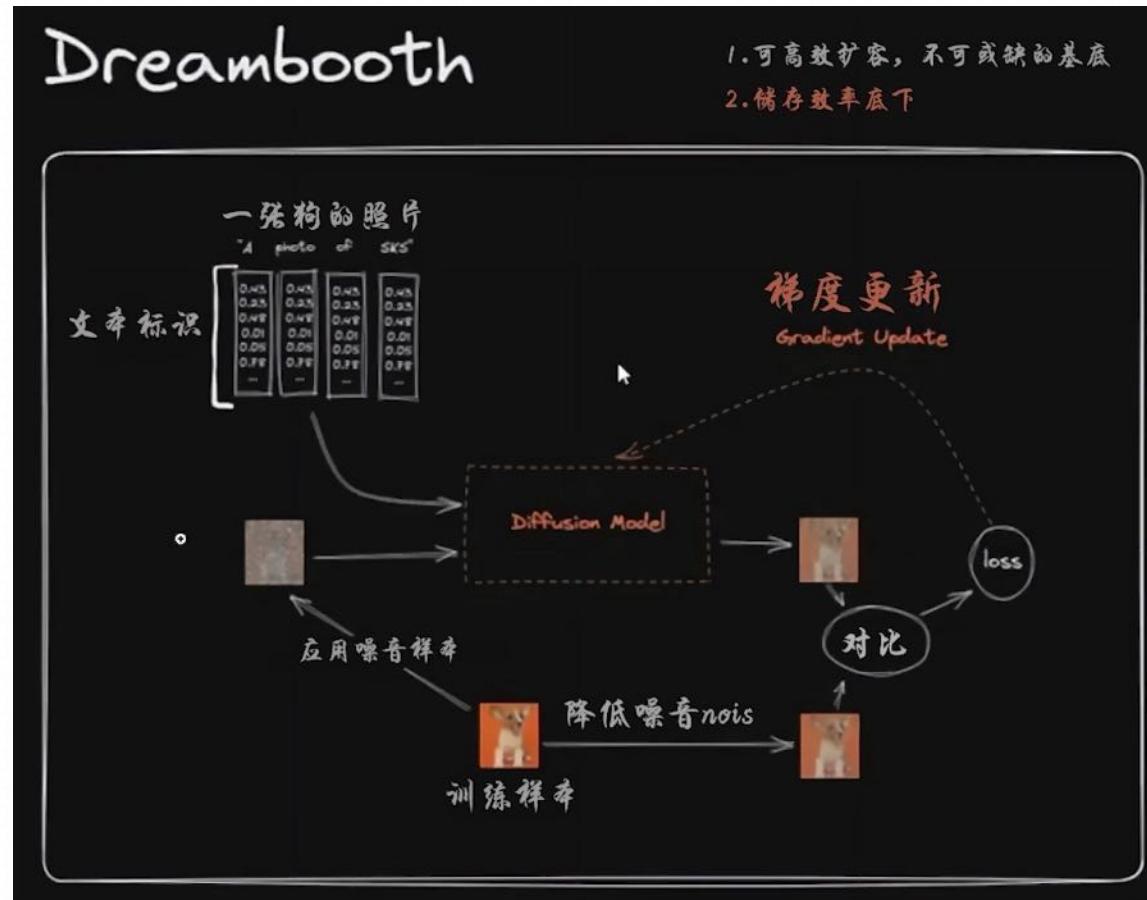
Checkpoint模型使用Dreambooth方法进行训练，通过直接修改整个原始模型来对模型输出结果进行调校，微调整个网络参数



如果硬件条件许可的话，搜集大量图片训练特定领域的checkpoint大模型最好（对比stable Diffusion 1.5版的模型输入了23亿张图片，网络上其他人训练至少几万张图片）

2. Checkpoint模型

模型训练--Dreambooth

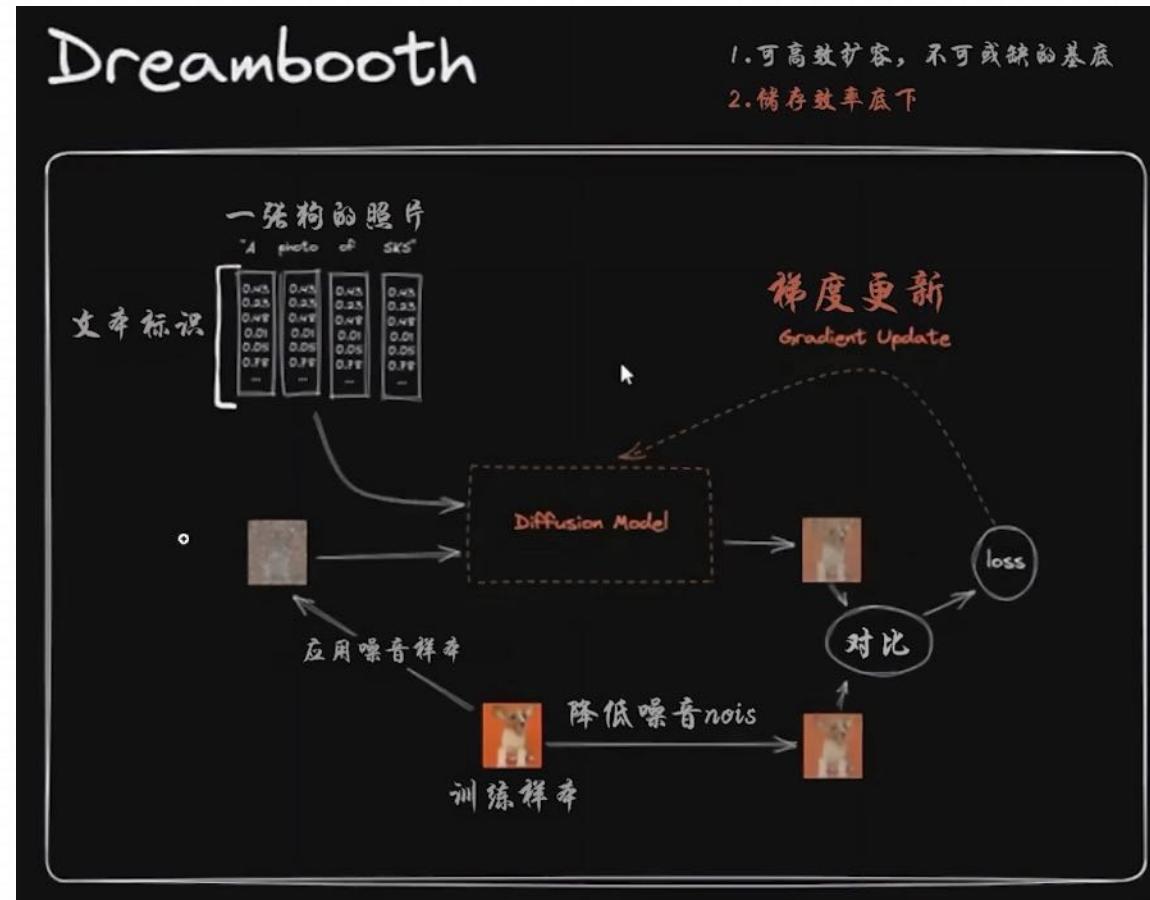


以左图为例，左图中Dreambooth所训练的事情，就是将柯基犬的图片与关键词SKS建立起关联关系

1. 给训练图片添加 n 步噪声，使其变成较为嘈杂的图片，另外再给训练图片添加较少一点的噪声($n-1$)，使其成为一张校准图片
 2. 然后我们来训练SD模型以左侧较嘈杂的图片作为输入，再加上特殊关键词指令的输入，能输出右侧较为清晰的图片

2. Checkpoint模型

模型训练--Dreambooth

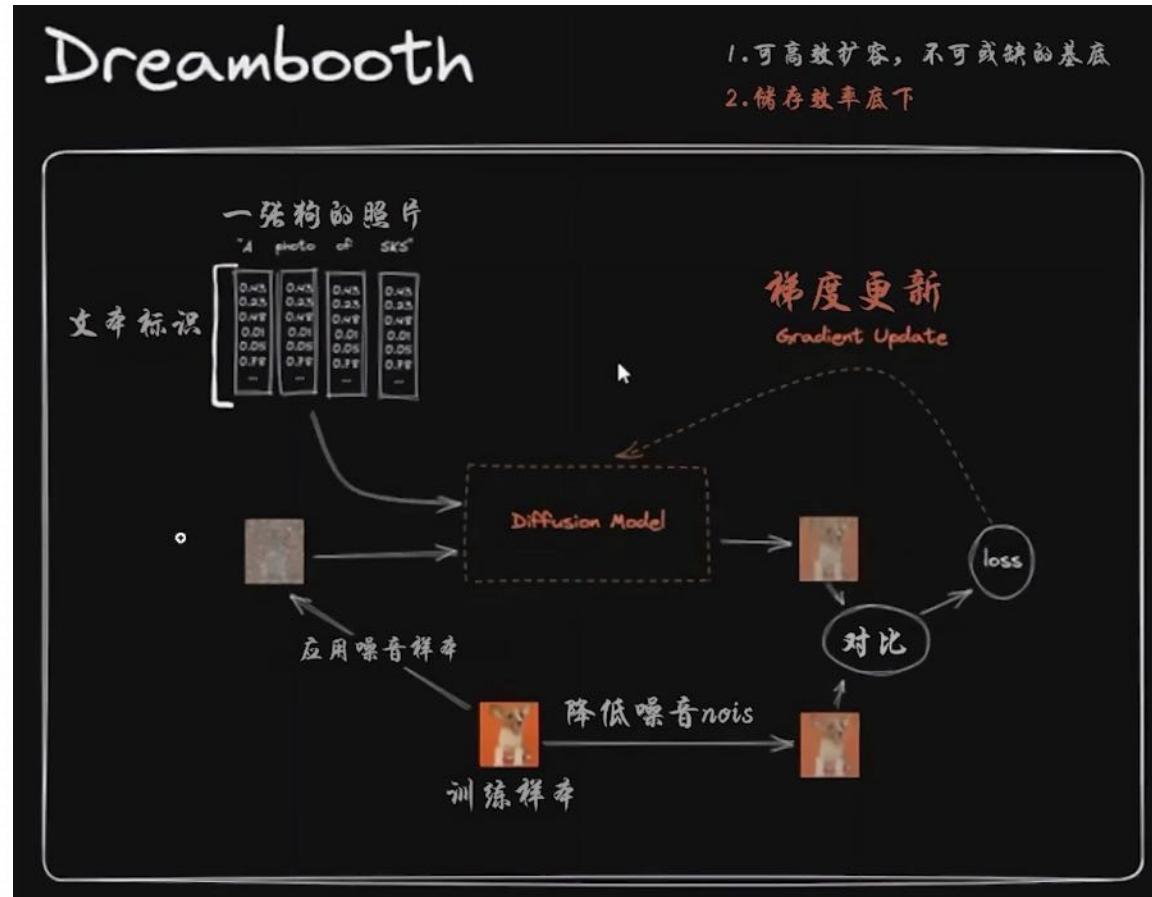


以左图为例，左图中Dreambooth所训练的事情，就是将柯基犬的图片与关键词SKS建立起关联关系

- 一开始，由于模型不识别新增的特殊关键词SKS，他可能输出了一个不是很好的结果。此时我们将该结果与目标图片（右侧较少噪声的图片）进行比较，得出一个loss结果，用以描述生成图像与目标图像的差异程度
- 接着Dreambooth会进行Gradient Update。可以简单理解为，如果Loss高的话它将惩罚模型，如果Loss低的话它将奖励模型

2. Checkpoint模型

模型训练--Dreambooth



以左图为例，左图中Dreambooth所训练的事情，就是将柯基犬的图片与关键词SKS建立起关联关系

5. 当训练重复了一段时间后，整个模型会逐渐认识到：当它收到SKS的词语输入时，生成的结果应该看起来比较像训练者所提供的柯基犬的图片，由此我们便完成了对模型的调校

2. Checkpoint模型

模型推荐



majicMIX系列模型

真人大模型，基于KanPiroMix + XSMix + ChikMix三个模型融合而来，包括：

- majicMIX realistic：主打真实系，是通用版本
- majicMIX sombre：和realistic差不多，但是带了一点点阴暗的色调。内置了VAE所以不需要单独用
- majicMIX fantasy：风格非常美好，但是远距离脸部需要inpaint以达成最好
- majicMIX lux:融合了realistic和fantasy
- majicMIX horror：偏恐怖主题

C站链接：<https://civitai.com/models/43331/majicmix-realistic>

2. Checkpoint模型

模型推荐



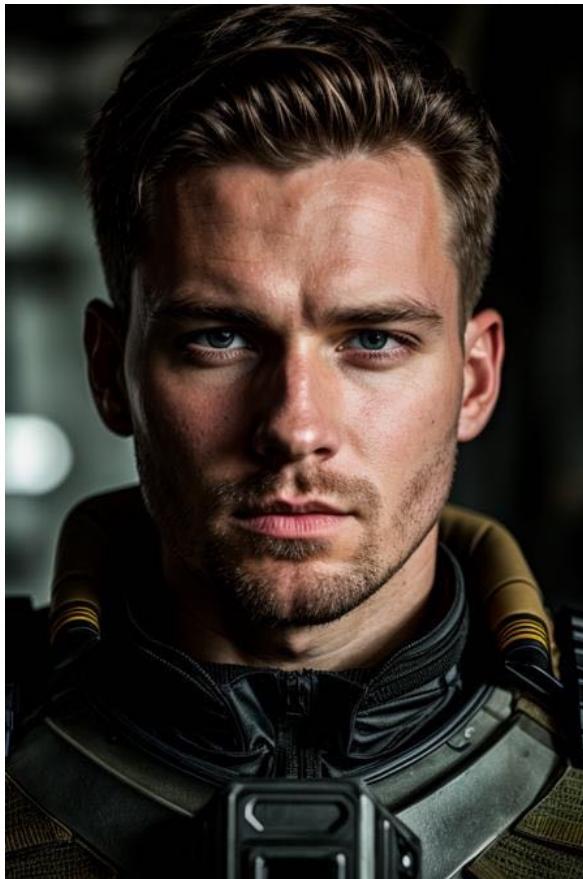
Deliberate

真实系模型，不管是人物、动物还是生活用品，还原的都非常到位，属于全面万能的真实系模型

C站链接：<https://civitai.com/models/4823/deliberate>

2. Checkpoint模型

模型推荐



Realistic Vision

真实系模型，生成人像的效果与照片类似，对毛发的还原达到了像素级别

C站链接: <https://civitai.com/models/4201/realistic-vision-v30>

2. Checkpoint模型

模型推荐



MeinaMix

二次元模型，擅长描绘二次元和动漫风格的图像

C站链接：

<https://civitai.com/models/7240/meinamix>

2. Checkpoint模型

模型推荐



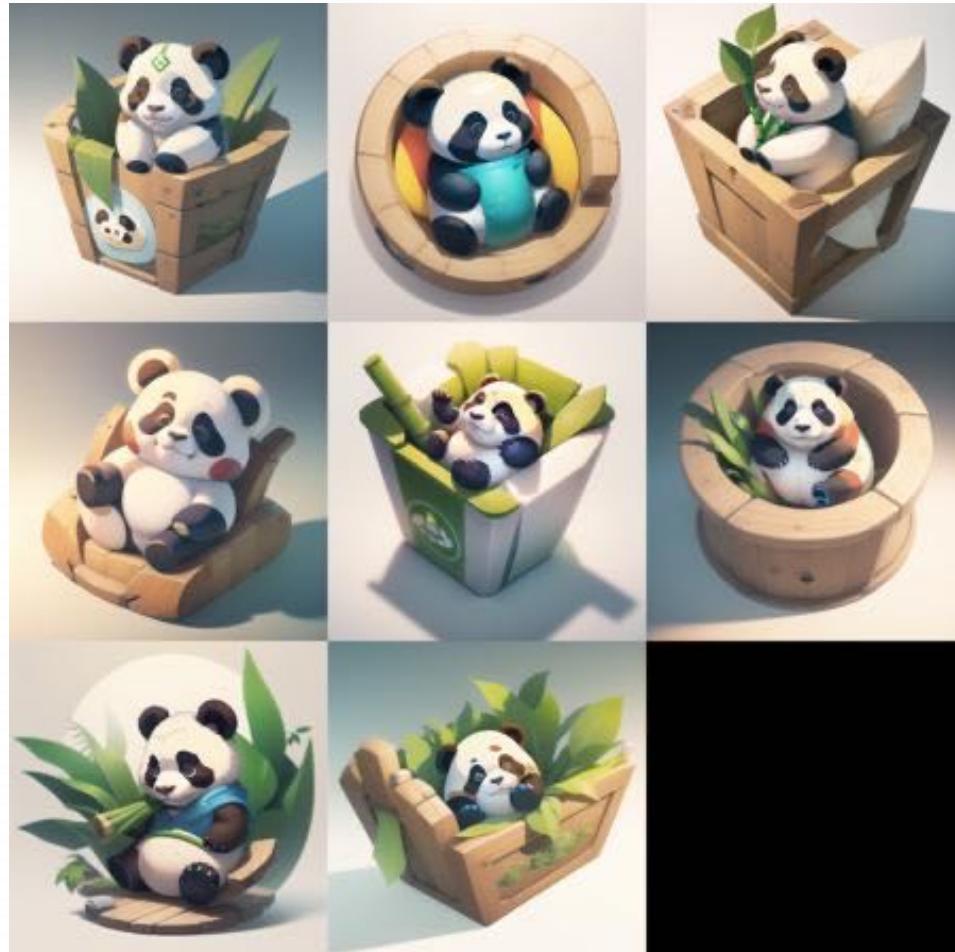
万象熔炉 | Anything

二次元模型，适用性很广，作者的更新速度
快，同时也非常适合搭配各种动漫人物的lora
模型使用

C站链接：<https://civitai.com/models/9409/or-anything-v5ink>

2. Checkpoint模型

模型推荐



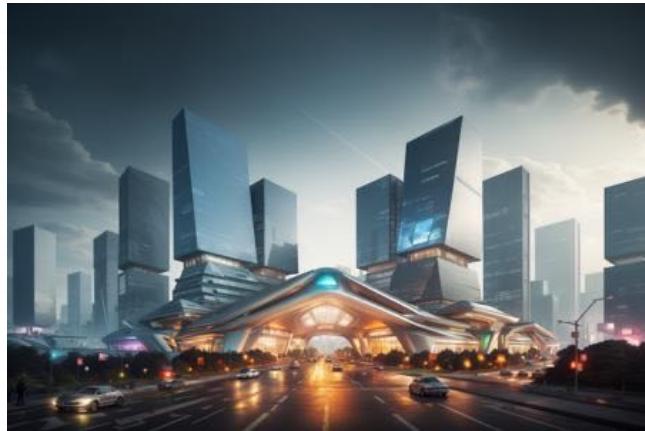
Game Icon Institute_mode

游戏Icon模型，2.0版本适合创建2D样式的图标，3.0版本适合创建3D样式的图标。

C站链接：<https://civitai.com/models/47800/game-icon-institutemode>

2. Checkpoint模型

模型推荐



ArchitectureRealMix

写实风格建筑大模型ArchitectureRealMix，适用于绝大部分建筑设计、景观设计、城市设计、室内设计场景

C站链接：<https://civitai.com/models/84958/architecturrealmix>

3. Embedding模型

模型概念

Embedding 模型，使用Textual inversion方法进行训练，通常用于控制人物的动作和特征，输出固定特征的人或物。

Embedding 技术将输入数据转换为向量表示，为模型的处理和生成提供了便利。通过使用 embedding，我们可以更加轻松地生成符合预期的样本，而不需要手动输入大量的描述词汇。

举个例子，如果我们想要生成一个开心的皮卡丘，通常需要输入很多描述词，如黄毛、老鼠、长耳朵、腮红等等。但是，如果引入皮卡丘的 embedding，我们只需要输入两个词：皮卡丘和开心。皮卡丘的 embedding 打包了所有皮卡丘的特征描述，这样我们就不用每次输入很多单词来控制生成的画面了。

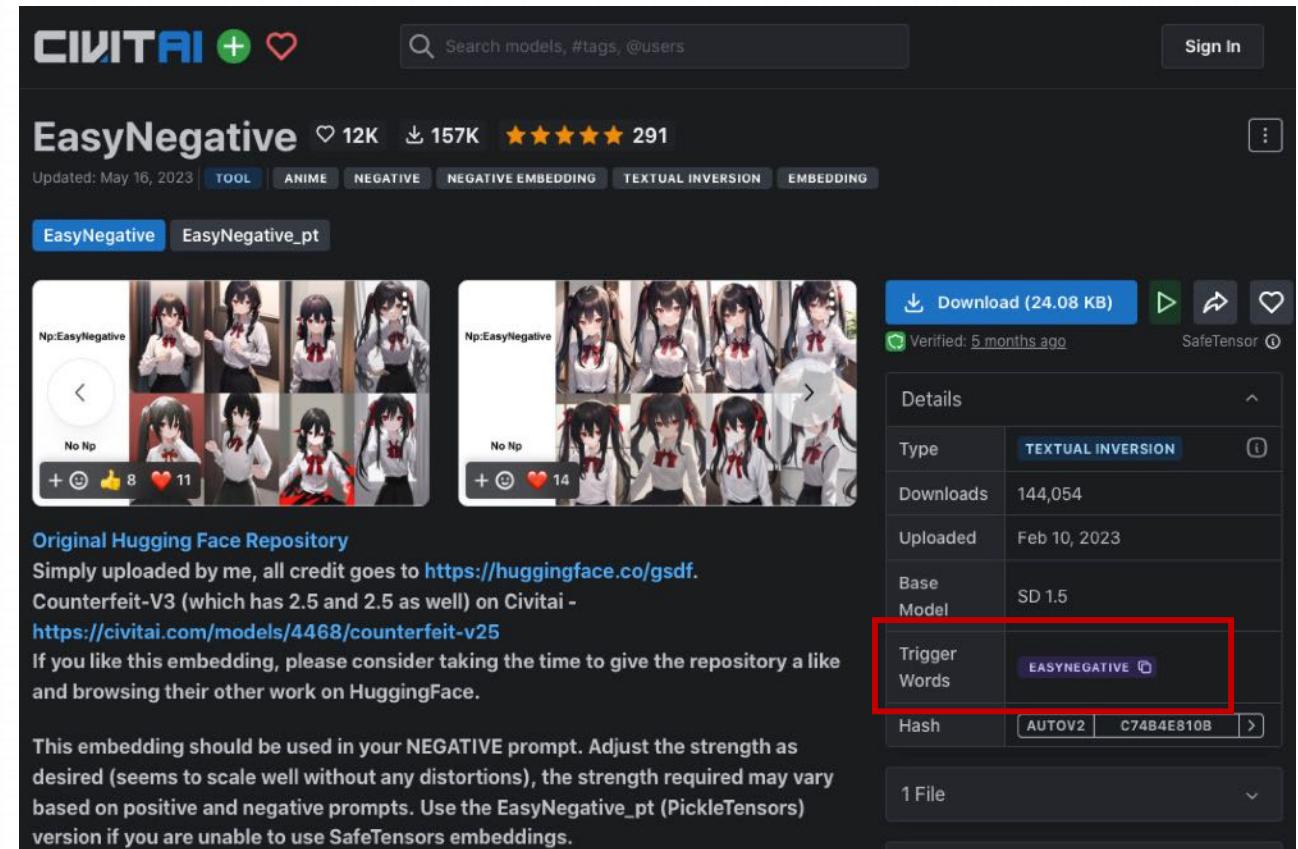
路径: \embeddings

Embedding模型通常为10~100 KB，常见格式为**pt**

3. Embedding模型

模型使用方法

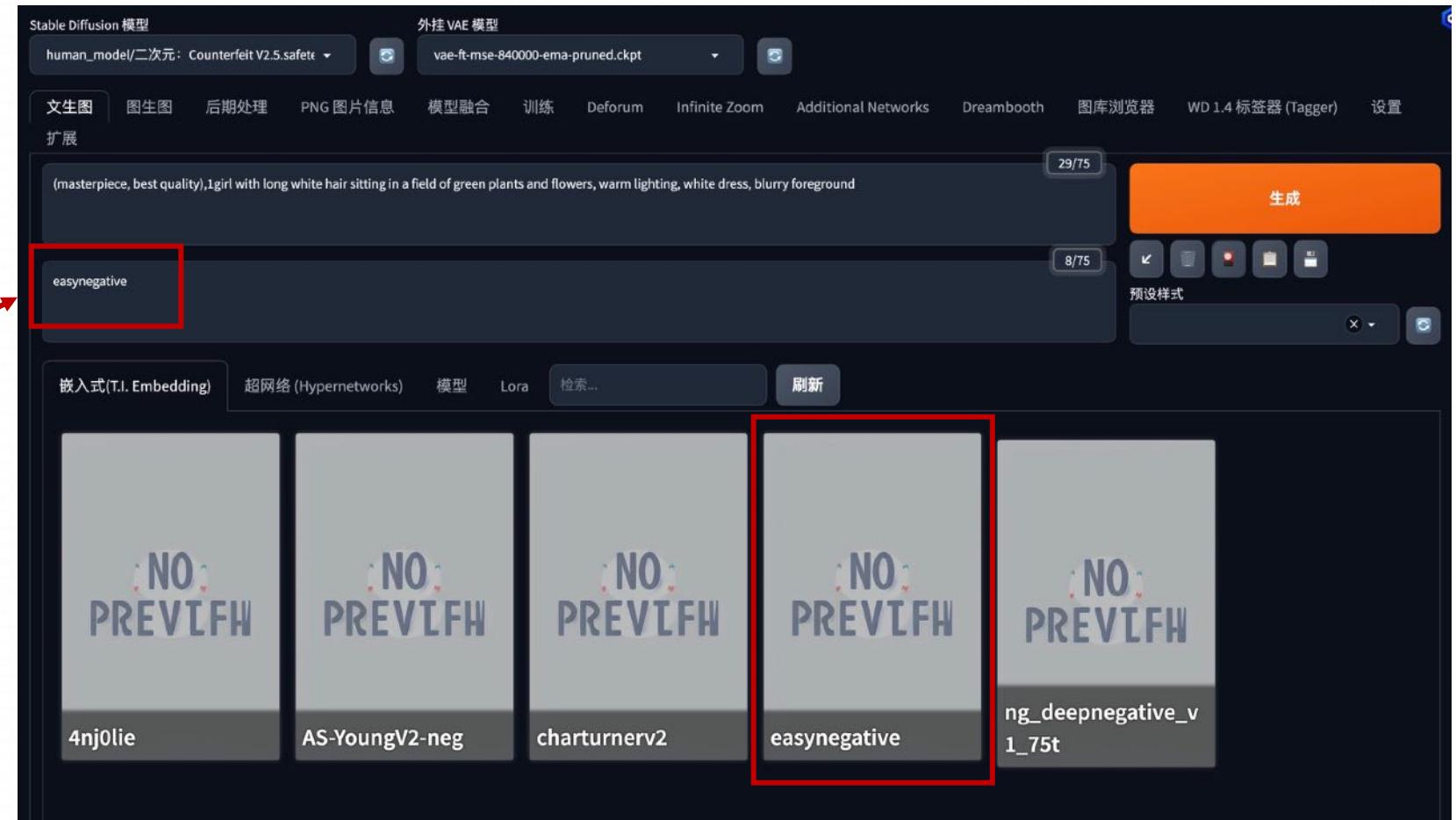
将embedding文件下载，拷贝至根目录下的embedding目录里。在下载安装之后，使用提示词来激活，只要在正面或者负面提示词中提到它的TriggerWords就相当于调用该Embedding模型



3. Embedding模型

模型使用方法

EasyNegative模型
Trigger Words: easynegative



3. Embedding模型

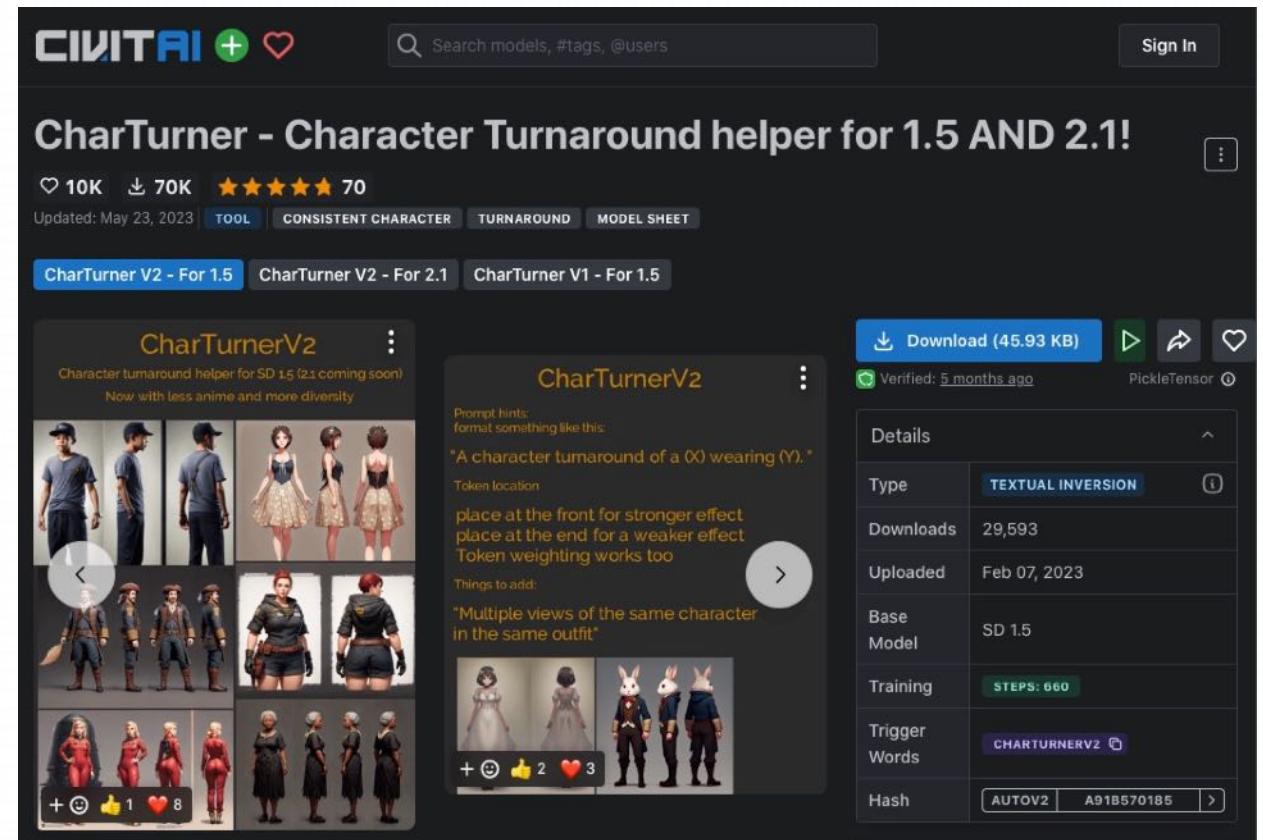
模型使用方法

Embedding模型需要和checkpoint模型搭配使用：

主模型：相当于画师，有着其擅长的画画技巧

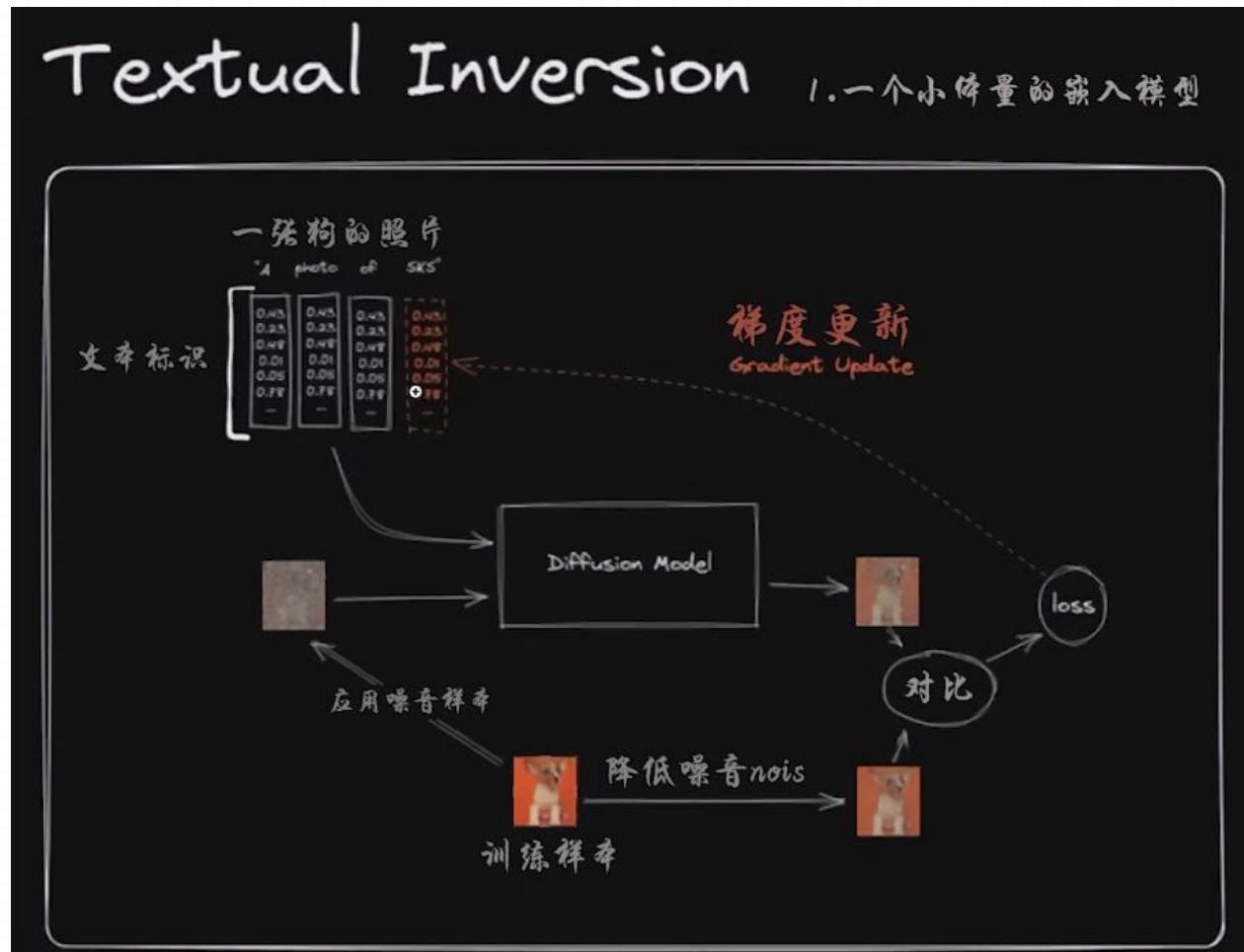
Embedding模型：相当于画师对文字要求的理解是否到位

如果没有主模型是无法画出内容的，有时候主模型选择不恰当，加上embedding模型之后效果也并不好。需求的图片和画师擅长的技巧不一样，理解的再恰当也画不出，例如需要画二次元的图片，而使用写实的主模型，效果并不好



3. Embedding模型

模型训练--Textual inversion

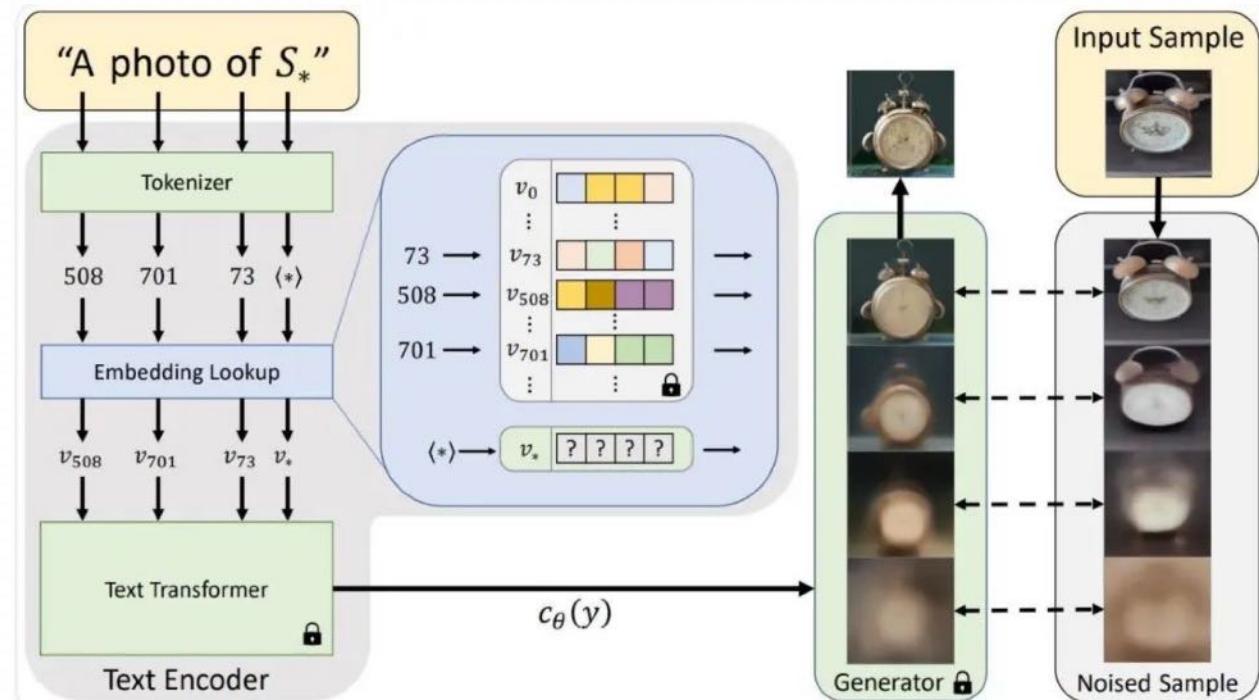


和主模型训练是同一套逻辑，但是最后梯度更新不会进入到模型里，而是进入text embedding文本标识这一步，改变模型如何理解输入的语言，在主模型之上附加的一个校正理解的小模型

不会改变模型，它只是定义新的关键字来实现某些样式

3. Embedding模型

模型训练--Textual inversion

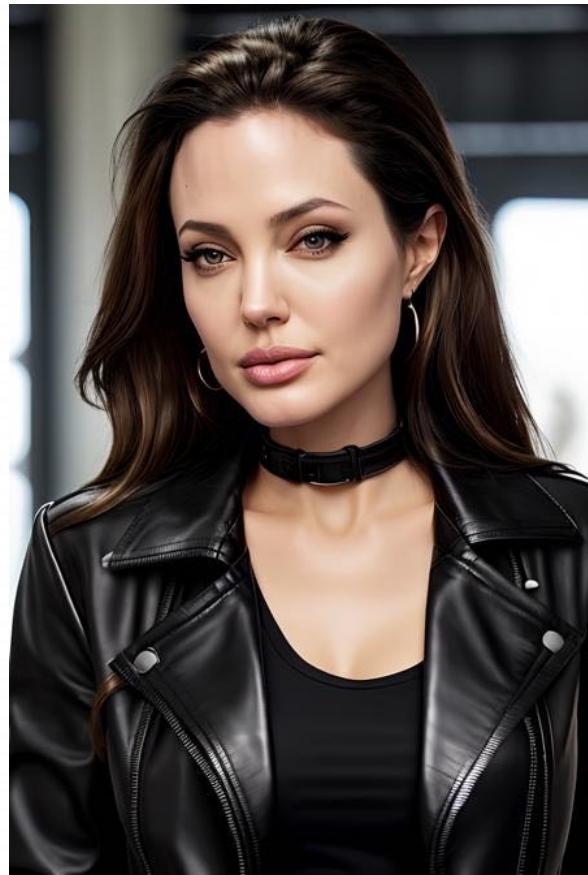


参数向量简单来说就是prompt经过语言模型处理后所得到的一个向量数据，后续的Diffusion Model将使用这些由输入prompt “A photo of S*”转换而来的一组向量数据作为实际的输入参数，来影响并指导Diffusion生成图像的过程

通过训练，Textual Inversion为prompt中的关键词找到了一个最合适的向量

3. Embedding模型

模型推荐



Angelina Jolie (JG):

针对特定人物安吉丽娜·朱莉的Embedding模型

C站链接: <https://civitai.com/models/99201/angelina-jolie-jg>

3. Embedding模型

模型推荐

Np:EasyNegative



No Np

EasyNegative:

是一种综合的，全方位的基于负面样本的提炼。里面包含了大量的负面词，可以减少每次打一堆负面词的工作

eg 将其放入negative prompt中可以在一定程度上解决AI不会画手、肢体错乱、颜色混杂等问题

针对Counterfeit训练的，对大多数二次元模型均有效

3. Embedding模型

模型推荐

NG_DeepNegative_V1_75T



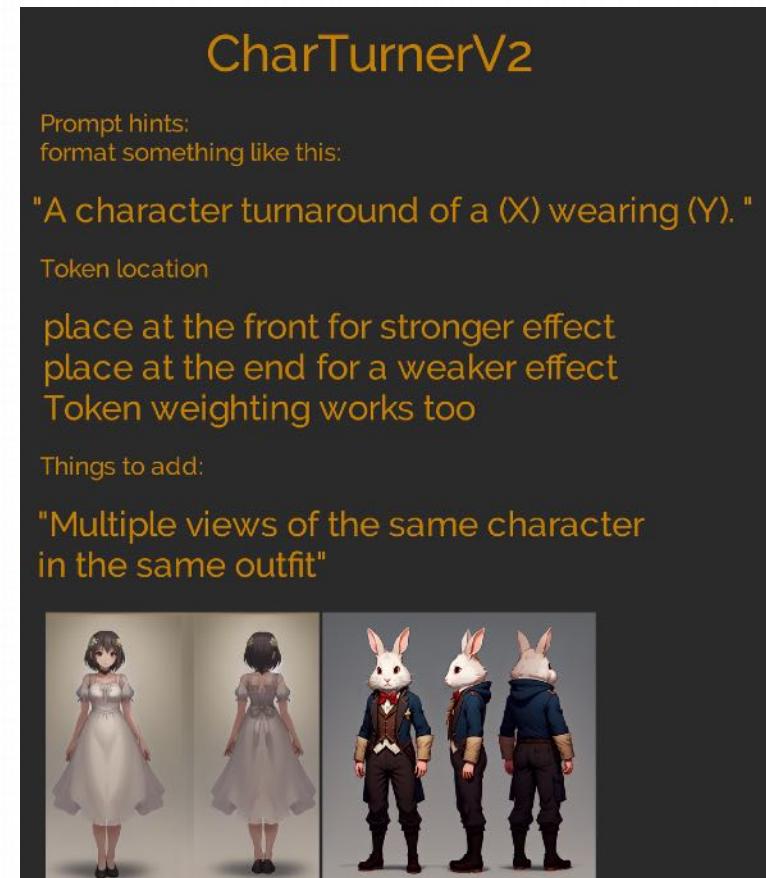
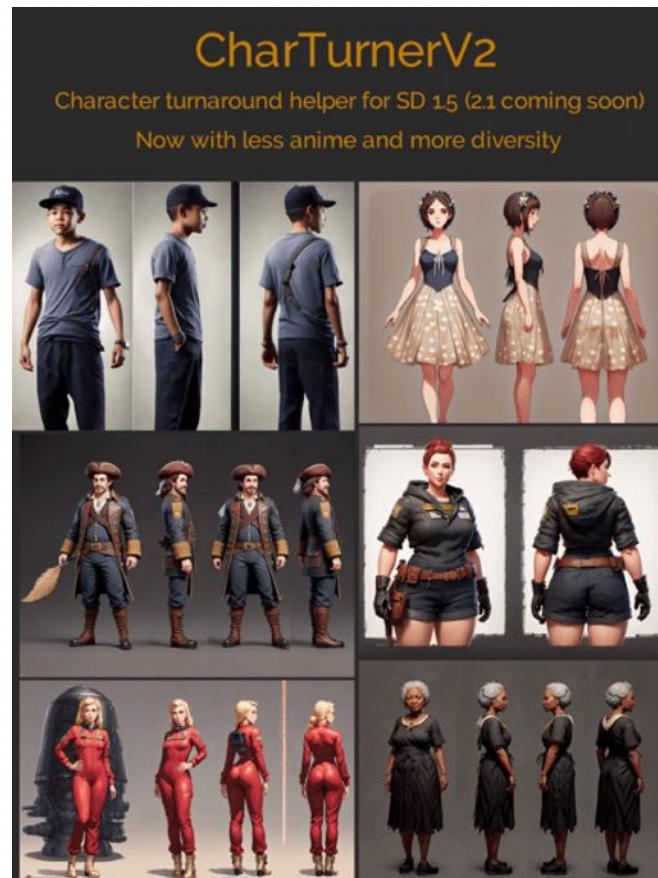
Deep Negative:

与EasyNegative相似，也是一种基于负面样本的提炼，不同之处是针对真人模型训练的

C站链接: <https://civitai.com/models/4629/deep-negative-v1x>

3. Embedding模型

模型推荐



CharTurner 角色转身Embeddings：

可以生成人物形象三视图设计

Prompt:

A Character turnaround of a (X) wearing (Y)

C站链接: <https://civitai.com/models/3036/charturner-character-turnaround-helper-for-15-and-21>

3. Embedding模型

模型推荐



Age Slider:

一些stable diffusion模型很难产生年龄较小的人。这个
Embedding模型将为您解决这个问题，它可以让任何人，在
任何LoRA，任何模型中，呈现出年轻的特征

C站链接: <https://civitai.com/models/65214/age-slider>

4. LoRA模型

模型概念

LoRA 模型，全称是Low-Rank Adaptation Model 大模型的“低秩适应”，最早在2021年由微软的研究团队提出并应用在其早期的大语言模型中，在引入LoRA之前，stable diffusion只能通过checkpoint模型来实现作画，如果对大模型产出的效果不满意，就只能再训练，对大模型不断的迭代、微调，而大模型的训练要求高，消耗的算力多，速度慢，而LoRA作为微调扩散模型的一种方法，效果好并且训练较为快速和简单，降低了模型训练的门槛并拓宽了模型的适用范围

LoRA可以用于对特定人物形象的描摹，对视觉艺术风格的实现，或者将某种特定元素如服饰、姿势融入画面等等，LoRA模型需要搭配Checkpoint模型一起使用，如果Checkpoint是一本大字典，LoRA就像其中的一张小夹页，字典的内容不会被夹页改变，但夹页可以在字典之外向作图的AI提供更多信息。与Embedding相比，LoRA模型的确定性更强，Embedding 和 LoRA 有功能交集的部分，也有互相不可取代的地方。

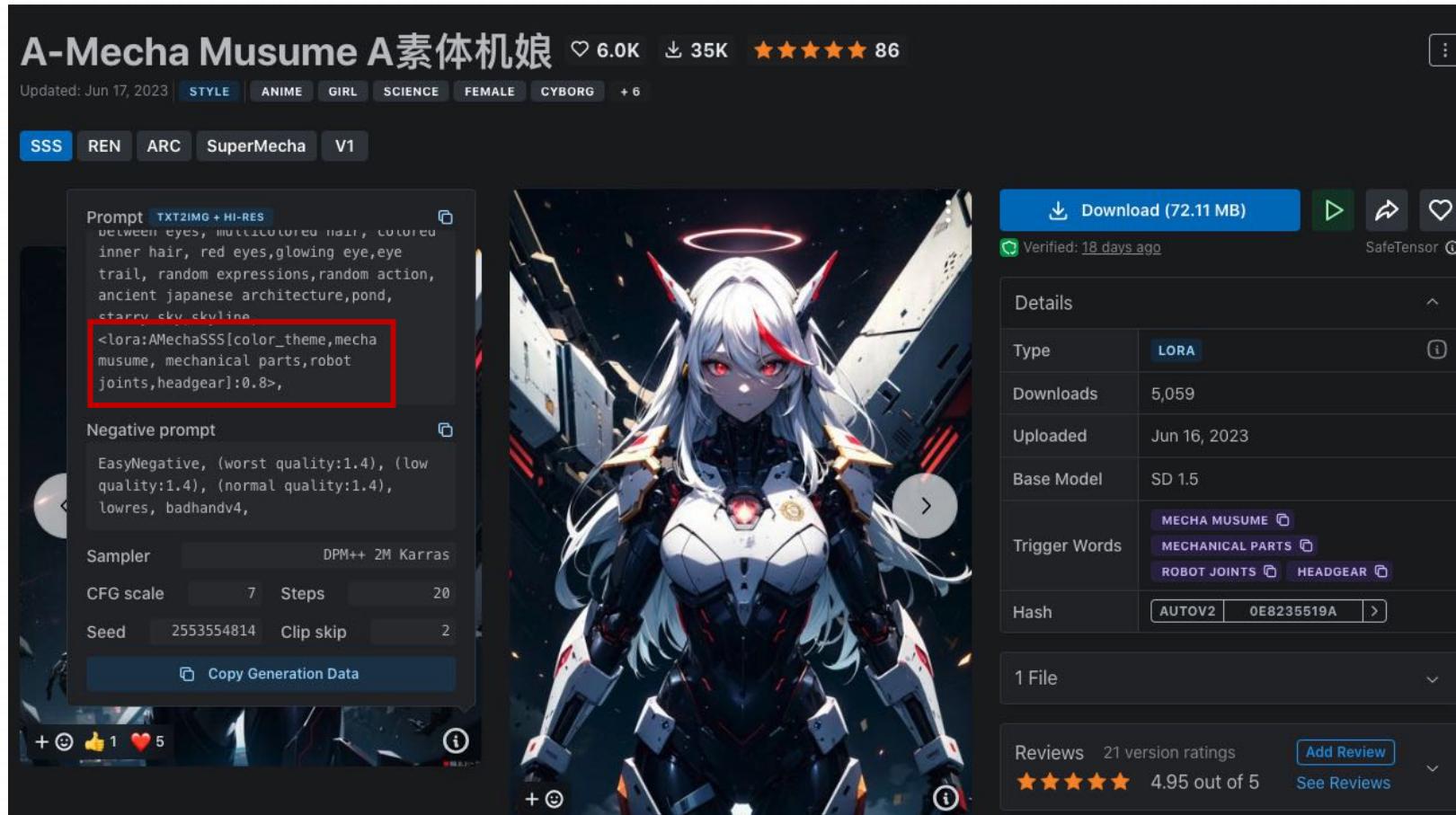
路径：\models\lora

LoRA模型通常为10~200 MB，常见格式为**ckpt (safetensors)**

4. LoRA模型

模型权重

在Stable Diffusion中，LoRA模型可以叠加使用，例如一个实现人物形象的描绘，另一个负责整体画风的把控



权重：LoRA的使用的过程中一般会对画风构成一些影响，注意使用LoRA时的权重（训练图源复杂）
0.5–0.8之间可以确切的保留特征，同时减弱对画面风格的影响
一般LoRA作者也会给出他们认为合适的LoRA权重

4. LoRA模型

模型使用方法

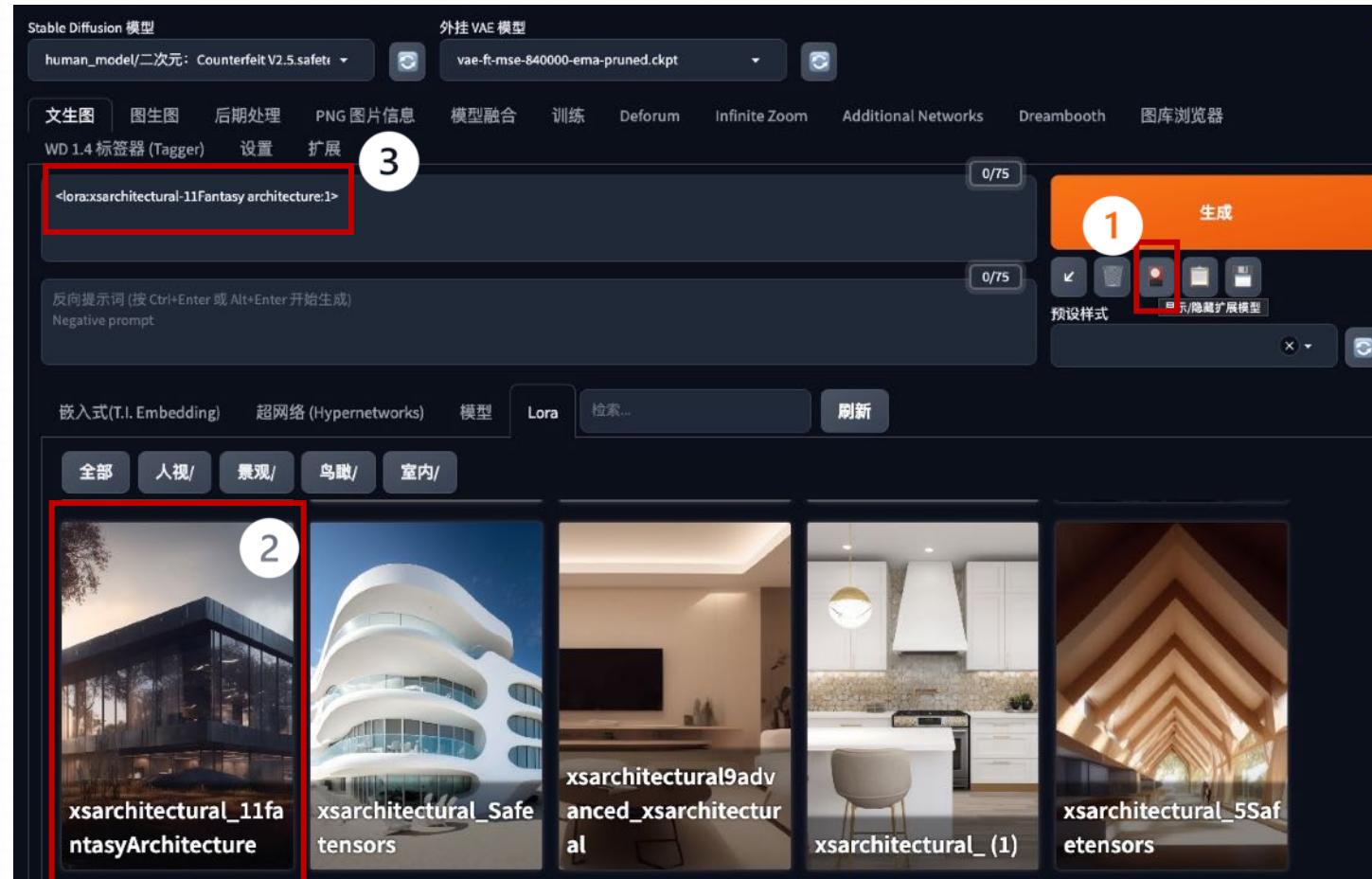


方法一：在Prompt中手动输入

格式：**<Lora: LoRA文件名（服务器中保存的）:权重>**

4. LoRA模型

模型使用方法



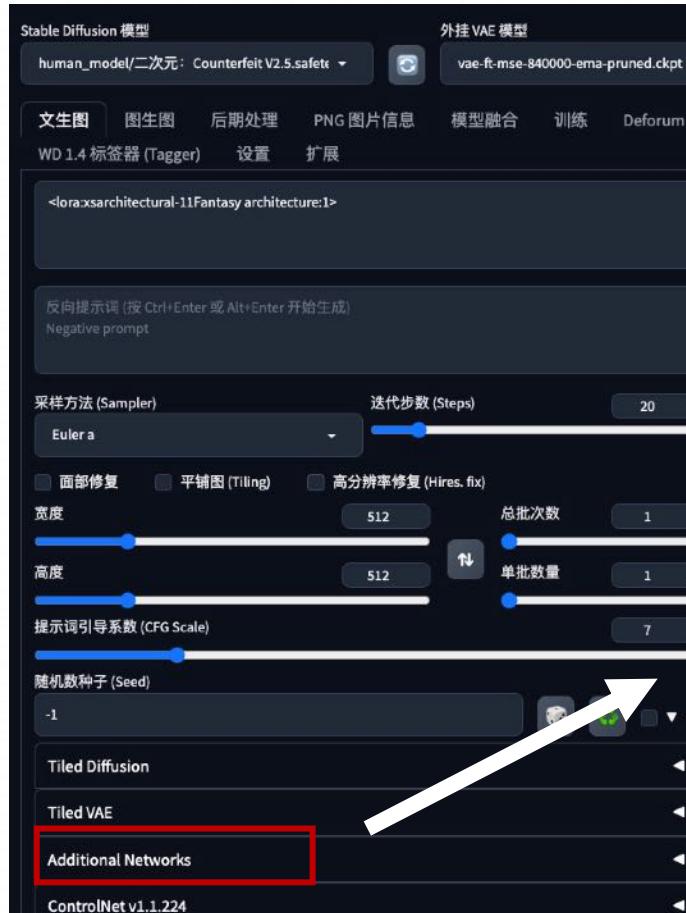
方法二：在附加模型选单中选择

步骤：

1. 点击“显示/隐藏扩散模型”按钮
2. 选择相应的LoRA模型
3. 自动将相应的模型添加到prompt中

4. LoRA模型

模型使用方法

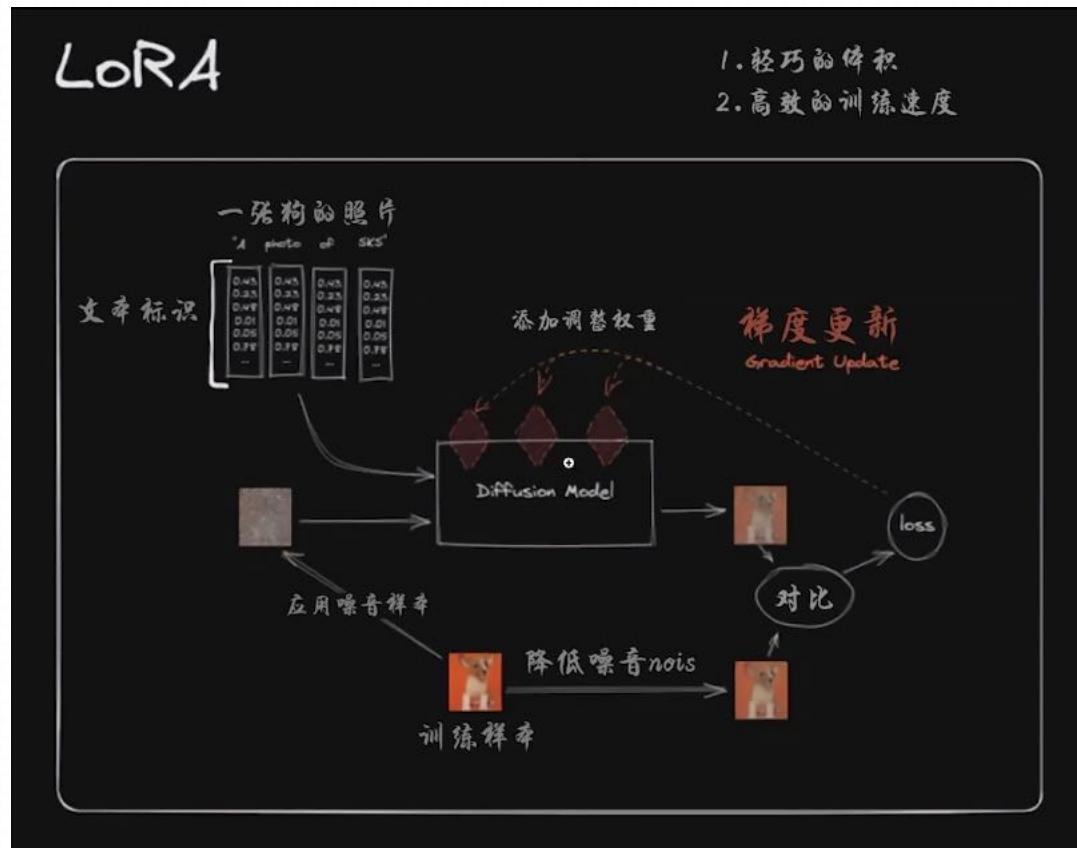


方法三：使用Additional Network添加

使用此种方法添加的LoRA模型不会显示在prompt中，可以简化prompt，使其更易读

4. LoRA模型

模型原理



LoRA的原理是冻结预训练好的模型权重参数，然后在每个Transformer块里注入可训练的层，由于不需要对模型的权重参数重新计算梯度，所以可以减少参数量和计算量，提高训练效率和生成质量

LoRA采用的方式是向原有的模型中插入新的数据处理层，这样就避免了去修改原有的模型参数，从而避免将整个模型进行拷贝的情况，同时其也优化了插入层的参数量，最终实现了一种很轻量化的模型调校方法

4. LoRA模型

模型推荐—Character人物形象

LoRA可以实现对于具体人物角色形象的创作把控，可以实现游戏动漫角色的二次创作和构建

应用案例：真人Coser → 真实系大模型 + 角色LoRA

Pokemon - Leaf Multiple Outfits 284 2.7K ★★★★★ 4

Updated: Jun 11, 2023 | CHARACTER | FEMALE | POKEMON | POKEMON FRPG

Default Outfit Sygna Suit (Old) Default Outfit (Old) Sygna Suit (Old) Team Rocket Outfit (Really Old) All-In-One

Details

Type	LORA
Downloads	948
Uploaded	Jun 10, 2023
Base Model	SD 1.5
Training	EPOCHS: 10
Trigger Words	LF
Hash	AUTOV2 0130AF7956

1 File

Reviews 1 version ratings Add Review

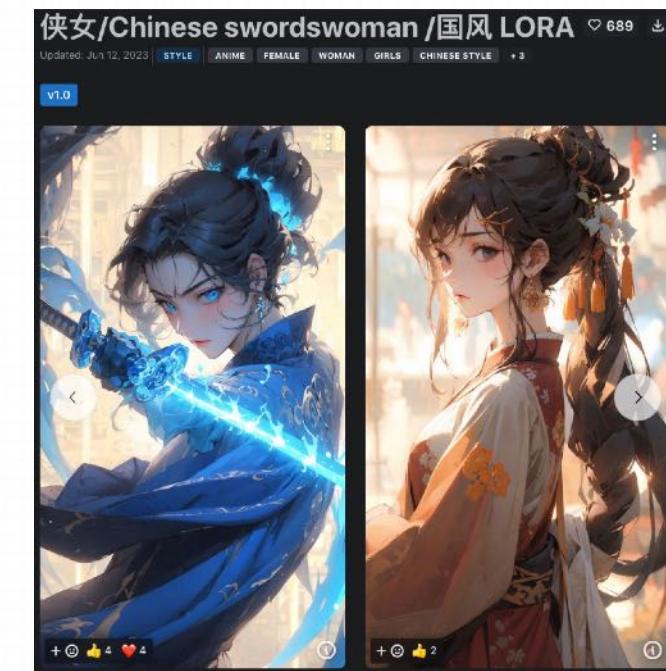
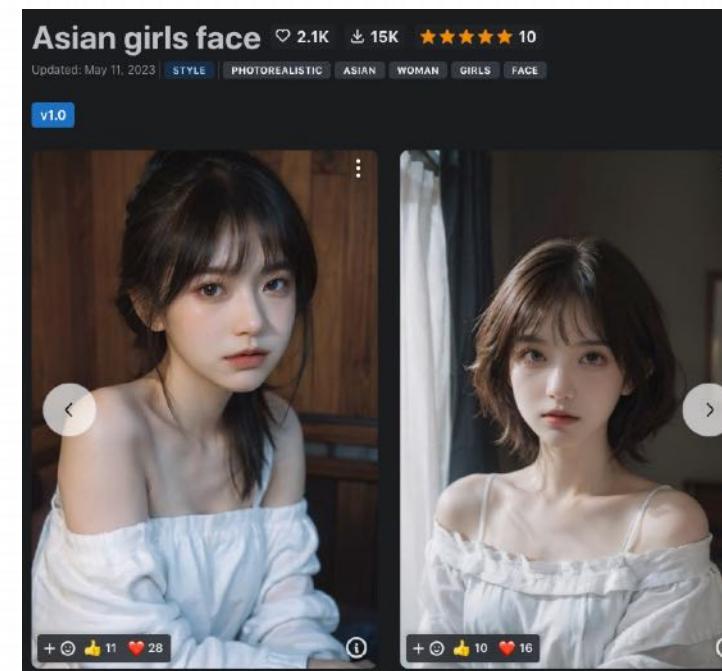
★★★★★ 5 out of 5 See Reviews



4. LoRA模型

模型推荐—Character人物形象

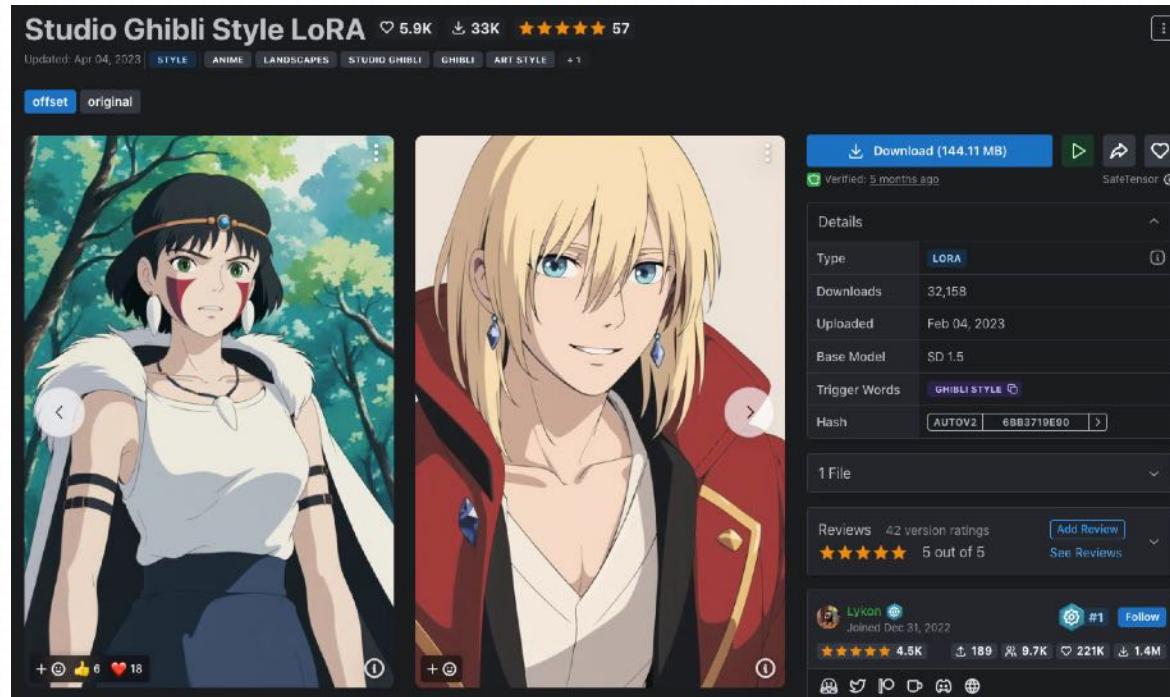
LoRA除了可以实现对于单一角色的把控之外，还可以控制生成某类特定的人物形象展示，例如亚洲男性人物、亚洲女性人物、国风侠女人物等等



4. LoRA模型

模型推荐—Style画风或风格

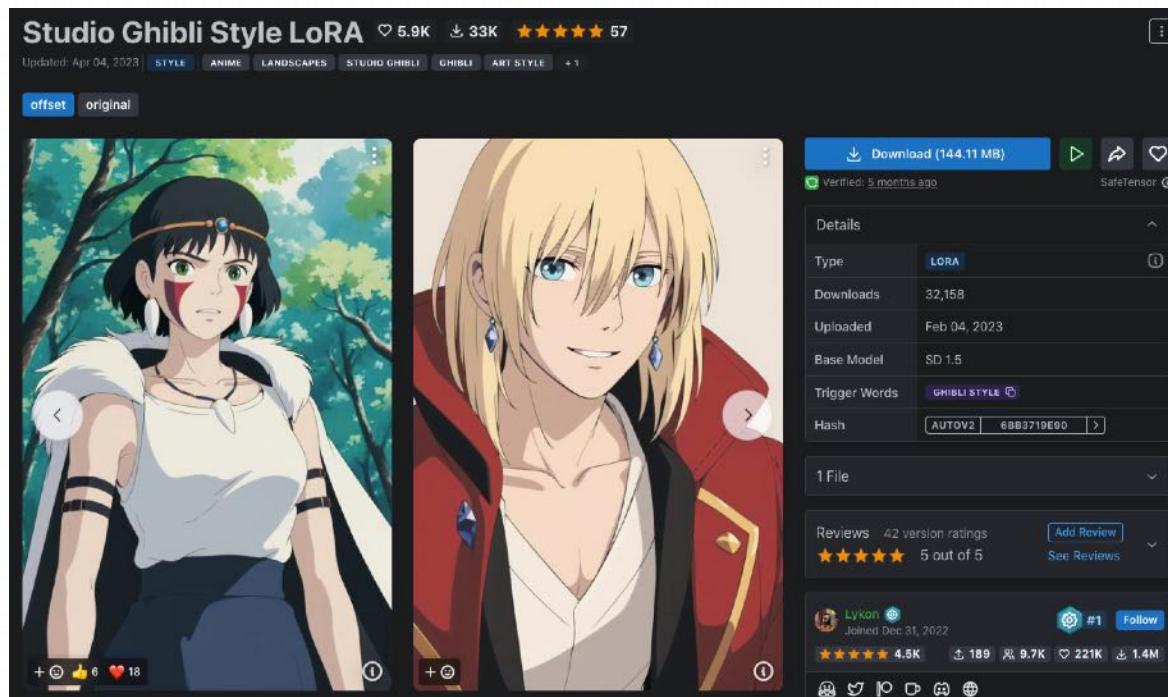
LoRA可以实现对于画风和视觉风格的微调，可以在大的checkpoint画风下进行更细致的微调，例如国画风格LoRA、吉卜力工作室画风LoRA、Kimpossible动画画风LoRA等等



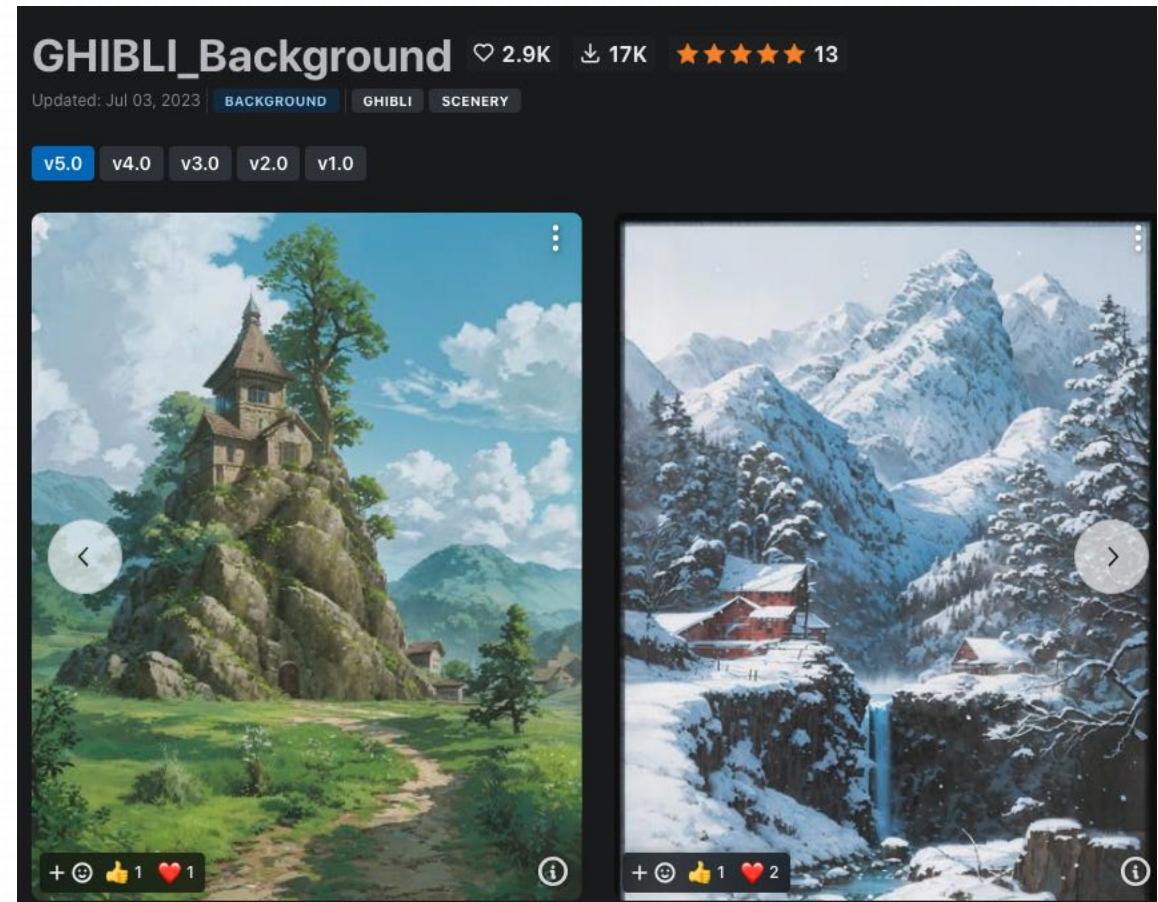
4. LoRA模型

模型推荐—Style画风或风格

吉卜力工作室画风LoRA 以及吉卜力风格背景LoRA



<https://civitai.com/models/6526/studio-ghibli-style-lora>

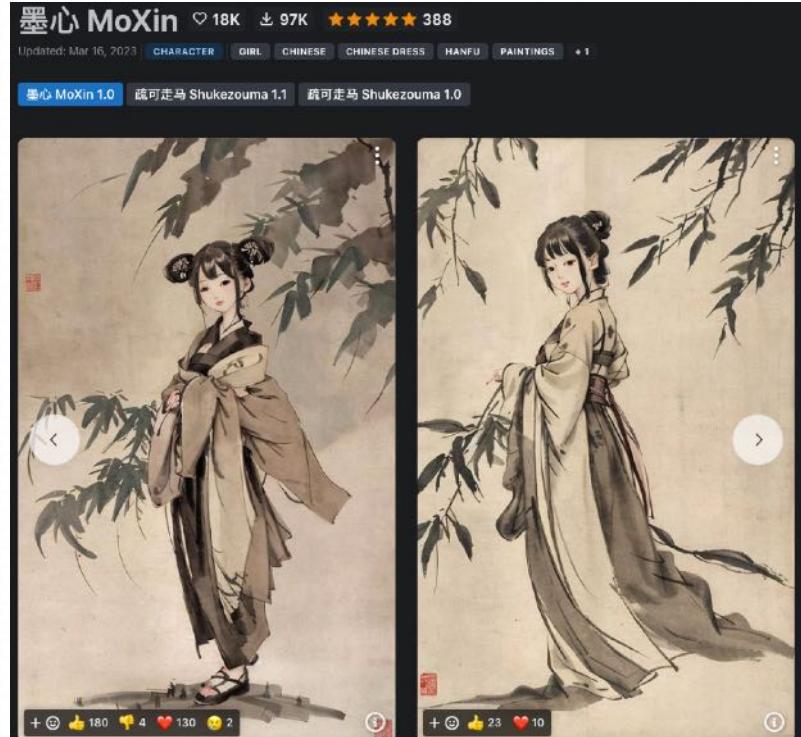


<https://civitai.com/models/54233/qhiblibackground>

4. LoRA模型

模型推荐—Style画风或风格

国画风格LoRA --墨心 MoXin



《墨心》——昔涓子《琴心》，王孙《巧心》，心哉美矣，故用之焉。
本品由安吉吴仓石、兴化板桥先生、八大山人、山阴伯年等大师之大小写意作品辅以现代人物训练而成。辅以恰当之提示词，诵先贤尊号，襄古今并用之意，明雅俗共举之美。



<https://civitai.com/models/12597/moxin>

4. LoRA模型

模型推荐—Style画风或风格

国画风格LoRA --墨心 MoXin



《疏可走马》——字画疏处可以走马，密处不使透风

这是一个和墨心搭配使用的构图Lora, 一旦使用并再最前前置提示词后，就会采用较大面积留白的构图风格。可以在版本处找到他

4. LoRA模型

模型推荐—Style画风或风格

国画风格LoRA --墨心 MoXin

注意事项：

1. CFG范围将会改变风格
 1~3 : 大小写意
 3~7 : 逐渐工笔
2. 推荐基础模型为ChilloutMix、国风3.2等
3. 《墨心》的推荐Lora权重为0.85以下
4. 《疏可走马》推荐Lora权重为0.7~1



4. LoRA模型

模型推荐—Concept概念

LoRA可以实现对于特定场景的创作，类似于某种特定画风或实现形式+形象的结合体

例如：Gacha splash LORA 针对各种氪金手游中的“抽卡”场景训练的LoRA



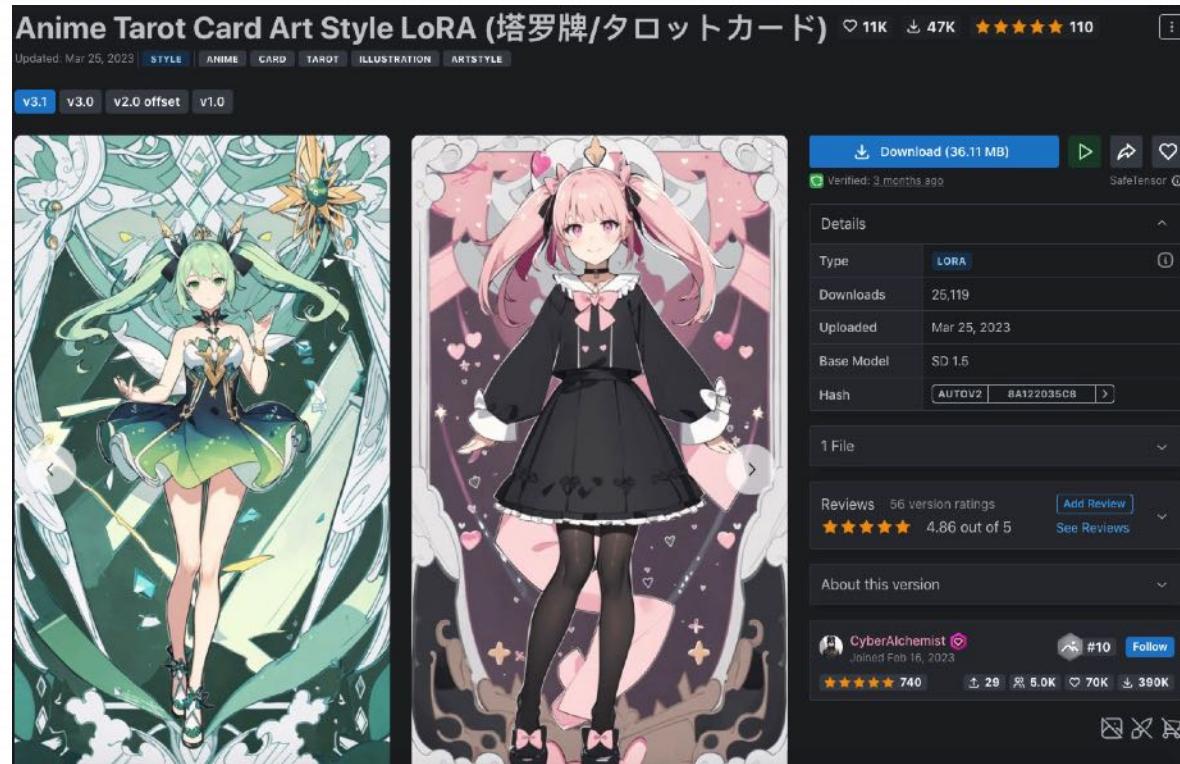
作者推荐Prompt模版：

[(white background:1.5)::5],
isometric OR hexagon , 1 girl, mid
shot, full body, <add your
background prompts here>

4. LoRA模型

模型推荐—Concept概念

LoRA可以实现对于特定场景的创作，类似于某种特定画风或实现形式+形象的结合体，例如塔罗牌、



<https://civitai.com/models/11177/anime-tarot-card-art-style-lora>

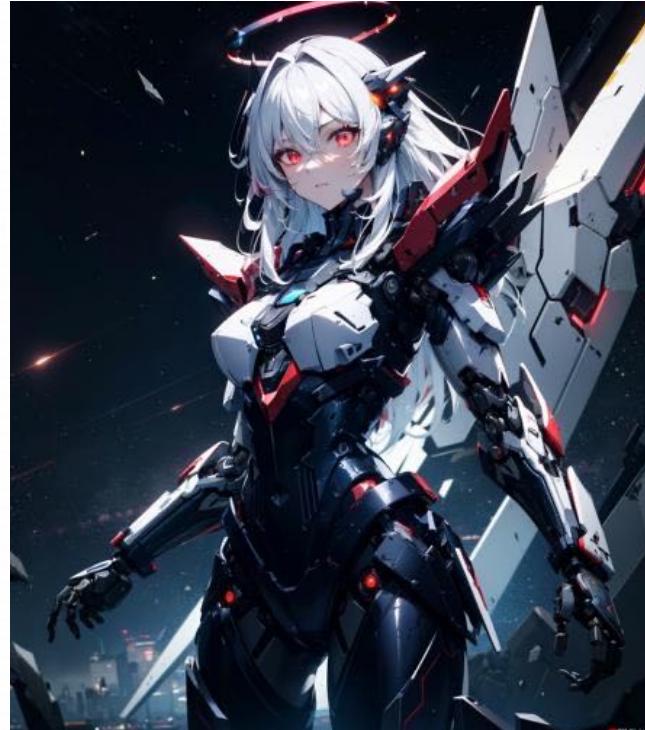
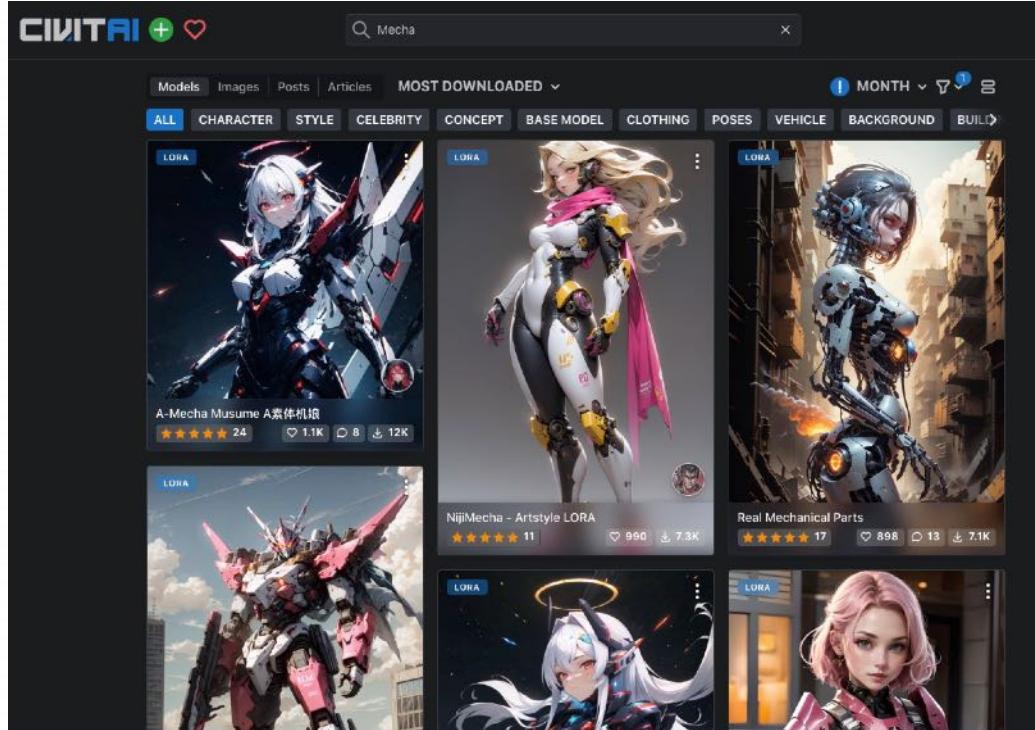
推荐主模型：Anything V4.5 & orangeminx VAE生成

推荐负面embedding：EasyNegative、badhandv4。

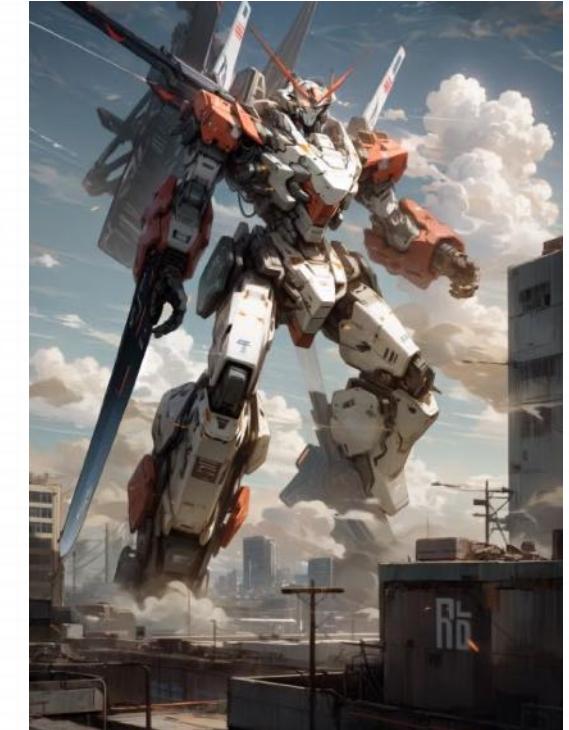
4. LoRA模型

模型推荐—Cloth服饰

许多常规的服饰可以直接用checkpoint画出，但LoRA可以实现对于更独特等服饰的绘制，例如赛博机甲服饰、汉服等等



<https://civitai.com/models/15464/a-mecha-musume-a>

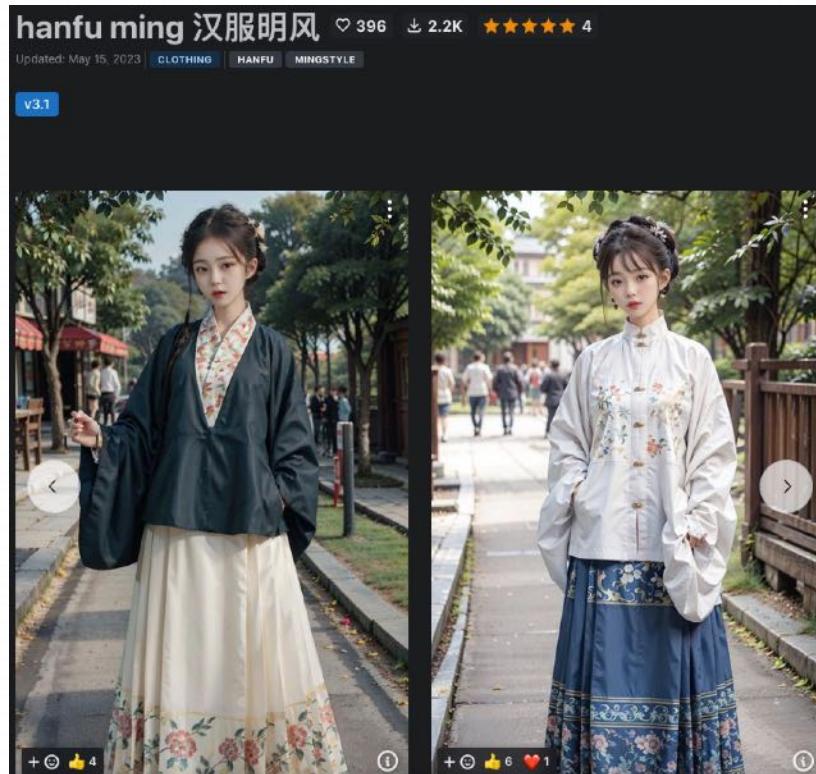


<https://civitai.com/models/69438/mechamix>

4. LoRA模型

模型推荐—Cloth服饰

许多常规的服饰可以直接用checkpoint画出，但LoRA可以实现对于更独特等服饰的绘制，例如赛博机甲服饰、汉服等等



<https://civitai.com/models/65314/hanfu-ming>



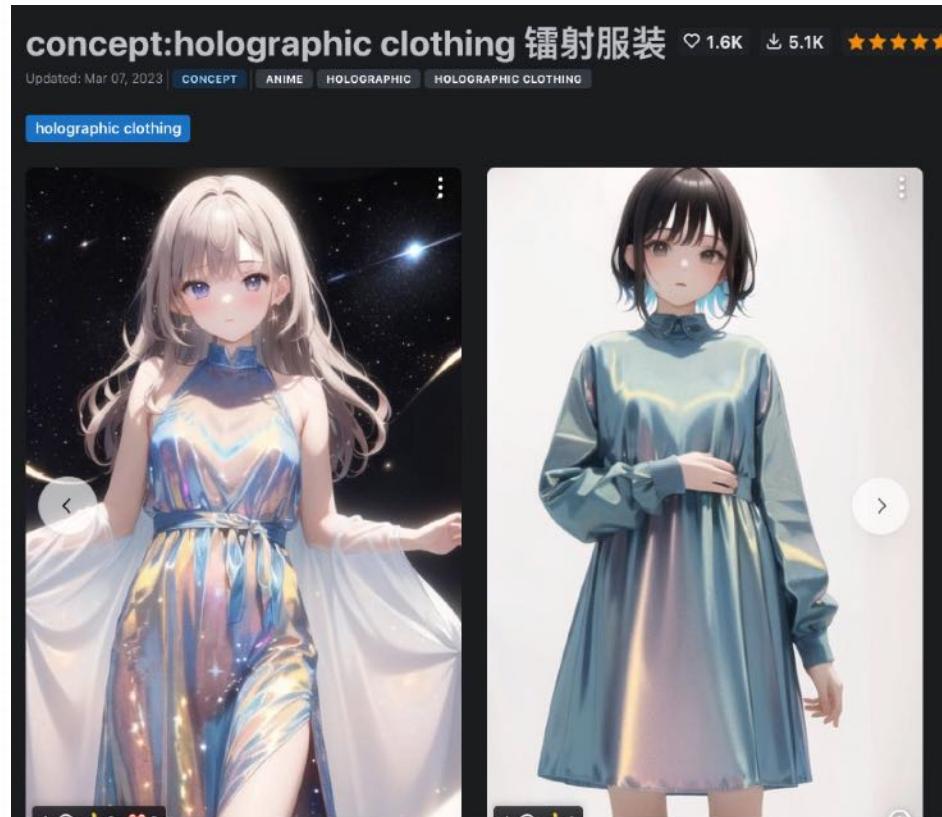
<https://civitai.com/models/47916/hanfu-song>

使用服饰类LoRA时要注意，权重一定不要太高，否则可能出现只有衣服没有人物的结果

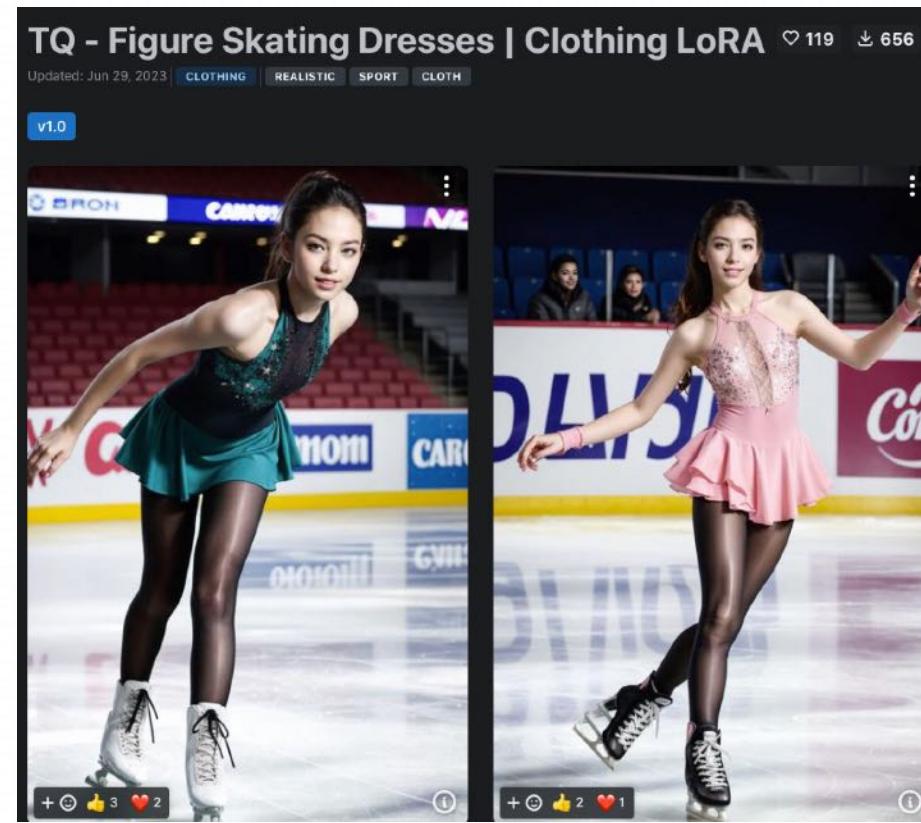
4. LoRA模型

模型推荐—Cloth服饰

许多常规的服饰可以直接用checkpoint画出，但LoRA可以实现对于更独特等服饰的绘制，例如赛博机甲服饰、汉服等等



<https://civitai.com/models/16451/conceptholographic-clothing>



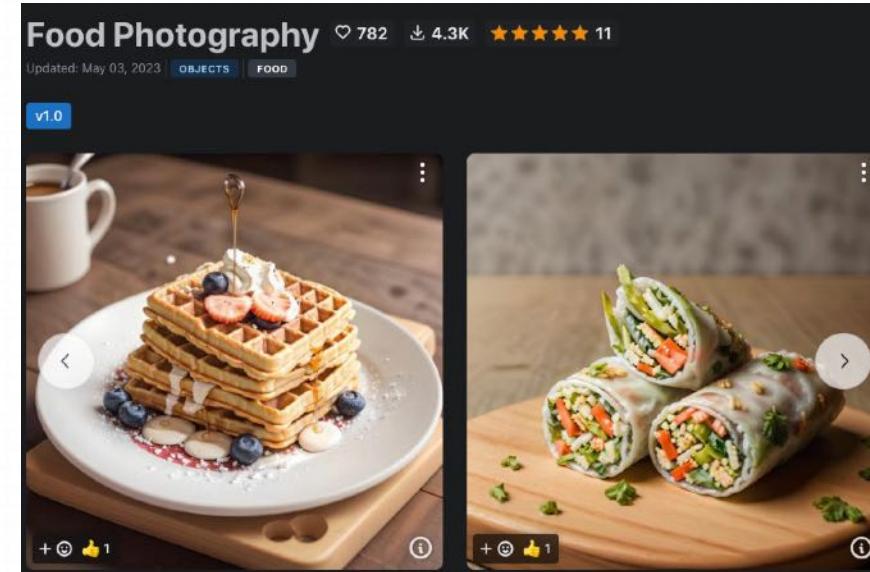
50

<https://civitai.com/models/99361/tq-figure-skating-dresses-or-clothing-lora>

4. LoRA模型

模型推荐—Object 物体/特定元素

这类LoRA是针对希望添加进画面的特定元素的，例如车、头盔、美食摄影、产品设计、产品蓝图等等



<https://civitai.com/models/54798/badass-cars>

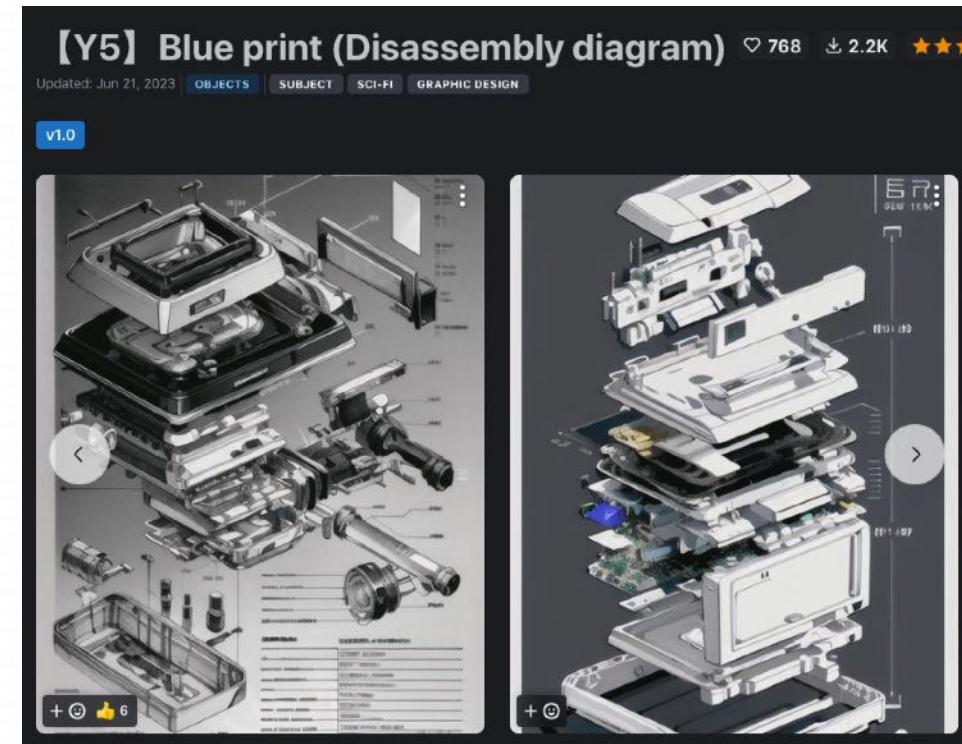
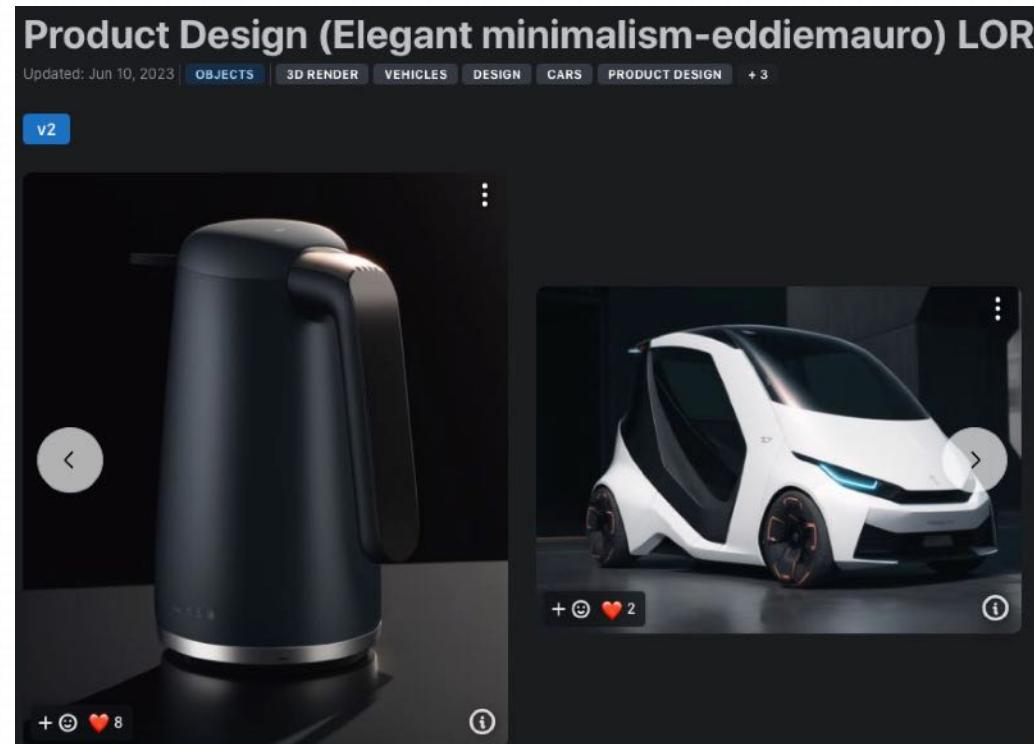
<https://civitai.com/models/45322/food-photography>

<https://civitai.com/models/84503/bottle-and-paper-bag>

4. LoRA模型

模型推荐—Object 物体/特定元素

这类LoRA是针对希望添加进画面的特定元素的，例如车、头盔、美食摄影、产品设计、产品蓝图等等



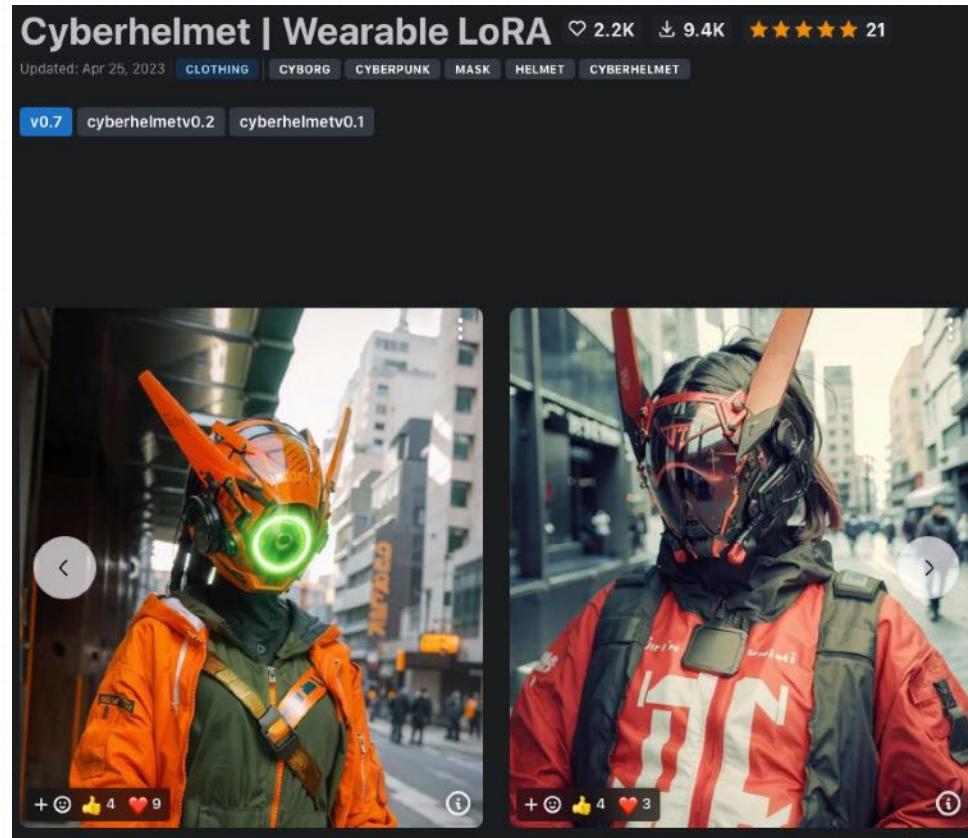
<https://civitai.com/models/58247/product-design-elegant-minimalism-eddiemauro-lora>

<https://civitai.com/models/33597/y5blue-print-disassembly-diagram>

4. LoRA模型

模型推荐—Object 物体/特定元素

这类LoRA是针对希望添加进画面的特定元素的，例如车、头盔、美食摄影、产品设计、产品蓝图等等



可以使用局部重绘的方法将其加入画面中：

1. 先绘制全图，此时不加LoRA
2. 再打开LoRA，使用重绘功能，加入Object

5. Hypernetwork模型

模型概念

Hypernetwork 模型，超网络模型，是 Stable Diffusion 的微调模型之一，一般用于改善生成图片的整体风格，Hypernetwork改善的画风比checkpoint中的画风要更为精细，例如雕塑风格、像素画、抽象画等等。、

Hypernetwork的作用与LoRA类似，但不如LoRA直接和高效，使用的频率较少

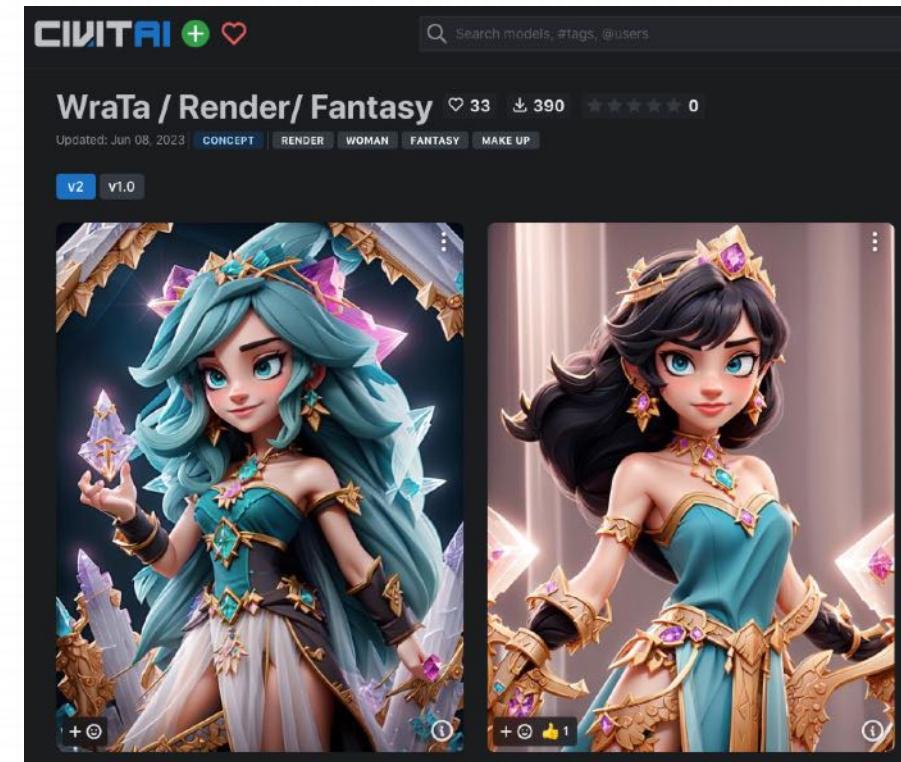
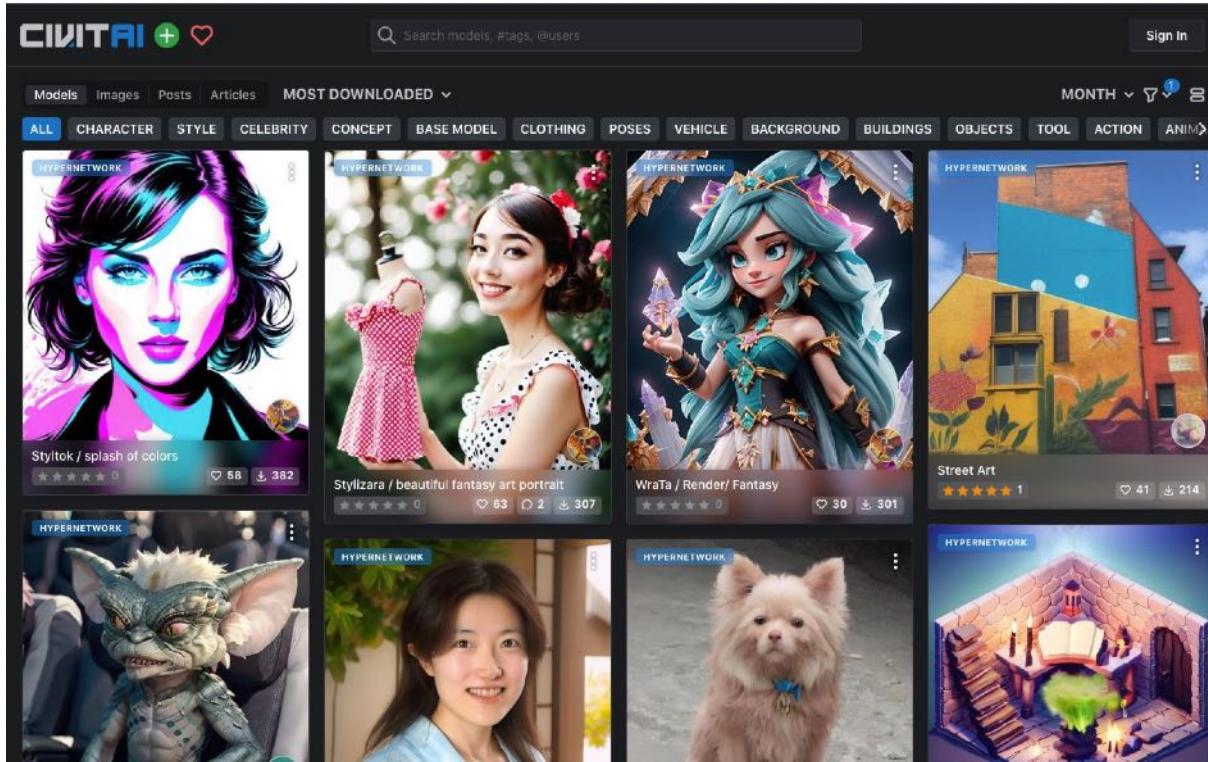
路径：\models\hypernetworks

Hypernetwork模型通常为 5~300 MB，常见格式为**pt**

5. Hypernetwork模型

模型概念

Hypernetwork 模型一般用于改善生成图片的整体风格，Hypernetwork改善的画风比checkpoint中的画风要更为精细，例如雕塑风格、像素画、抽象画等等



6. VAE模型

模型概念

VAE模型，Variational Auto Encoder 变分自解码器，负责将加噪后的潜空间数据转化为正常的图像，可以理解为AI绘图的一种“调色滤镜”，例如有的模型不加VAE的画出图会发灰或者发白。VAE模型主要影响的是画面的色彩质感练，通常用于提升图像色彩效果，并对图像细节进行细微调整。

目前多数比较新的模型都已经把VAE整合进大模型文件里，少数没有整合VAE的模型，其作者可能会推荐他们认为合适的VAE，也有一些普遍适用于大多数模型的VAE，例如kl-f8-anime2

路径：\models\VAE

VAE模型通常是300~800MB，常见格式为**pt、ckpt (safetensors)**

6. VAE模型

模型概念

VAE模型主要影响的是画面的色彩质感，通常用于提升图像色彩效果，并对图像细节进行细微调整。



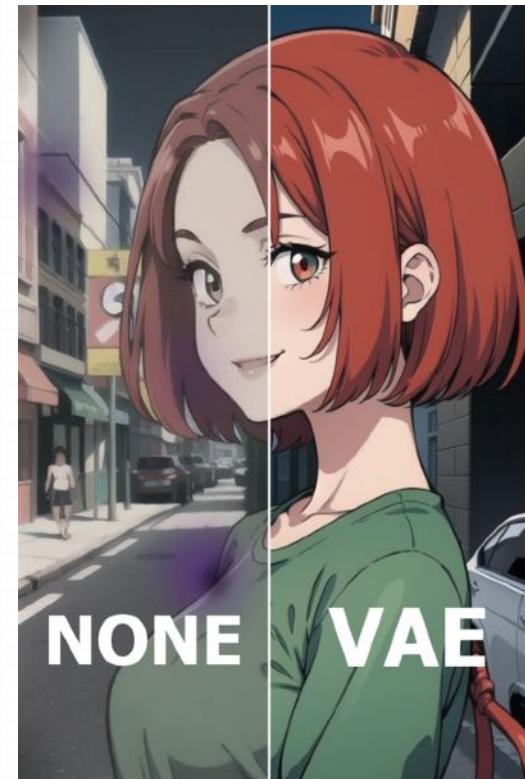
No VAE



kl-f8-anime2



vae-ft-mse-840000-ema-pruned



NONE VAE

7. 模型训练

LoRA模型训练---训练数据集准备

1. 训练素材处理

首先确定训练主题，比如某个人物、某种物品、某种画风等。

确定好画风后，就需要准备用于训练的素材图片，素材图的质量直接决定了模型的质量，好的训练集有以下要求：

1. 不少于 15 张的高质量图片，一般可以准备 20–50 张图；
2. 图片主体内容清晰可辨、特征明显，图片构图简单，避免其它杂乱元素；
3. 如果是人物照，尽可能以脸部特写为主（多角度、多表情），再放几张全身像（不同姿势、不同服装）；
4. 减少重复或相似度高的图片。

7. 模型训练

LoRA模型训练---训练数据集准备

1. 训练素材处理

首先确定训练主题，比如某个人物、某种物品、某种画风等。

确定好画风后，就需要准备用于训练的素材图片，素材图的质量直接决定了模型的质量，好的训练集有以下要求：

1. 不少于 15 张的高质量图片，一般可以准备 20–50 张图；
2. 图片主体内容清晰可辨、特征明显，图片构图简单，避免其它杂乱元素；
3. 如果是人物照，尽可能以脸部特写为主（多角度、多表情），再放几张全身像（不同姿势、不同服装）；
4. 减少重复或相似度高的图片。

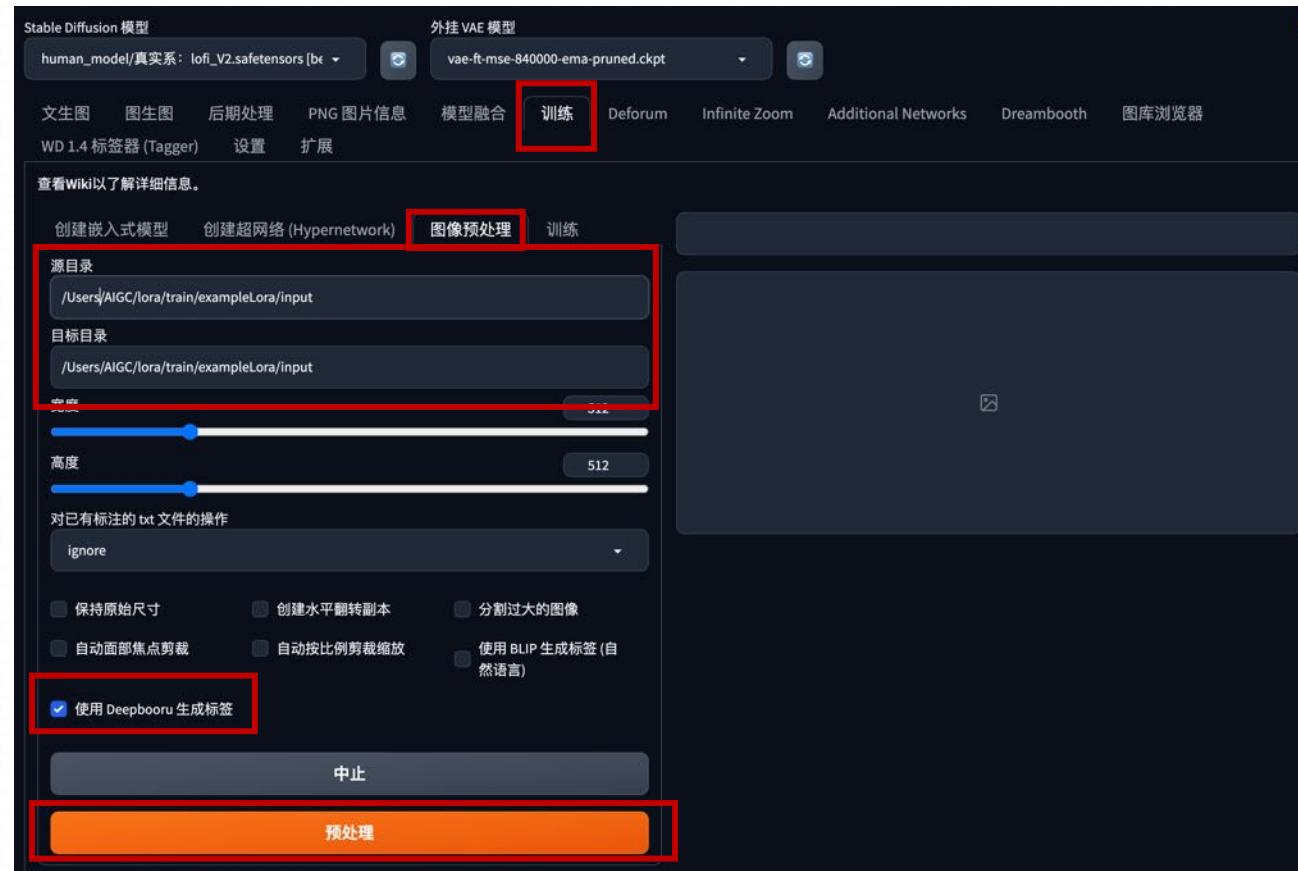
素材图准备完毕后，需要对图片做进一步处理：

- 对于低像素的素材图，可以用 Stable Diffusion 的 Extra 功能进行高清处理；
- 统一素材图分辨率，注意分辨率为 64 的倍数，显存低的可裁切为 512x512，显存高的可裁切为 768x768

7. 模型训练

LoRA模型训练---训练数据集准备

2. 图像预处理



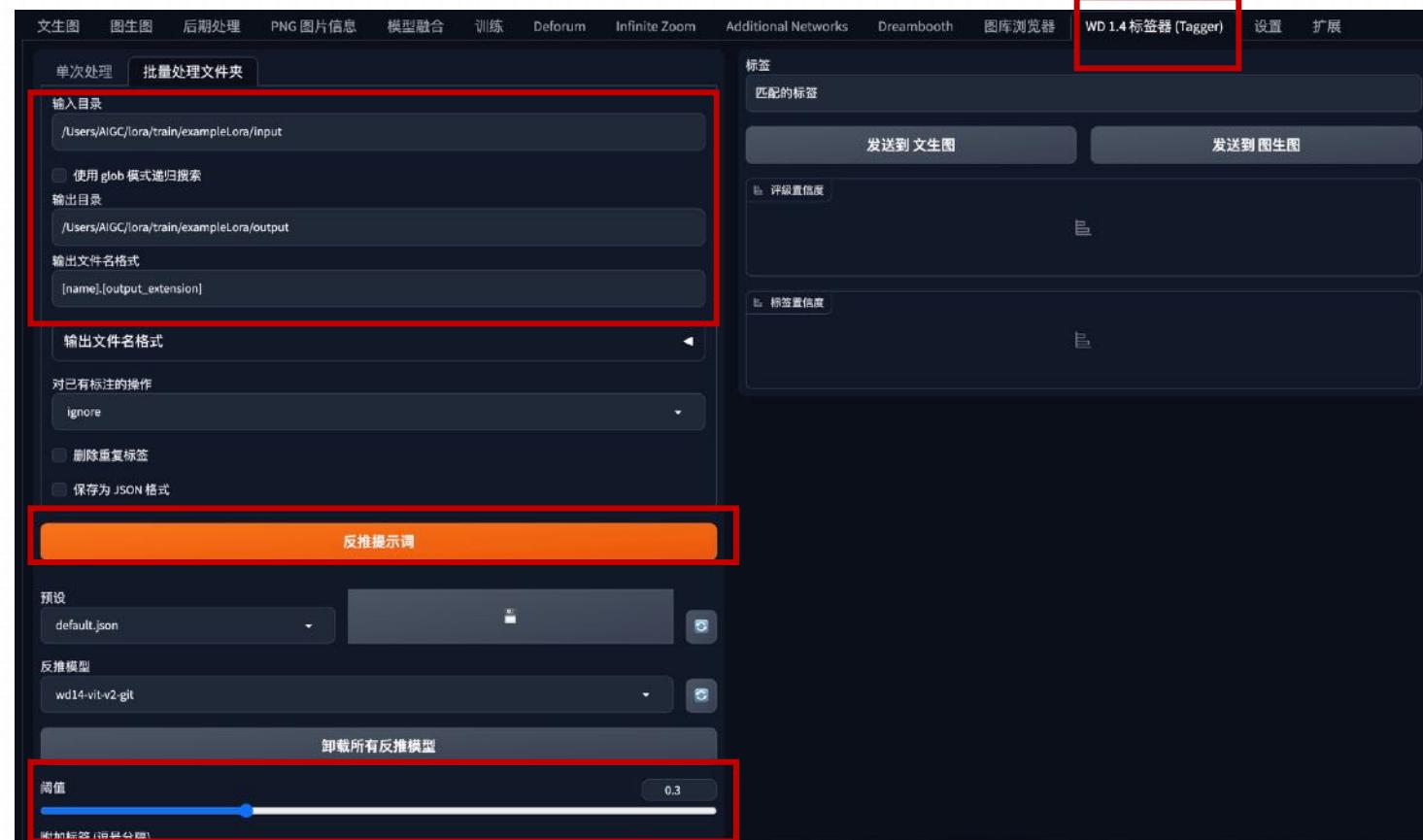
这一步的关键是对训练素材进行打标签，从而辅助 AI 学习。
这里介绍两种打标签的方法：

方法一：把训练素材文件夹路径填写到 Stable Diffusion 训练模块中的图像预处理功能，勾选生成 DeepBooru，进行 tags 打标签。

7. 模型训练

LoRA模型训练---训练数据集准备

2. 图像预处理



这一步的关键是对训练素材进行打标签，从而辅助 AI 学习。
这里介绍两种打标签的方法：

方法二：安装 tagger 标签器插件，进行 tags 打标签。选择批量处理，输入目录填写处理好的图片目录，设置标签文件输出目录，阈值设置为 0.3（生成尽可能多的标签来描述图片内容），开始打标签。

7. 模型训练

LoRA模型训练---训练数据集准备

3. 打标优化

预处理生成 tags 打标文件后，就需要对文件中的标签再进行优化，一般有两种优化方法：

方法一：保留全部标签

对这些标签不做删标处理，直接用于训练。一般在训练画风，或想省事快速训练人物模型时使用。

优点：不用处理 tags 省时省力，过拟合的出现情况低。

缺点：风格变化大，需要输入大量 tag 来调用、训练时需要把 epoch 训练轮次调高，导致训练时间变长。

批量打标：有时要优化等标签会比较多，可以尝试使用批量打标工具，如BooruDatasetTagManage

7. 模型训练

LoRA模型训练---训练数据集准备

3. 打标优化

方法二：删除部分特征标签

比如训练某个特定角色，要保留蓝眼睛作为其自带特征，那么就要将 blue eyes 标签删除，以防止将基础模型中的 blue eyes 引导到训练的 LoRA 上。简单来说**删除标签即将特征与 LoRA 做绑定**，保留的话画面可调范围就大。

- 一般需要删掉的标签：如人物特征 long hair, blue eyes 这类。
- 不需要删掉的标签：如人物动作 stand, run 这类，人物表情 smile, open mouth 这类，背景 simple background, white background 这类，画幅位置等 full body, upper body, close up 这类。

优点：调用方便，更精准还原特征。

缺点：容易导致过拟合，泛化性降低。

7. 模型训练

LoRA模型训练---训练

训练数据集准备完毕后，开始训练环境配置。一般有本地和云端两种训练环境：

本地训练：要求 N 卡，推荐 RTX 30 系列及以上显卡，训练环境可以用秋叶大佬的一键训练包，或者安装 Stable Diffusion WebUI 的训练插件。

<https://github.com/liasece/sd-webui-train-tools>

云端训练：如在 AutoDL、Google Colab 等云平台上训练，推荐 kohya-ss 训练脚本。云端训练的好处在于不占用本机资源，训练的同时还可以干其他事。

7. 模型训练

LoRA模型训练---云端训练

训练环境配置

这里推荐使用基于 kohya-ss 的训练脚本

进入 Colab 后，点击连接：

https://colab.research.google.com/github/WSH032/kohya-config-webui/blob/main/kohya_train_webui.ipynb

7. 模型训练

LoRA模型训练---云端训练

训练环境配置

```

kohya_train_webui.ipynb
文件 修改 视图 插入 代码执行程序 工具 帮助 无法保存更改

(一) 环境配置
▶ 初始化常量与挂载谷歌硬盘（只要重启过colab就要再运行一次）
    ① 挂载谷歌硬盘
    use_google_drive: ✅
    显示代码
    Mounted at /content/drive
    Tue May 30 01:49:48 2023
    +-----+
    | NVIDIA-SMI 525.85.12   Driver Version: 525.85.12   CUDA Version: 12.0 |
    +-----+
    | GPU Name Persistence-M| Bus-Id Disp.A Volatile Uncorr. ECC |
    | Fan Temp Perf Pwr:Usage/Cap| Memory-Usage GPU-Util Compute M. |
    |          MIG M.          |
    +-----+
    | 0 Tesla T4      Off 00000000:00:04.0 Off   0 |
    | N/A 35C P8 9W / 70W | 0MiB / 15360MiB | 0% Default |
    |                      N/A |
    +-----+
    | Processes:          GPU GI CI PID Type Process name GPU Memory |
    | ID ID ID   ID Usage |
    +-----+
    | NO running processes found |
    +-----+
    显示代码
    克隆github的库、安装依赖
  
```

① 建立训练文件夹

连接成功后，展开（一）环境配置：

1. 运行初始化常量与挂载谷歌硬盘。
2. 成功挂载谷歌硬盘后，在 content – drive 目录下建一个 Lora 训练文件夹，在训练文件夹中建立 input 文件夹用于放置输入数据集，建立 output 文件夹用于放置输出的训练模型。
3. input 文件夹内建一个训练数据集文件夹，注意该文件夹的命名有格式要求：Repeat 值_主题名，这里 Repeat 值的含义代表每张素材图的训练步数。越精细的图，Repeat 值也越高，一般二次元可以 15–30，三次元可以 50–100。

7. 模型训练

LoRA模型训练---云端训练

训练环境配置

② 运行克隆 github的库、安装依赖

modelName: Stable-Diffusion-v1-5.safetensors

base_model_url: "在此处插入"text"

③ 设置训练用底模型

modelName: 可以选择环境中已经提供的模型 如 Stable-Diffusion-v1-5.safetensors。

base_model_url: 也可以选择自定义模型，在 huggingface 上搜到想要模型的地址，复制过来。

7. 模型训练

LoRA模型训练---云端训练

训练参数配置



7. 模型训练

LoRA模型训练---云端训练

训练参数配置--基础参数设置

基础设置

1. **train_data_dir:** 训练集输入目录，把之前建立的数据集文件夹路径复制过来，如 /content/drive/MyDrive/Lora/input。
2. **底模:** 填入底模文件夹地址 /content/Lora/sd_model/，刷新加载底模。
3. **resolution:** 训练分辨率，支持非正方形，但必须是 64 倍数。一般方图 512x512、768x768，长图 512x768。
4. **batch_size:** 一次性送入训练模型的样本数，显存小推荐 1，12G 以上可以 2–6，并行数量越大，训练速度越快。
5. **max_train_epochs:** 最大训练的 epoch 数，即模型会在整个训练数据集上循环训练的次数。如最大训练 epoch 为 10，那么训练过程中将会进行 10 次完整的训练集循环，一般可以设为 5–10。
6. **network_dim:** 线性 dim，代表模型大小，数值越大模型越精细，常用 4~128，如果设置为 128，则 LoRA 模型大小为 144M。
7. **network_alpha:** 线性 alpha，一般设置为比 Network Dim 小或者相同，通常将 network dim 设置为 128，network alpha 设置为 64。

7. 模型训练

LoRA模型训练---云端训练

训练参数配置--基础参数设置

输出设置

1. 模型输出地址: 模型输出目录, 把之前建立的训练输出文件夹路径复制过来, 如 /content/drive/MyDrive/Lora/output
2. 输出模型名称: 可以填模型主题名, 如 bighand
3. 保存模型格式: 模型保存格式, 默认 safetensors

7. 模型训练

LoRA模型训练---云端训练

训练参数配置--基础参数设置

学习率设置

1. `unet_lr`: unet 学习率, 默认值为 0.0001
2. `text_encoder_lr`: 文本编码器的学习率, 一般为 unet 学习率的十分之一 0.00001
3. `lr_scheduler`: 学习率调度器, 用来控制模型学习率的变化方式, 一般默认。
4. `lr_warmup_steps`: 升温步数, 仅在学习率调度策略为“`constant_with_warmup`”时设置, 用来控制模型在训练前逐渐增加学习率的步数, 一般不动。
5. `lr_restart_cycles`: 退火重启次数, 仅在学习率调度策略为“`cosine_with_restarts`”时设置, 用来控制余弦退火的重启次数, 一般不动。

7. 模型训练

LoRA模型训练---云端训练

训练参数配置—采样参数设置

采样参数设置

1. Sample every n epochs: 每 N 轮采样一次，一般设置为 1。
2. Sample every n steps: 比如设置为 100，则代表每训练 100 步采样一次。
3. Sample prompt: 采样提示词，设置之后，LoRA 训练的同时会每隔设定的步数或轮次，生成一副图片，以此来直观观察 LoRA 训练的进展。

7. 模型训练

LoRA模型训练---云端训练

模型训练

训练参数配置保存完成后，点击
开始训练

训练完成后，模型文件会保存到
设置的输出目录。

```

kohya_train_webui.ipynb
文件 修改 视图 插入 代码执行程序 工具 帮助 无法保存更改
+ 代码 + 文本 复制到云端硬盘
https://localhost:17061/
开始训练
若正确运行，训练完成后，模型会自动保存至你在WebUI里设置的地址
默认训练配置文件在 /content/sd-scripts/config_file.toml
默认采样参数文件在 /content/sd-scripts/sample_prompts.txt

如果你想用自己的配置文件，或者采样文件，请填入下方 填入意味着启用
config_file_self_path: "在此处插入"text"
sample_prompts_self_path: "在此处插入"text"
显示代码

running training / 学習開始
num train images * repeats / 学習画像の数*繰り返し回数: 600
num reg images / 正則化画像の数: 0
num batches per epoch / epochのバッチ数: 300
num epochs / epoch数: 5
batch size per device / バッチサイズ: 2
gradient accumulation steps / 勾配を合計するステップ数 = 1
total optimization steps / 学習ステップ数: 1500
steps: 0% 0/1500 [00:00<0, ?it/s]
epoch 1/5
steps: 20% 300/1500 [07:19<29:19, 1.47s/it, loss=0.0303]
saving checkpoint: /content/drive/MyDrive/Lora/output/bighand/bighand-000001.safetensors

generating sample images at step / サンプル画像生成 ステップ: 300
prompt: (masterpiece, best quality, hires:1.2), lgirl, solo,
negative_prompt: (worst quality, bad quality:1.4), lowres, bad anatomy, bad hands, text, error, missing fingers, extra digit, fewer digits, cropped, w
height: 768
width: 768
sample_steps: 24
scale: 7.0

0% 0/24 [00:00<0, ?it/s]
4% 1/24 [00:00<0:01, 1.86it/s]
8% 2/24 [00:01<0:01, 1.95it/s]
12% 3/24 [00:01<0:01, 1.97it/s]
17% 4/24 [00:02<0:01, 1.99it/s]
21% 5/24 [00:02<0:01, 1.99it/s]
25% 6/24 [00:03<0:01, 2.00it/s]
29% 7/24 [00:03<0:01, 2.00it/s]
33% 8/24 [00:04<0:01, 2.00it/s]
38% 9/24 [00:04<0:01, 2.00it/s]
42% 10/24 [00:05<0:01, 2.00it/s]
46% 11/24 [00:05<0:01, 2.00it/s]
50% 12/24 [00:06<0:01, 2.00it/s]
54% 13/24 [00:06<0:01, 2.00it/s]
58% 14/24 [00:07<0:01, 2.00it/s]
62% 15/24 [00:07<0:01, 2.00it/s]
67% 16/24 [00:08<0:01, 2.00it/s]
71% 17/24 [00:08<0:01, 2.00it/s]
75% 18/24 [00:09<0:01, 2.00it/s]
79% 19/24 [00:09<0:01, 2.00it/s]
83% 20/24 [00:10<0:01, 2.00it/s]
88% 21/24 [00:10<0:01, 2.00it/s]
92% 22/24 [00:11<0:01, 2.00it/s]
96% 23/24 [00:11<0:01, 2.00it/s]
100% 24/24 [00:12<0:01, 1.99it/s]

epoch 2/5
steps: 40% 600/1500 [14:56<22:24, 1.49s/it, loss=0.0268]
saving checkpoint: /content/drive/MyDrive/Lora/output/bighand/bighand-000002.safetensors

```

7. 模型训练

LoRA模型训练---本地训练

训练参数环境配置

The screenshot shows the LoRA training interface with two main sections highlighted by red boxes:

- Section 1 (Top Left): Current Dataset to be trained**

 - Shows a grid of images of Hatsune Miku.
 - Below the images is a "Textbox" containing the text: "Dataset: 20_mii".
 - A red number "1" is overlaid on this section.

- Section 2 (Bottom Right): Train base model**

 - Contains fields for "Train base model" (set to "v1-5-pruned-emaonly.safetensors [6ce0161689]"), "Batch size" (set to "1"), "Number of epochs" (set to "10"), and "Save every n epochs" (set to "1").
 - Includes checkboxes for "Generate all checkpoint preview after train finished" and "Begin train".
 - A red number "2" is overlaid on this section.

1 -- 这个区域显示了真正会用于训练的数据。

2 -- 这个区域是训练的参数设置区域。

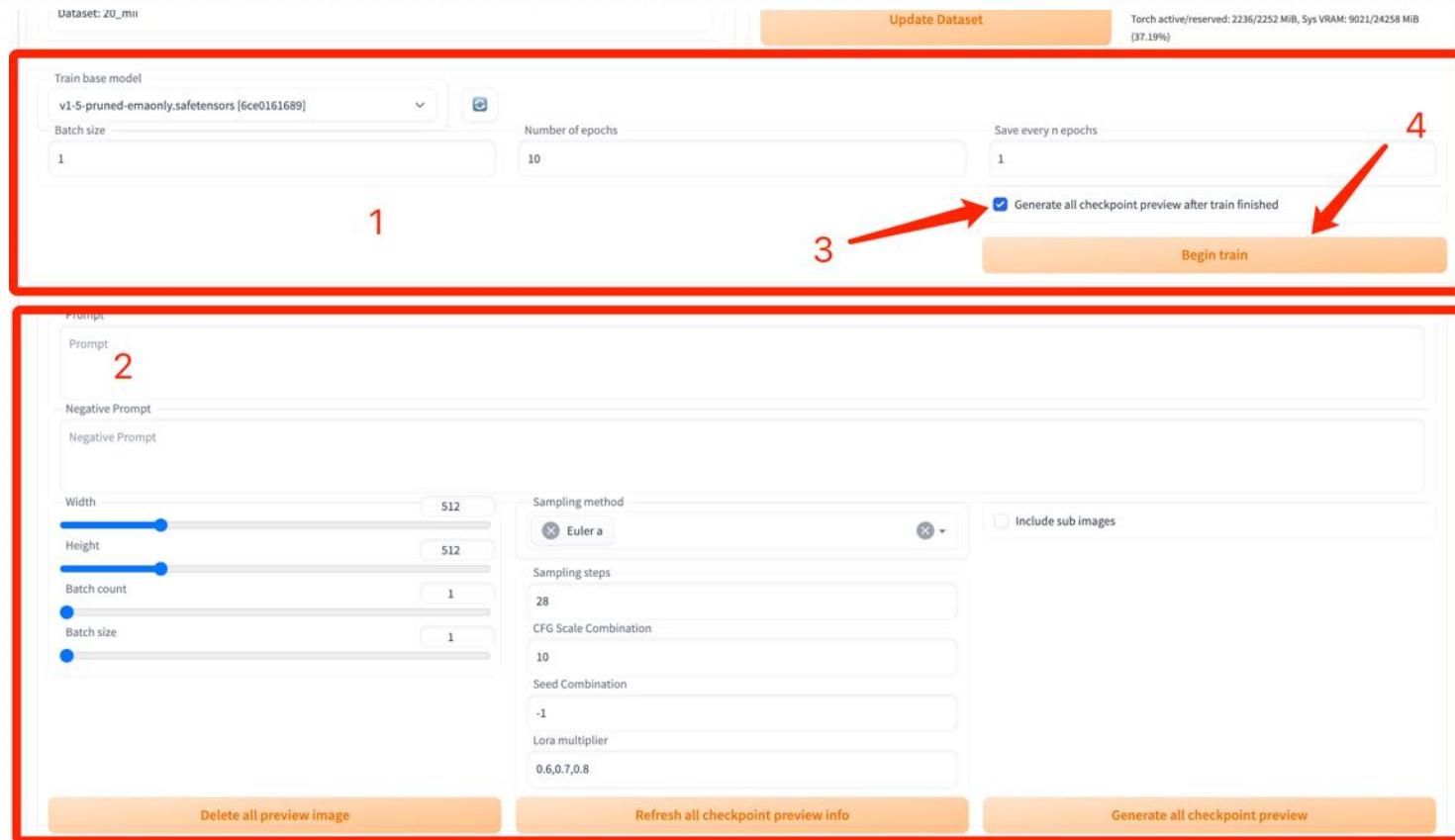
<https://www.uisdc.com/lora-model>

<https://github.com/liasece/sd-webui-train-tools>

7. 模型训练

LoRA模型训练---本地训练

训练参数环境配置



这里有一些训练的参数，你可以根据自己的需要进行调整。默认参数已经可以满足一般情况。Train base model 很重要，一个好的基础模型很大程度影响你的训练效果。

- 1 -- 训练配置区域。
- 2 -- 预览参数配置区域。因为训练结束后可以自动预览训练结果，所以这里有一些预览的参数可以一并配置。
- 3 -- 训练完成后自动运行预览图生成。
- 4 -- 开始训练。

<https://www.uisdc.com/lora-model>

<https://github.com/liasece/sd-webui-train-tools>

7. 模型训练

LoRA模型训练---本地训练

训练



```

100% | 8/8 [00:01<00:00, 7.79it/s]
import network module: networks.lora
create LoRA network. base dim (rank): 128, alpha: 64.0
create LoRA for Text Encoder: 72 modules.
create LoRA for U-Net: 192 modules.
enable LoRA for text encoder
enable LoRA for U-Net
prepare optimizer, data loader etc.
use Adafactor optimizer | {'weight_decay': 0.1, 'relative_step': True}
relative_step is true / relative_stepがtrueです
learning rate is used as initial_lr / 指定したlearning rateはinitial_lrとして使用されます
use adafactor_scheduler / スケジューラにadafactor_schedulerを使用します
override steps. steps for 10 epochs is / 指定エポックまでのステップ数: 1600
running training / 学習開始
  num train images * repeats / 学習画像の数×繰り返し回数: 160
  num reg images / 正則化画像の数: 0
  num batches per epoch / 1epochのバッチ数: 160
  num epochs / epoch数: 10
  batch size per device / バッチサイズ: 1
  gradient accumulation steps / 勾配を合計するステップ数 = 1
  total optimization steps / 学習ステップ数: 1600
steps:  0%
epoch 1/10
steps:  1%

```

The terminal window shows the configuration and start of a LoRA training session. It includes parameters like rank (128), alpha (64.0), and learning rate (initial_lr). The process starts with 1600 steps over 10 epochs. The progress bar at the bottom right indicates the current step (14/1600) and time (00:05<11:05, 2.38it/s, loss=0.16).

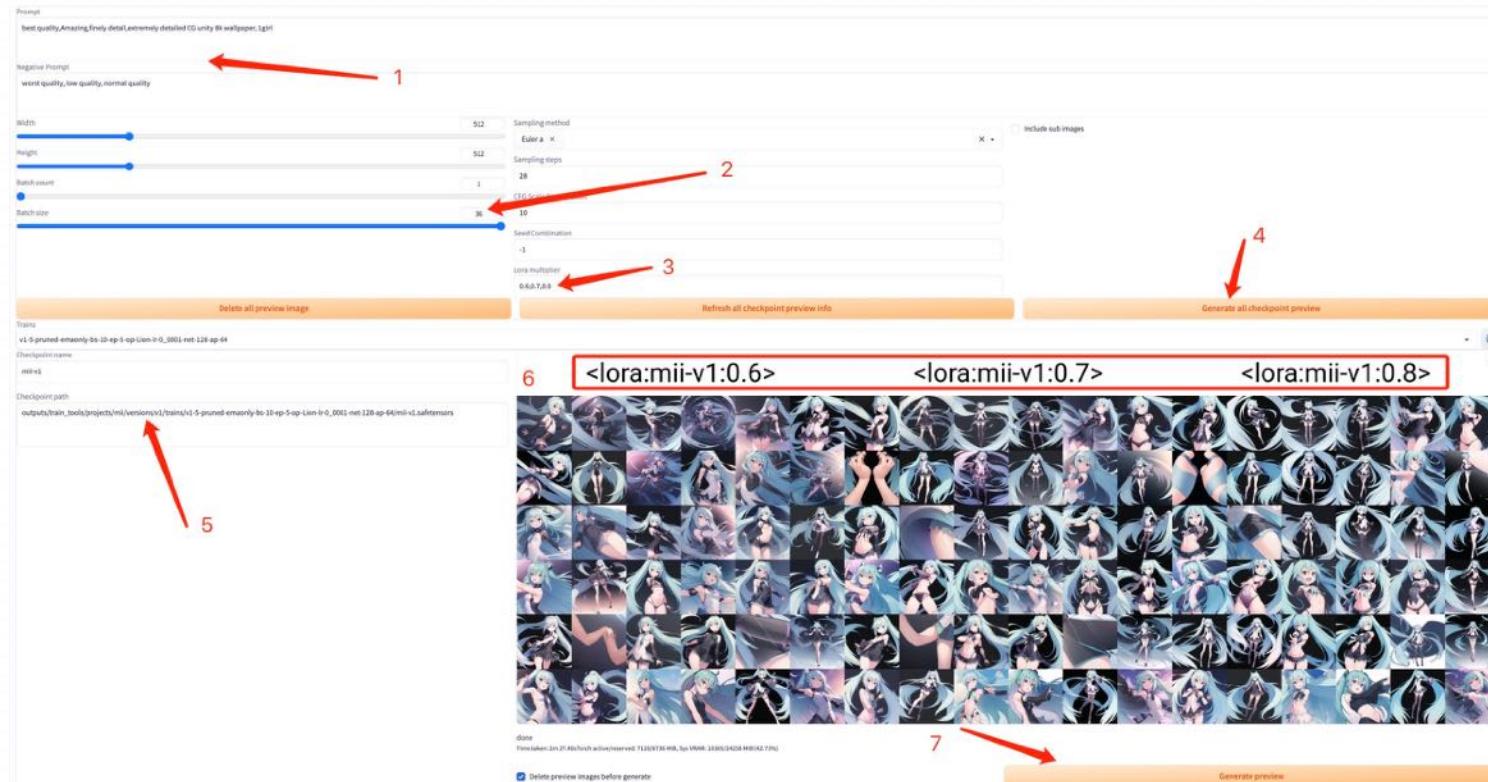
在 stable-diffusion-webui 的运行命令行中可以看到训练的过程及进度。

<https://www.uisdc.com/lora-model>

7. 模型训练

LoRA模型训练---本地训练

预览训练结果



不同的训练数据/训练参数/训练次数都会导致训练的效果不一样。
可以在这里预览训练结果。方便地挑选较好的结果。

1 -- 要在检查点上预览的关键词。和 txt2img 中的含义基本一致。

2 -- 如果显存允许，可以设置成 4 或者 9 等 $n \times n$ 的值，这样可以在一个图中预览更多的结果。

3 -- 这里是想为每个检查点预测几个 Lora 权重，用 "," 分隔。

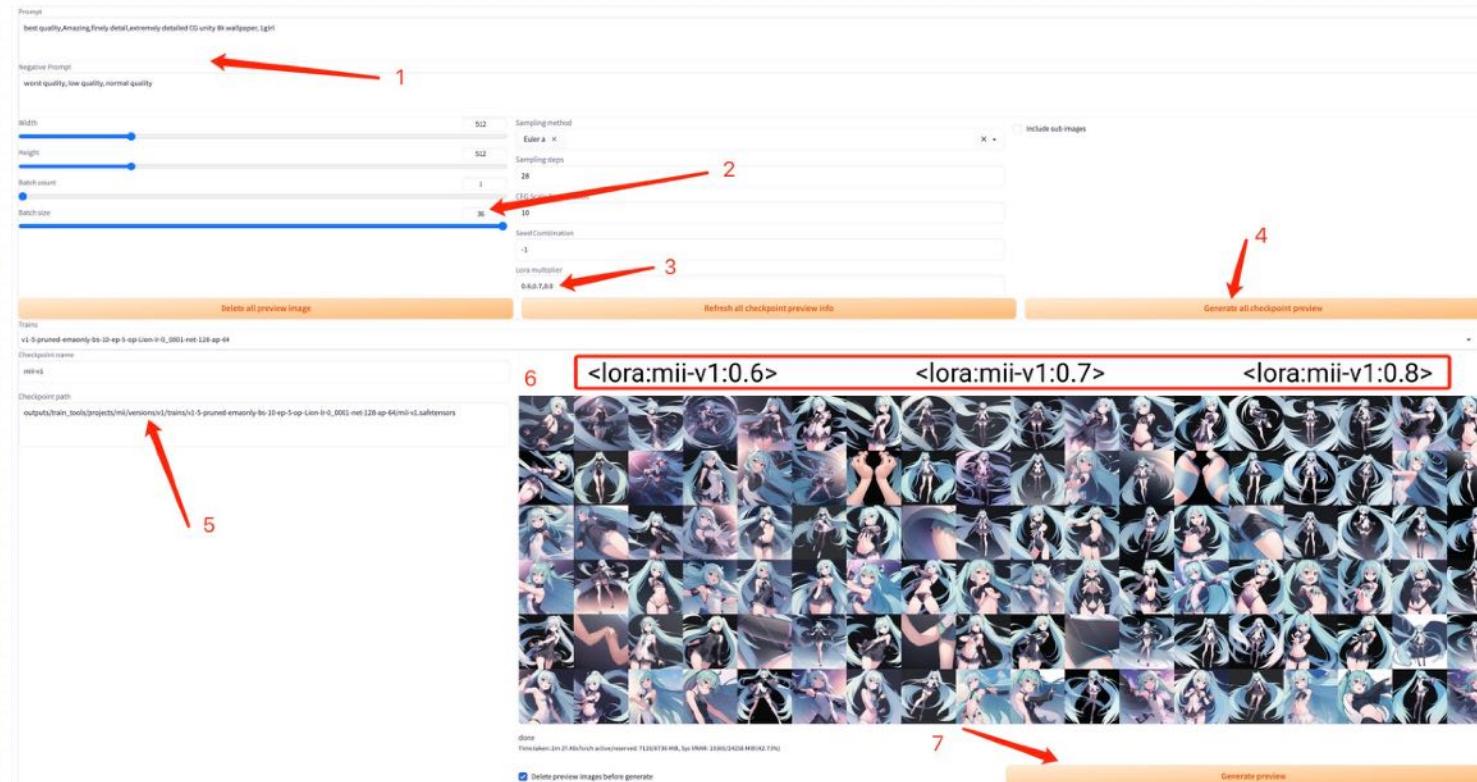
<https://www.uisdc.com/lora-model>

<https://github.com/liasece/sd-webui-train-tools>

7. 模型训练

LoRA模型训练---本地训练

预览训练结果



不同的训练数据/训练参数/训练次数都会导致训练的效果不一样。
可以在这里预览训练结果。方便地挑选较好的结果。

4 -- 点击这个按钮，将为下面列表中的所有检查点执行相同的 seed 的预测图片生成，这样可以更方便地对比不同检查点的效果。

5 -- 这个检查点的保存路径。如果你认为它的结果好，可以去这里找到它。

6 -- 类似于 xyz 脚本，这一张图列出了所有参数下它的预测结果。

7 -- 每个检查点下都有自己的预测按钮，点击它就会只用这个检查点生成图片。

<https://www.uisdc.com/lora-model>

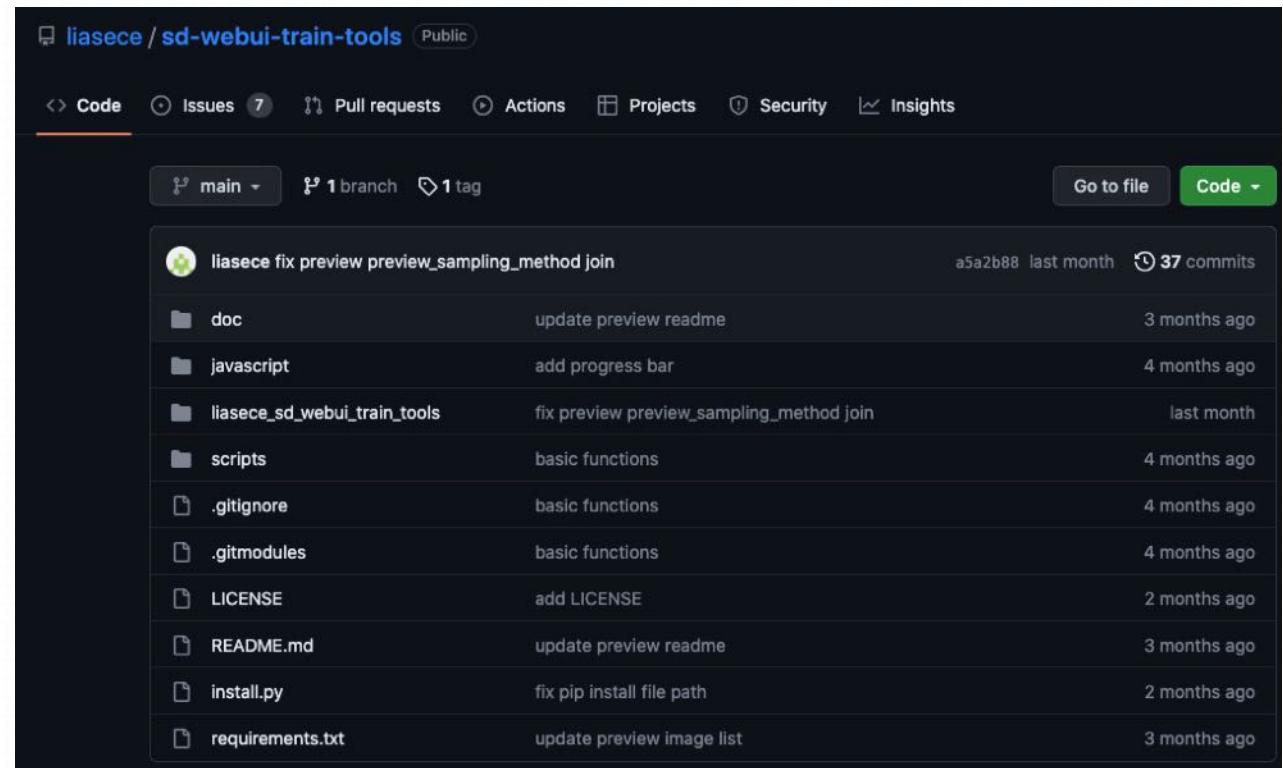
<https://github.com/liasece/sd-webui-train-tools>

7. 模型训练

LoRA模型训练---本地训练

保存训练结果

目前没有在 UI 中提供下载模型的功能，可以去 stable-diffusion-webui > output > train_tools > 工程名 > versions > 版本名 > checkpoints 中找到训练好的模型。



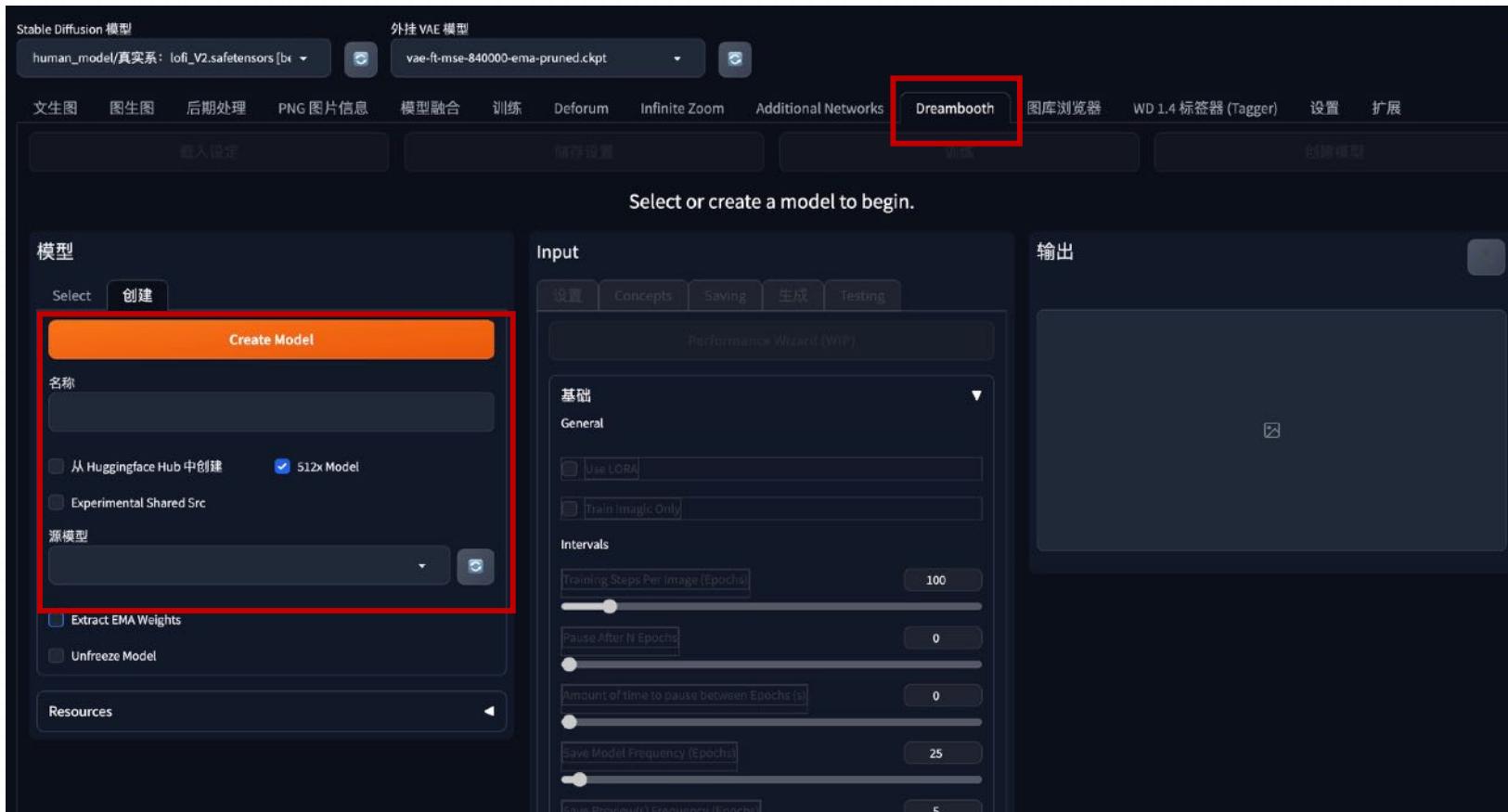
The screenshot shows a GitHub repository page for 'liasece / sd-webui-train-tools'. The repository is public and has 1 branch and 1 tag. The commit history lists 37 commits from user 'liasece' over the last month. The commits include fixes for preview sampling methods, updates to documentation and progress bars, and basic function implementations. The commits are dated from 3 months ago to last month.

Commit Message	Date
fix preview preview_sampling_method join	a5a2b68 last month
update preview readme	3 months ago
add progress bar	4 months ago
fix preview preview_sampling_method join	last month
basic functions	4 months ago
basic functions	4 months ago
basic functions	4 months ago
add LICENSE	2 months ago
update preview readme	3 months ago
fix pip install file path	2 months ago
update preview image list	3 months ago

<https://www.uiscd.com/lora-model>

7. 模型训练

Dreambooth模型训练—创建模型



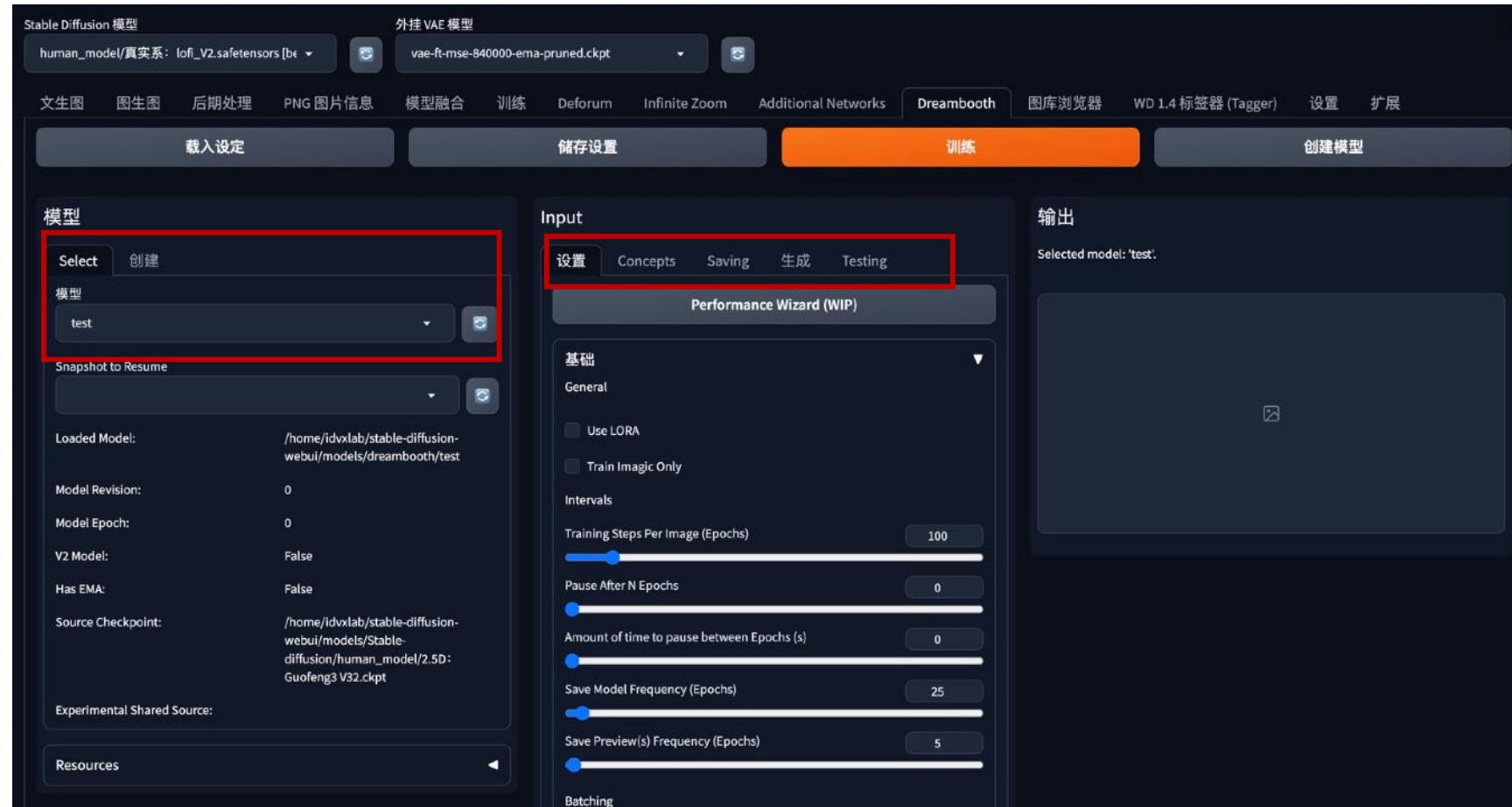
图像预处理部分与前面的LoRA模型相似

首先是创建模型

Source Checkpoint 初始化模型
(ckpt文件模型)，设置好之后，点击黄色的 Create Model，稍等在 Output 处可以看到 successfully 的日志，以及在左边 Model Selection 处可以看到自己刚 Create 出来的模型

7. 模型训练

Dreambooth模型训练—设置参数

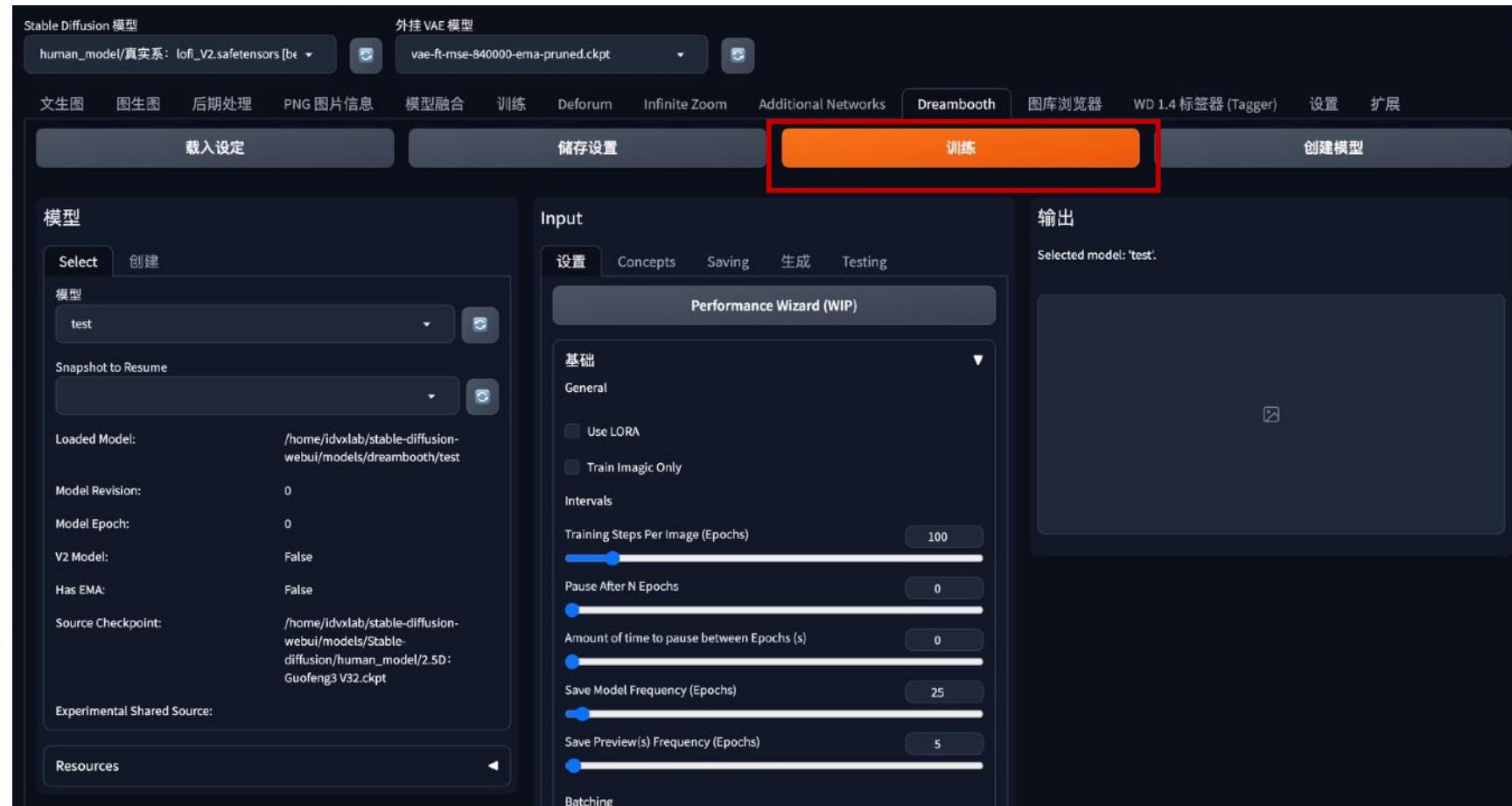


选择上一步创建的模型，并在中间模块选择相应的参数

参数选择方法参考下方git链接

7. 模型训练

Dreambooth模型训练—训练

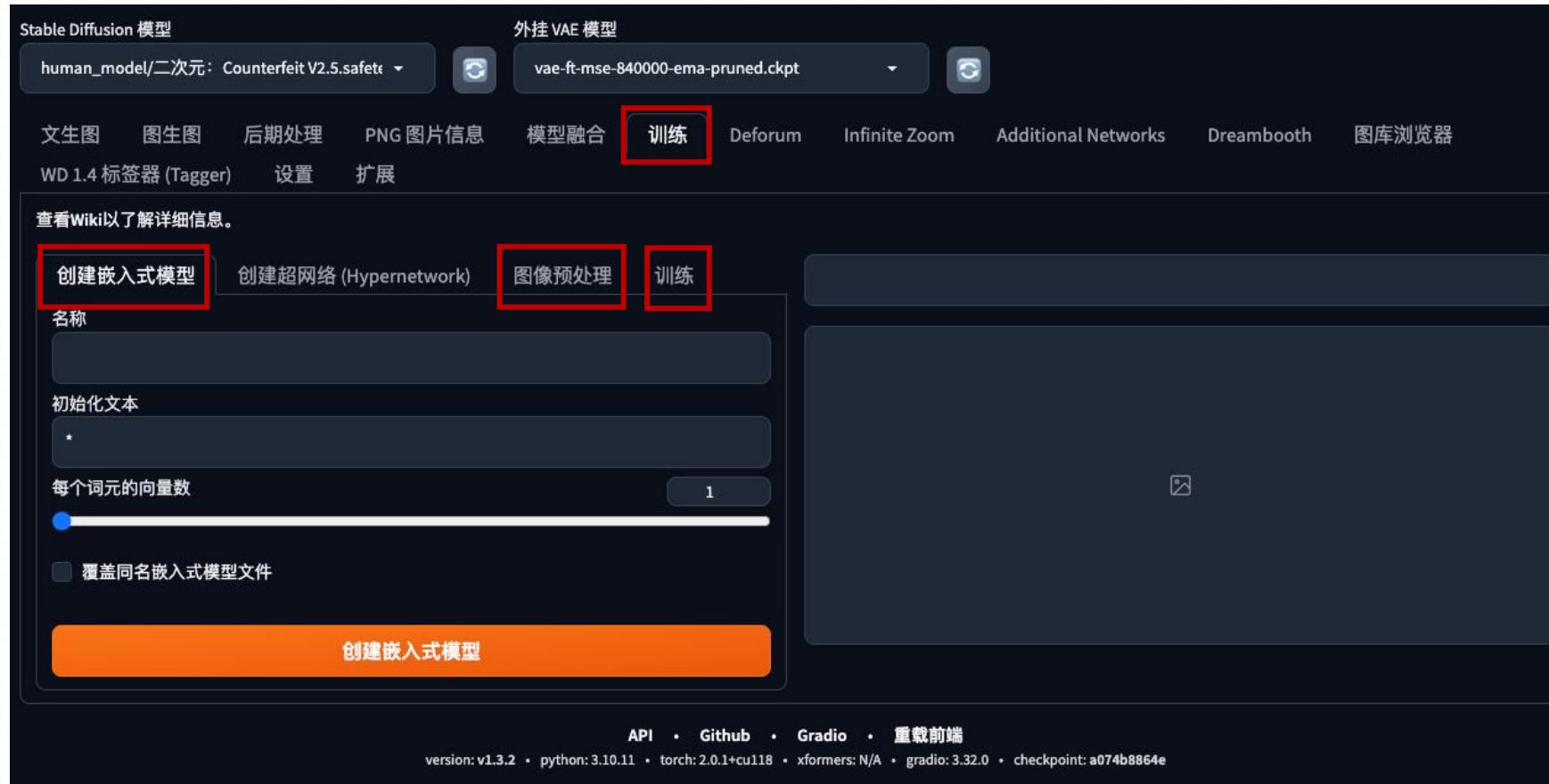


训练和监控击上面橙色的
Train，就可以开始训练了

查看日志/WebUI 的 Output 界
面查看下训练过程中的 loss,
以及模型生成出来的图片

7. 模型训练

Embedding



进入webui的
Textual Inversion
训练部分

7. 模型训练

Embedding—创建embedding



第一块是create a new embedding，用来创建自己的词条的

Name可以随意填写(之后生成图片的时候就是用这个name当作tag)

Initialization text是初始化这个词条的描述，Initialization text是训练的起点，比如要训练某个动漫角色，这里就是对此角色的描述 blonde hair,green eyes,long hair 如果用其他的角色或者画风，这里就换成那个角色或者画风的描述。

numbers of vectors per token，可以理解为tag中包含的信息量，这个参数与你所提供的数据集有关，建议100张以下的数据集最好不要调到3以上，可能会出现过拟合的现象(比如无法给角色换衣服或者换表情。)

7. 模型训练

Embedding—数据预处理



第二块是preprocess images，这一块是用来预处理数据的，因为它的模型输入只能是512*512的图片

图像预处理部分与前面的LoRA模型相似

7. 模型训练

Embedding—训练

第三块是Training，训练模块

- Embedding选项：拉下选项框选择上面创建的embedding模型就可以了
- learning rate建议使用默认的参数(如果觉得收敛太慢的话可能可以调高一些)
- Dataset Directory直接复制上文的Destination Directory就可以了。
- Log directory是你训练log的输出文件夹，这里使用默认的就可以，之后可以进入stable-diffusion-webui/textual_inversion文件夹看log文件



7. 模型训练

Embedding—训练

第三块是Training，训练模块

- Prompt template file是你输入给模型的模板文件，这里建议训练角色的时候使用subject_filewords.txt，训练画风的时候使用style_filewords.txt
- Max steps是你训练的步数，这里建议设置在10000(算力不够的话至少设置在4000)。
- 最后两个选项是你每训练多少步输出一次log和参数，这里设置成默认的就行。如果设置步数太高，导致出现了过拟合现象，你可以到textual_inversion文件夹里找到之前步数的pt文件替换就可以了





智能大数据可视化实验室

INTELLIGENT BIG DATA VISUALIZATION LAB