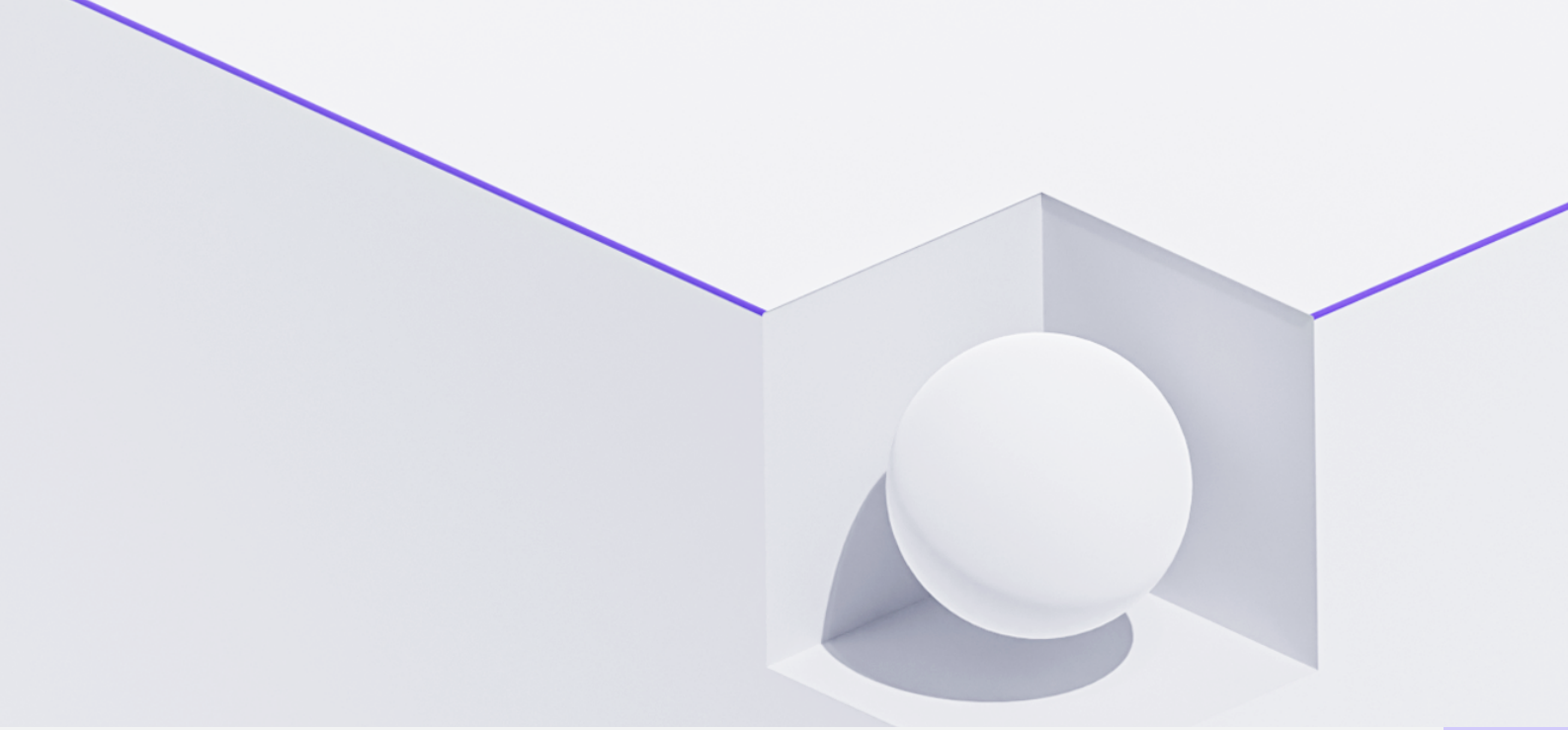


ArtLab

基于 Kohya 的 LoRA 模型训练实践

(2024)

Practice of Training LoRA Model
Based on Kohya

PAI ArtLab LoRA模型训练实践

一、基础介绍

1.Kohya工具基础介绍

首先大家对Stable Diffusion应该比较了解了。Stable Diffusion是深度学习文生图的一种模型有不同版，相对Mid journey来说它的优势是开源的。Stable Diffusion WebUI就是Stable Diffusion的一个“可视化浏览器操作界面”。在Stable Diffusion WebUI里集成了丰富的功能可以让我们通过界面进行文生图、图生图等操作；还可以通过安装各类插件以及导入多种模型等等来满足我们更高程度的定制化的绘图需求，就可以生成一个较为可控的结果。

Kohya，是当前应用比较广泛的，训练LoRA模型的开源服务。Kohya's GUI提供了可以用于模型训练的“界面”，它是一个程序包，整合了训练需要用到的环境，提供用户界面。所有东西都是在它自己的这个环境里运行，不会干扰其他的程序。比如在SDWebUI里其实也可以用一些扩展插件去做模型训练的，但是都在SD里操作势必有时候会互相干扰，产生报错，不太方便。

通过Stable Diffusion WebUI服务生成图像，需要使用多种模型，模型的能力直接决定生成图片的效果。而不同的模型具有不同的侧重点，需要使用不同的训练素材、不同的训练方式来得到。其中LoRA是一种轻量化的模型微调训练方法，在原大模型的基础上对模型微调，生成特定的角色或画风。LoRA模型训练方式速度较快，模型文件大小适中，对训练的配置要求较低。**本篇文档会重点介绍，如何训练LoRA模型**。其他模型微调方法可以参考文档——模型介绍

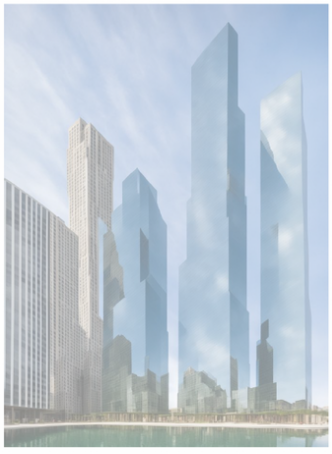
2.LoRA微调模型的意义

如果说底模（例如：Stable Diffusion v1.5 Model/Stable Diffusion v2.1 Model/Stable Diffusion XL base 1.0 Model等）是一包基础原生态的食材，那LoRA模型就是基于这些食材研发的某一种特殊风味的调料包。**为了让自己料理具备多样性和创新性，我们不能总是依赖同一包基础原生态食材**。这种调料包，可以打破基础原生态食材在料理过程中的局限性。**可以更简单更好的被人所用，去做出更具独特风味的料理。**

> 例：Stable Diffusion v1.5 模型局限性：
>
> 1. 生成内容细节准确性问题：在生成特定细节和复杂内容时，可能无法精确重现，导致生成的图像缺乏细节或不够逼真。
> 2. 生成内容逻辑性问题：生成的图像在物体排列、比例或光影效果可能不合逻辑。
> 3. 生成内容和风格一致性问题：生成图像的复杂性和和随机性很强，生成结果不可控。在执行风格迁移或特定风格的图像生成时，可能会难以保持原风格特征的一致性。

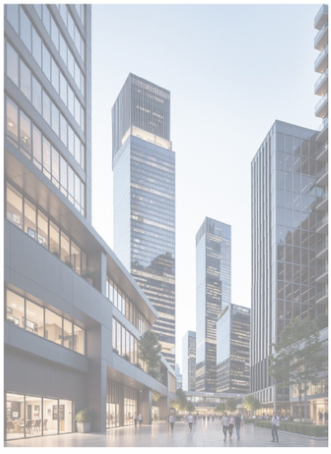
目前模型生态社区已经有很多非常优秀的微调模型，都基于底模微调训练产生。生成图像的效果，相对于基础底模来说，细节更丰富，风格特征更明显，生成内容更可控。

> 例：Stable Diffusion v1.5 模型和微调模型生图效果对比：



sd 1.5

chilloutmix



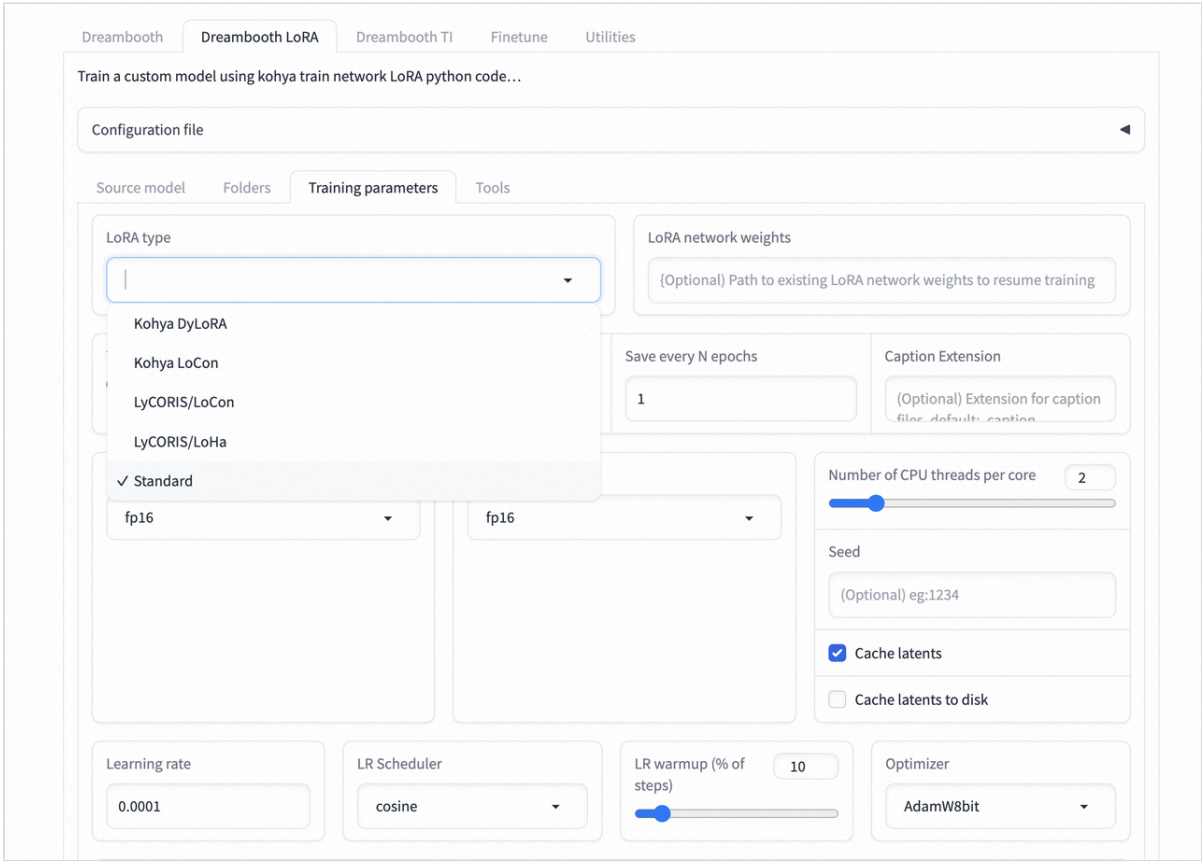
laowang_ARCH_MIX_V0.5

3.LoRA模型的基础介绍

- LoRA（Low-Rank Adaptation of Large Language Models），可以基于底模通过对数据集的训练，得到一个风格化的模型，实现更定制化的生图效果。

- **文件大小**：通常个位到百级MB，大小取决于训练时赋予的参数和基于的底模类型
- **文件格式**：通常以.safetensors后缀
- **文件应用**：需要搭配checkpoint底模使用
- **文件版本**：需要区分Stable Diffusion v1.5版和Stable Diffusion XL版本，不同版本之间不通用

- 在用kohya做LoRA模型训练时，我们可以看到在LoRA type选项区域有很多种不同的LoRA模型类型，我们这里就**默认选择standard类型即可**。那么可能会有同学好奇，**不同的LoRA模型类型之间区别是如何的**，详情可以查阅下方的文档。



4.LoRA模型的不同类型（微调技术&分类）

4.1.LyCORIS（LoHa/LoCon的前身)

- LyCORIS是LORA的增强，可以对26层神经网络进行微调，LORA可以对17层神经网络进行微调。LyCORIS会比LORA表现力更强，相较于LORA有更多的参数，可以承载更多的信息量。LyCORIS包括两个独立的改进也就是LoHa和LoCon。

（**LoCon**用于调整SD模型里的每一层，**LoHa**是在原来基础之上*2的数据信息。）

- “基础用法和lora一样，进阶用法可以调整三个参数
  - 文本编码的权重
  - Unet的权重
  - DyLoRa的权重”

如果应用常规的LoRA

语法结构：
NOFACE:1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1

示例：
<lora:hypolylora:0.6:NOFACE>

如果应用LyCORIS的LoRA

语法结构：
LyNOFACE:1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,0,0,0,0,1,1,1,1,1,1

示例：（仅指定权重）
<lora:hypolylora:1:LyNOFACE>

示例：（单独指定文本编码器、unet和dylora的权重）：
<lora:hypolylora:1:1:0:LyNOFACE>

※LyCORIS版的hypolylora暂时没写。（为了使示例易于理解而统一名称）

文献图片来自知乎-董步云

- “第1层是base层，第2～13层是IN输入层，14层中间层，剩下12层是OUT输出层。你可以使用XYZ plot功能查看对比每一层的作用”

示例：

与 XY 图一起使用，可以检查层次结构的每个级别的影响。

设置值如下。

```
NOT:0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
ALL:1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
INS:1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0
IND:1,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0
INALL:1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0
MIDD:1,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0
OUTD:1,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0
OUTS:1,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1
OUTALL:1,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1
```

文献图片来自知乎-董步云

引用一些博主的帖子给大家参考：

关于LoRA、LyCORIS和LoCon的类型说明、使用方法和插件神器（结尾提供改参数Excel表格）

https://zhuanlan.zhihu.com/p/626260866

Lycoris画风模型权重优化

https://www.bilibili.com/video/BV18c411J7Ms/?vd_source=f2b9f05be114193fbf8dcf2ba6a77988

Stable Diffusion终于会画手了：LyCORIS画手模型

https://zhuanlan.zhihu.com/p/631659806

超越LoRa，Stable Diffusion革命性的微调模型：LyCORIS

https://zhuanlan.zhihu.com/p/631370055

Stable Diffusion比LORA更好用的微调模型-LyCORIS

https://juejin.cn/post/7251786165392867388

LyCORIS：更自由的LoRA变体

https://www.youtube.com/watch?v=qd7lAvF4owM

LyCORIS - Lora beYond Conventional methods, Other Rank adaptation Implementations for Stable diffusion.

https://github.com/KohakuBlueleaf/LyCORIS

stable diffusion Lora的升级版，LyCORIS使用教程

https://www.bilibili.com/read/cv23595878/

> LoRA（Line of Action Representation）的使用方法和种类，以及如何在 LyCORIS 中使用 LoRA
> https://zhuanlan.zhihu.com/p/626260866
> Comparing LoRA Types in the Kohya_ss GUI
> https://ashejunius.com/comparing-lora-types-in-the-kohya-ss-gui-bfc8ccda7cf1
> https://openreview.net/pdf?id=d71n4ftoCBy

- ## 4.2.LoCon

LoCon (Conventional LoRA): LoRA 只调整了 cross-attention layers，LoCon 还用同样的方法调整了 ResNet 矩阵。更多信息参见 LoCon - LoRA for Convolution Network （目前LoCon已经被LyCORIS合并，过去扩展的LoCon已经不再需要了）

- ## 4.3.LoHa

LoHa (LoRA with Hadamard Product): 用 Hadamard Product 替换掉了原方法中的矩阵点乘，理论上在相同的 下能容纳更多（丢失更少）的信息。该方法来自论文： FedPara Low-Rank Hadamard Product For Communication-Efficient Federated Learning

> **引用一些博主的帖子给大家参考：**
> 基于LyCoRIS的LoRA模型：LoHa
> https://www.bilibili.com/video/BV1cj411A73a/?spm_id_from=333.337.search-card.all.click&vd_source=f2b9f05be114193fbf8dcf2ba6a77988
> 🌟LyCORIS loha 训练模型方法 stable diffusion
> https://www.bilibili.com/video/BV1es4y1271H/?spm_id_from=333.337.search-card.all.click&vd_source=f2b9f05be114193fbf8dcf2ba6a77988

- ## 4.4.DyLoRA

“根据论文，LoRA的rank并不是越高越好，而是需要根据模型、数据集、任务等因素来寻找合适的rank。使用DyLoRA，可以同时在指定的维度(rank)下学习多种rank的LoRA，从而省去了寻找最佳rank的麻烦”

> 引用一些博主的帖子给大家参考：
>
> DyLoRA: Parameter Efficient Tuning of Pre-trained Models using Dynamic Search-Free Low-Rank Adaptation
>
> https://arxiv.org/abs/2210.07558
>
> 使用动态无搜索低秩适应的预训练模型的参数有效微调
>
> https://zhuanlan.zhihu.com/p/623455592
>
> 🌟Stable Diffusion LoRA 训练不完全指北
>
> https://www.bilibili.com/read/cv23072377/
>
> 关于LoRA的学习
>
> https://github.com/kohya-ss/sd-scripts/blob/main/docs/train_network_README-zh.md

二、LoRA模型训练

1.数据集准备

- ### 1.1.确定LoRA模型类型
  - 首先需要确定，你要训练的Lora模型的类型，比如是人物角色、插画、建筑等类型。不同的模型类型，准备的数据集和对数据集的要求也不尽相同。所以先确定好类型，再去针对这一类型模型训练的去准备对应的训练数据集 .
  - 例：我这边需要训练一个阿里云进化设计语言体系下的阿里云三维产品图标的风格模型。

- **1.2.数据集内容要求**
  - 训练Lora模型，数据集是由图片和图片标注（图片对应的文本描述标注）两种文件组成的。

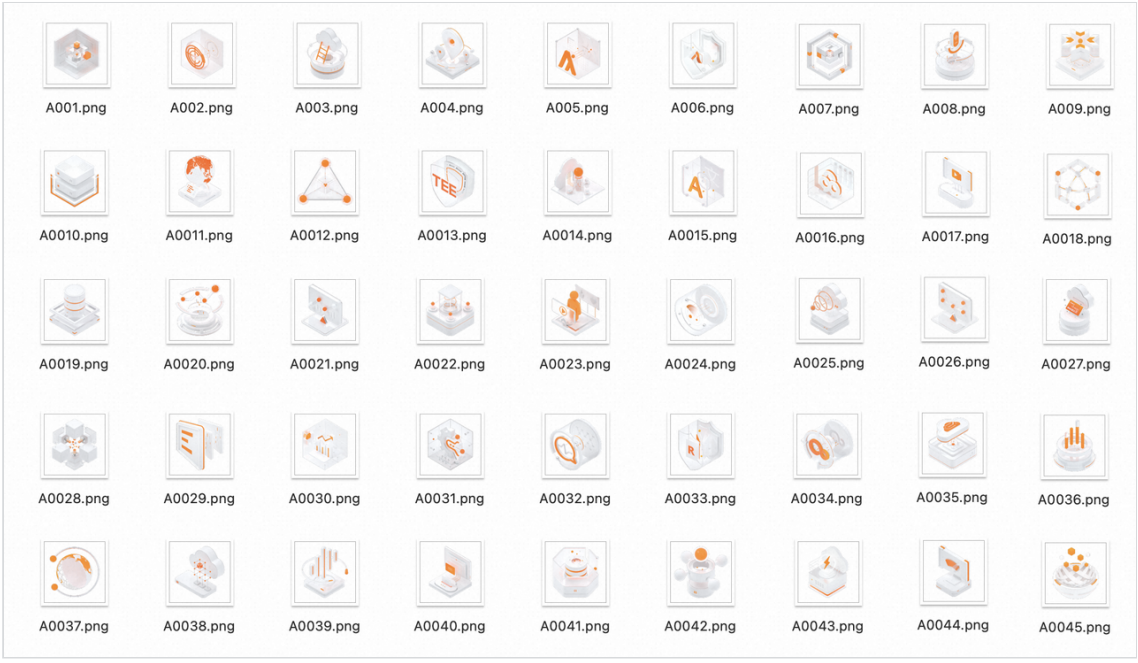
- **1.3.数据集内容准备-图片**

- **1.3.1.图片要求**
  - 对于训练LoRA风格模型，图片收集的标准大致为:
    > - 数量：15张以上（一般不小于10张）
    > - 质量：分辨率适中，画质清晰，清晰展示出你希望机器去学习的风格和特征

    - 风格：需要一套统一风格的图片内容，保障数据集的一致性（如果你要你的模型比较多样化，那数据集也可以多样性去准备，但过于迥异多样的数据集，也会导致机器学习不到位）
    - 内容：图片需凸显要训练的主体物形象，不宜有不好的角度、复杂背景以及其他无关的内容（文字、水印等）
    - 尺寸：注意分辨率在512-768之间，64的倍数，显存低可以裁剪512*512，显存高可裁768*768。

- **1.3.2.图片预处理**
    - 前面提到，对图片会有各种维度的要求，比如已有的图，对质量、尺寸怎么调整
        - 质量调整
            - 图片分辨率适中即可，保证画质清晰，但也无需太大。画质会影响模型训练的结果。如果自己准备的图片分辨率比较小，不是很清晰，可以使用SD WebUI中Extra页面中进行分辨率放大，也可以使用其他图像处理工具去处理图像的分辨率。
        - 尺寸调整量调整
        - 可以前往像birme这种站点批量裁剪后批量下载，也可以使用SDWEBUI裁剪或手动裁剪。

- **1.3.3.图片部分准备完毕示例1-三维产品图标**
    - 以下就还算一套相对比较符合要求的数据集图片内容了

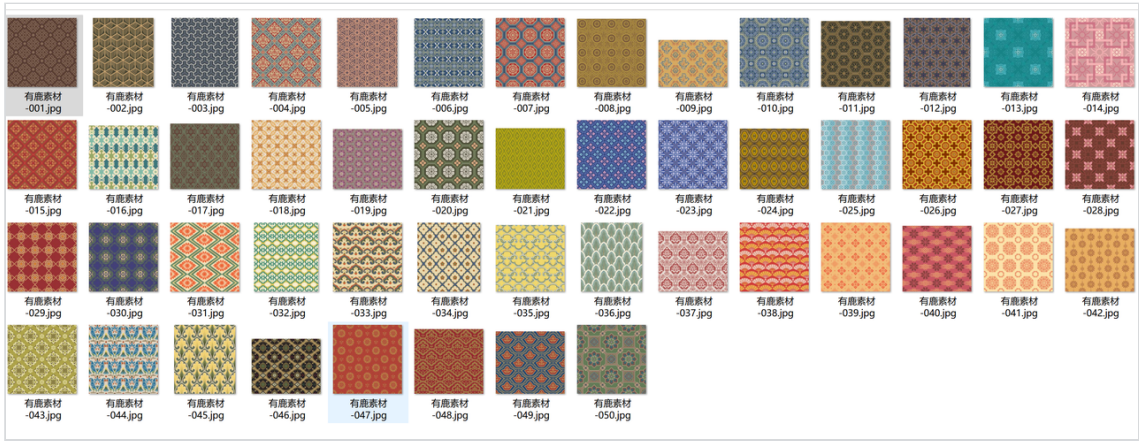


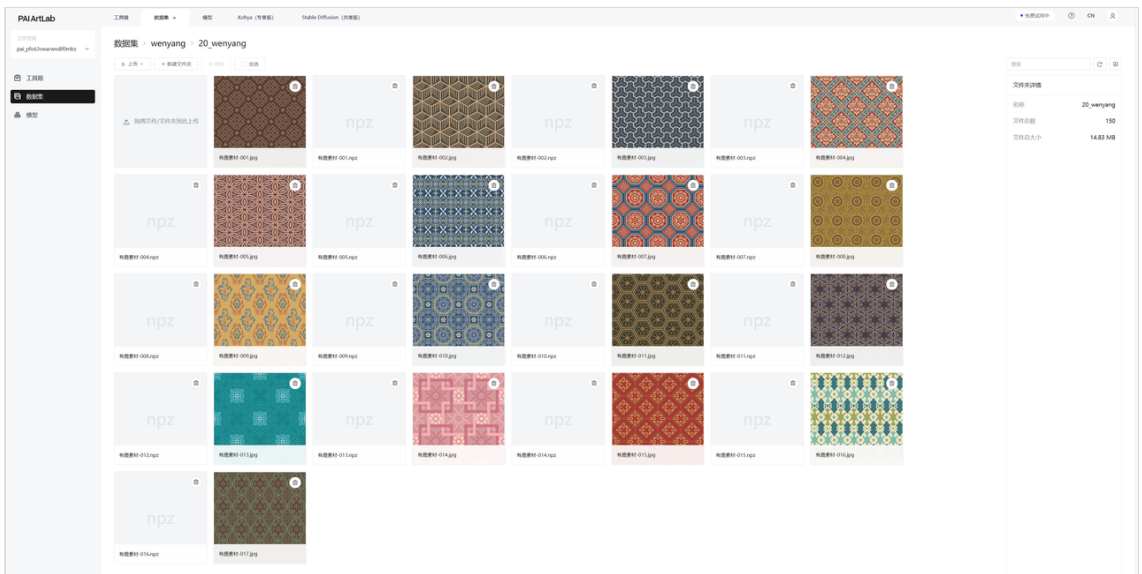
    - 我习惯性会把他们整理在我本地的文件夹里，并且按照这个文件夹命名规则来命

名，“数字+下划线+任意英文名称”，对于为什么这样命名，稍后会讲解到。

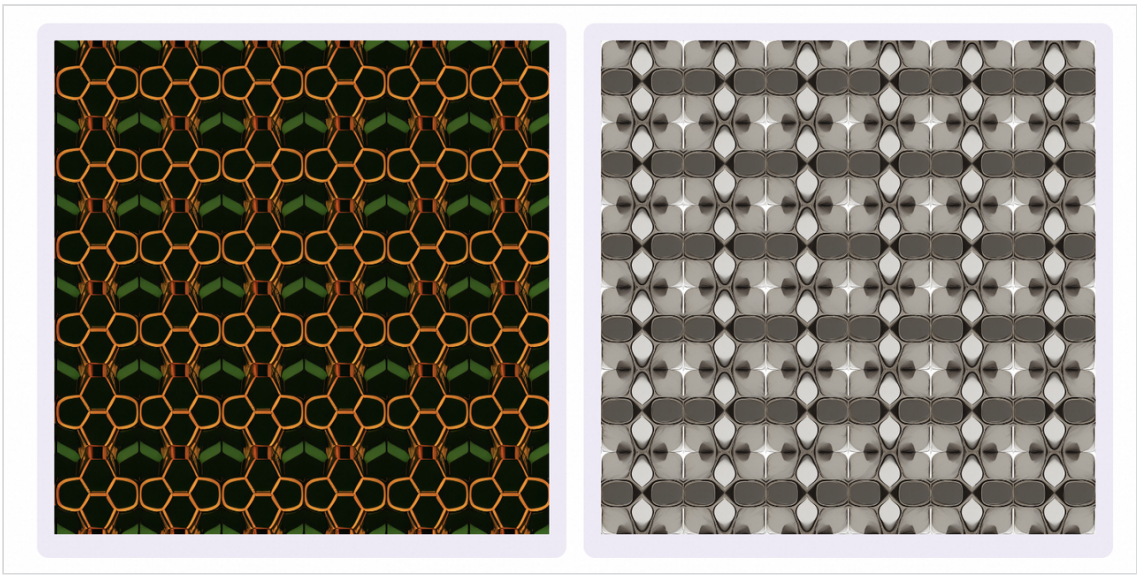
- 

- **1.3.4.图片部分准备完毕示例2-传统服饰纹样**
  - 接下来我们再来看看这一套数据集，图像多样性过于丰富，多样性大的情况下每一种类型的数据集图像数量也较少，图像内容过于抽象，同时图像的打标我们不够细致。通过实践，这样的数据集图像训练效果不太理想。

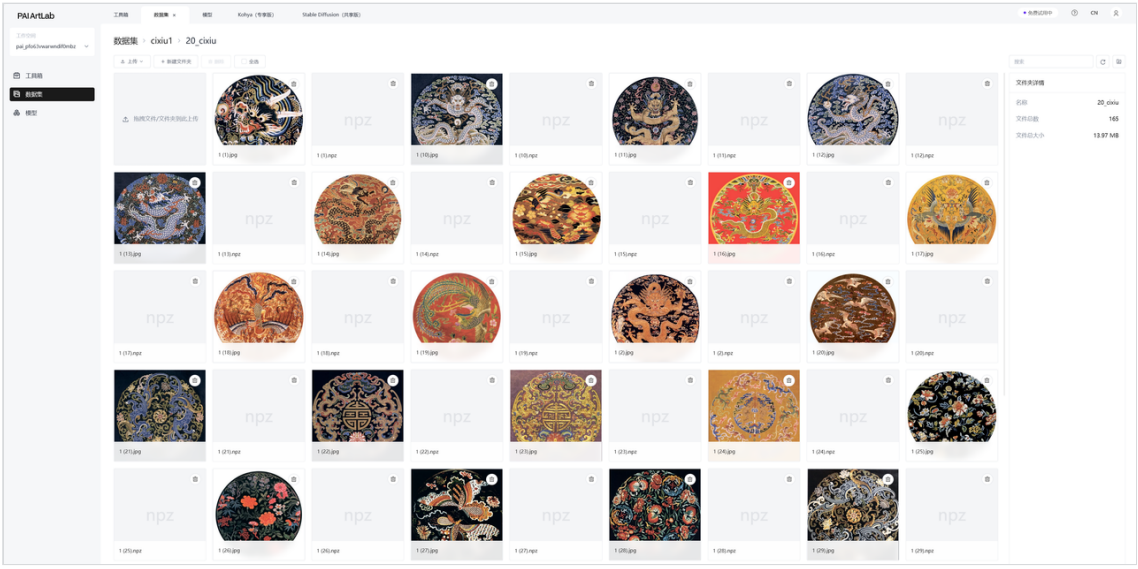




  - 通过训练，模型生成图像的效果如下（不理想）



- 再来看一下这套数据集的图像部分再结合比较充分的打标，通过实践，这样的数据集图片训练效果较好。



- 通过第一轮比较仓促的训练，模型生成图像的效果如下（还需要继续微调迭代，才会达到内心的理想效果）

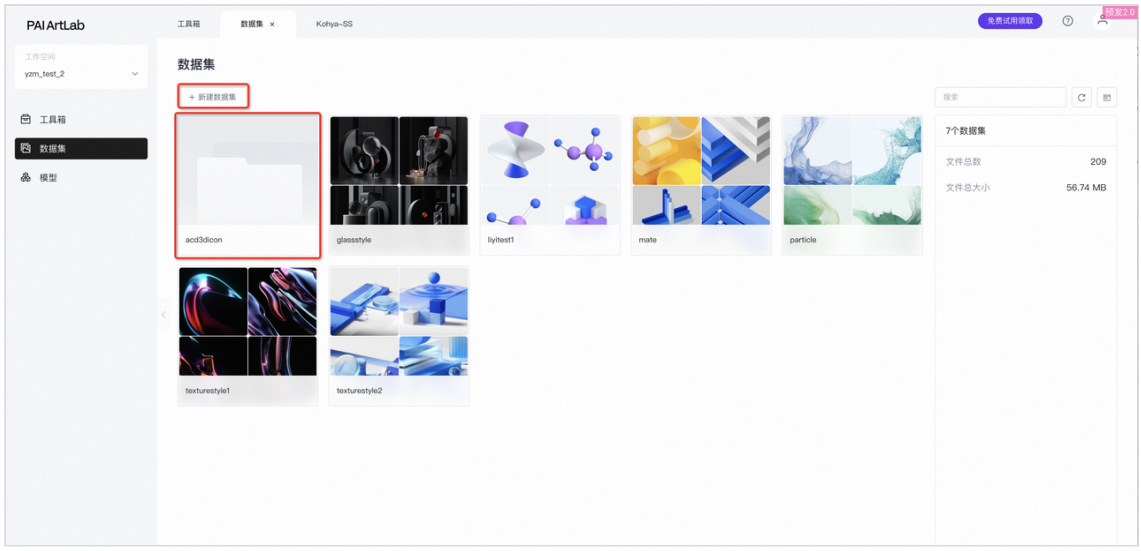
1.4.ArtLab数据集的创建和图片文件的上传

1.4.1.上传前，要注意文件的属性和命名要求

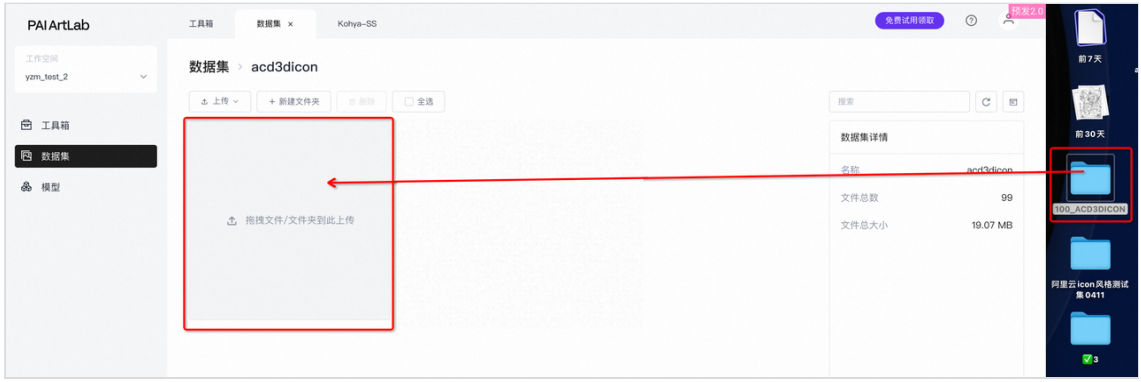
- 如果只是用平台管理数据集文件，或者给图片打标，那直接上传文件或文件夹都可以，对这些文件和文件夹的命名也没有特殊要求
- 因为接下来，我们这个数据集打标完之后，是需要用平台的kohya做lora模型训练的，那对于上传的文件属性和命名是有要求的
- 首先，必须上传文件夹
- 其次，命名要求如下
  - **命名格式：数字+下划线+任意名称**
  - 命名含义：
  - 数字：训练总步数 = 图片数量 * repeat 数量 * 设置的epoch / batch_size，总训练次数一般要求>1500，因此若文件夹内包含15张图片，则每张图片训练1500/15=100次，图片文件夹名数字部分可为“100”。
  - 下划线：**必须有**
  - 任意名称：本例中为“100_ACD3DICON”，大家可自行命名

1.4.2.创建数据集

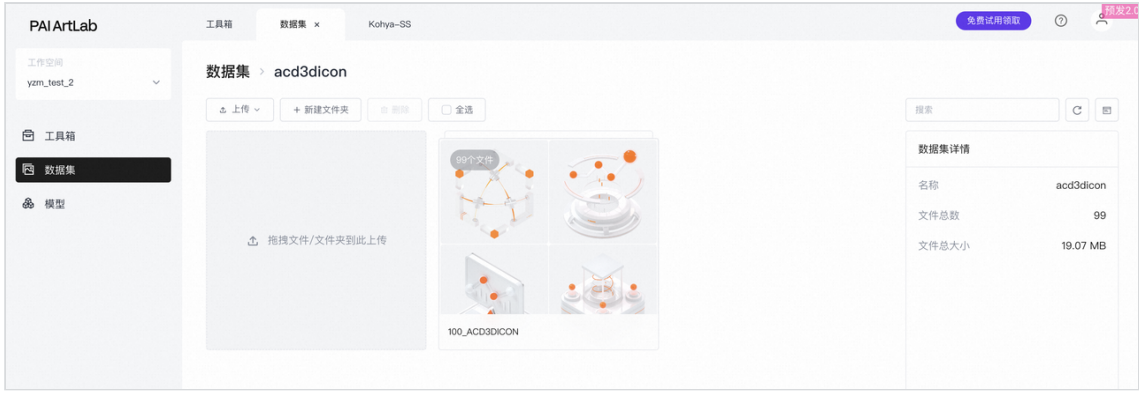
- 来到平台数据集，点击新建一个空的数据集，我这边建好了，叫做acd3dicon



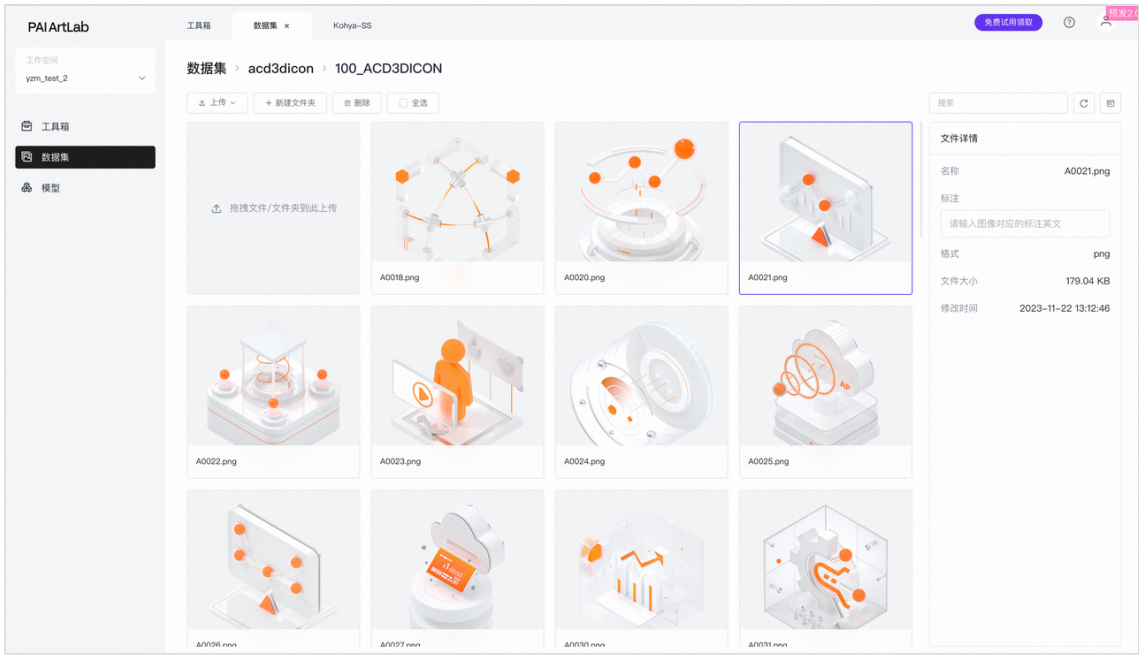
- **1.4.3.上传数据集文件**
  - 点击进入到刚刚创建好的叫做acd3dicon的数据集内，将自己整理好的数据集图片文件夹从本地拖拽上传



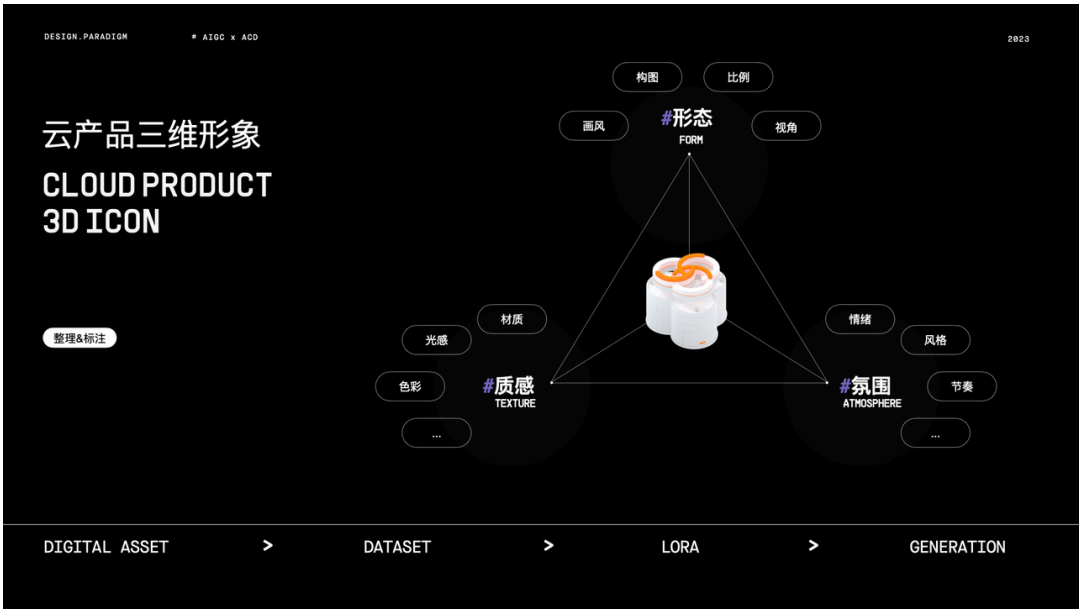
  - 上传好了



  - 进入到文件夹里可以查看到我们的图片



- **1.5.数据集内容准备-图片标注**
- **1.5.1.什么是图片标注**
  - 这里图片标注，是指每张图片对应的文字的描述。文字描述的标注文件，是与图片同名的txt格式的文件。
- **1.5.2.图片标注的要求**
  - B端元素一般有明确的结构、规范的透视、特殊规范的光影，所以打标时，跟处理人像、景观等这类数据集图片会有些差别。建议我们在这里只做基础的描绘打标，比如对这个纬图标的顶层、中层和底部的一些基础形态做描述：球体、正方体等。
    - 例：针对这类“3D图标”画面打标的信息维度的拆分



	分类	关键词
业务	产品/业务	数据库、云安全、计算平台、容器、云原生等
	云计算元素	Data processing , Storage , Computing , Cloud cor Elastic computing , Distributed storage , Cloud dat Virtualization , Containerization , Cloud security , C architecture , Cloud services , Server , Load balanc Automated management , Scalability , Disaster rec availability , Cloud monitoring , Cloud billing
设计（质感）	环境&构图	viewfinder，isometric，hdri envriroment，white ba negative space
	材质	glossy texture , matte texture , metallic texture , gla frosted glass texture
	照明	studio lighting , soft lighting
	色彩	alibaba cloud orange , white , black , gradient oran transparent ， silver
	情绪	rational , orderly , energetic , vibrant
	质量	UHD，ccurate，high details，best quality，1080P

设计（氛围）	...	...

- 例：针对紫砂壶这个物品打标的信息维度的拆分

- **1.5.3.如何给图片添加标注**
  - 首先我们可以手动为每个图片添加对应的文字描述，但当图片量非常大的情况下，手动打标是非常耗时耗精力的。这时我们可以选择借助神经网络，来为我们完成对所有图片批量生成文本描述的工作。
  - 我们可以在Kohya中选择使用一个叫做BLIP的图像打标模型。虽然目前图像打标的模型，自动识别图片生成的文本描述暂时不那么完美，不完美的情况，后期我

们可以进行手动微调，也可以满足大部分的需求。

1.6.数据集批量打标方法

1.6.1.选择类别

- 选择Utilities、Captioning、BLIP Captioning

1.6.2.找到路径

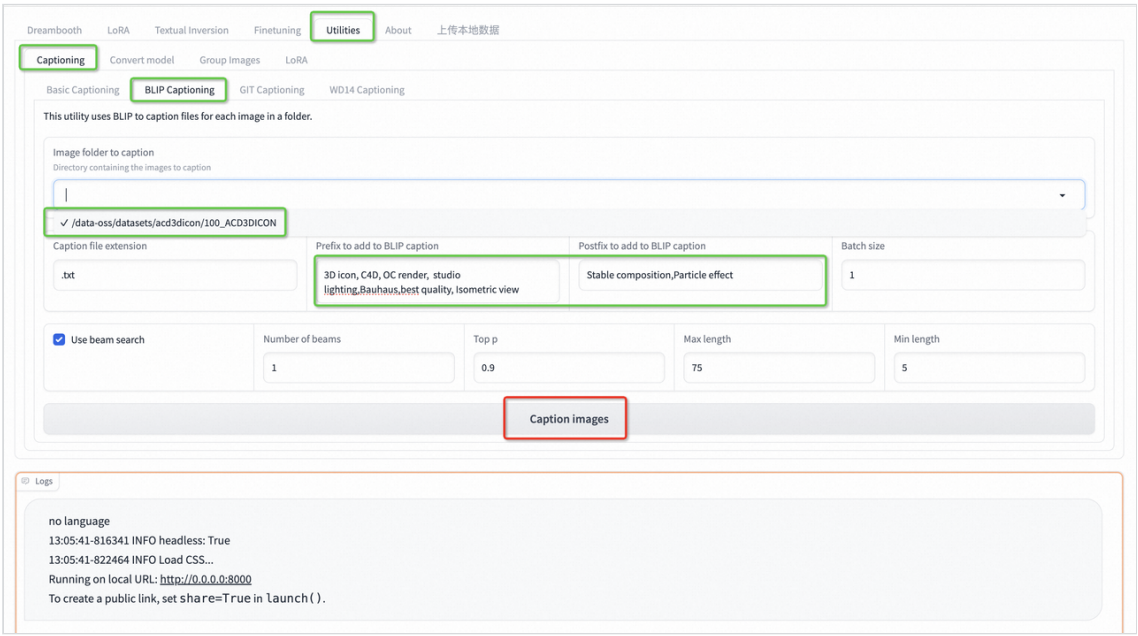
- 点击下拉箭头，找到我们在创建的数据集里面上传的图片文件夹，告诉机器接下来要给这个文件夹里的图片批量打标

1.6.3.输入预置词

- 输入一些预置词，这是让机器的给每一张图片都批量加上这里你输入的标注文本。这里我先随便输入了一些，大家可以结合自己对数据集图片拆分的维度去添加预置词，不同类型的图片打标的维度也不同

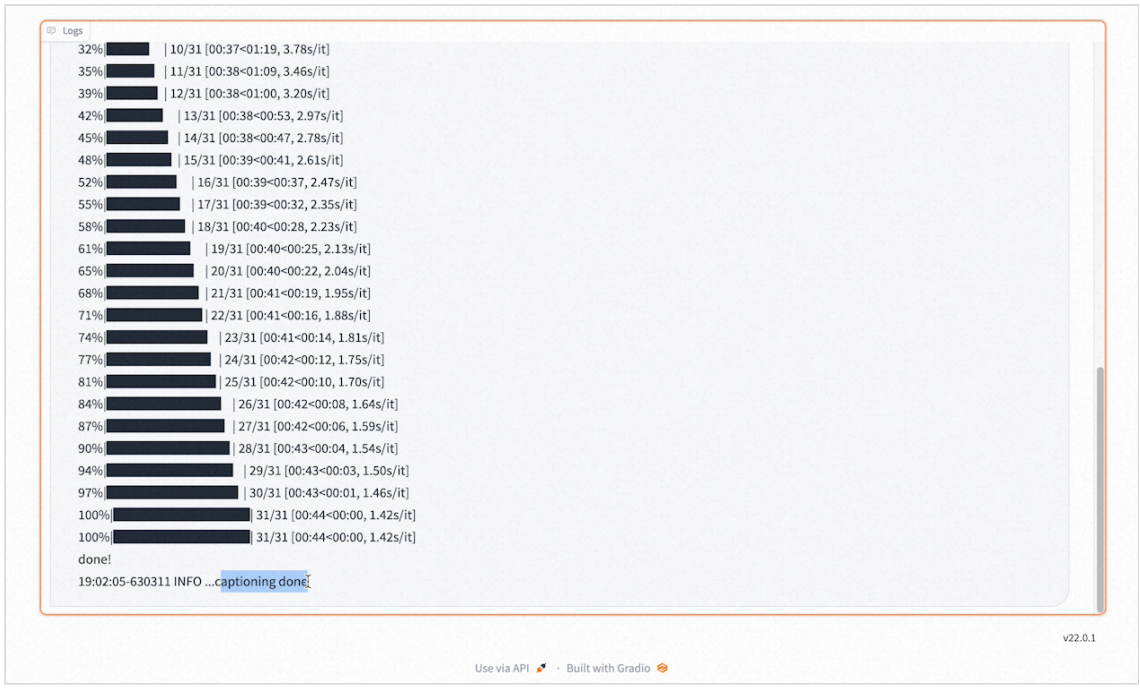
1.6.4.点击打标按钮

- 点击Caption Image即可开始打标

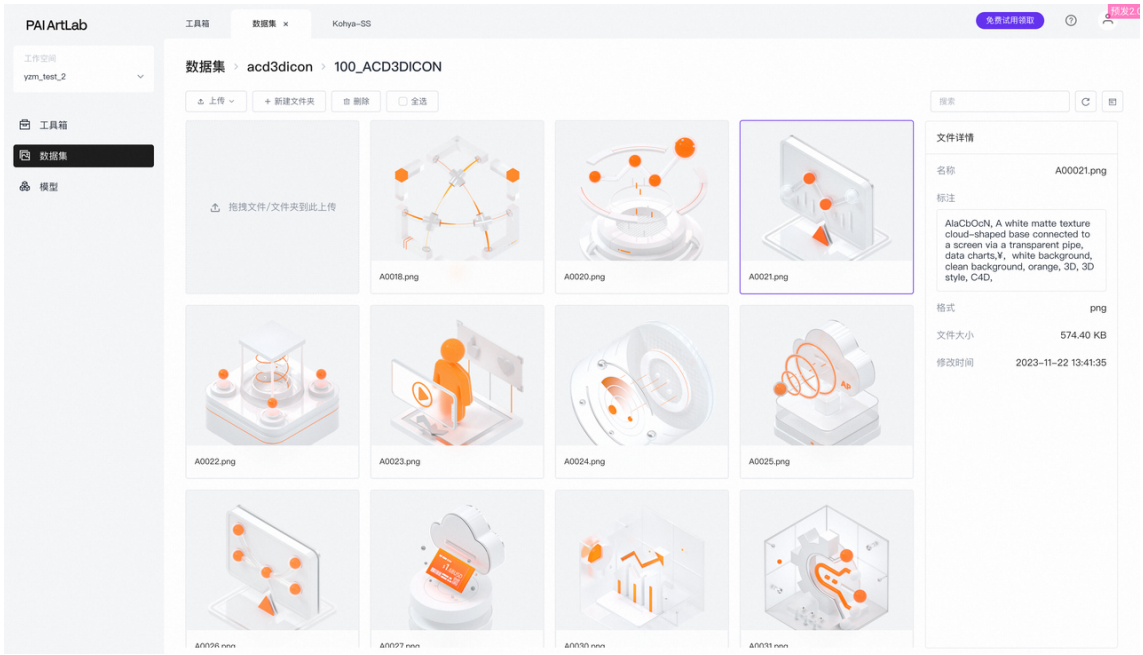


1.6.5.观测打标状态

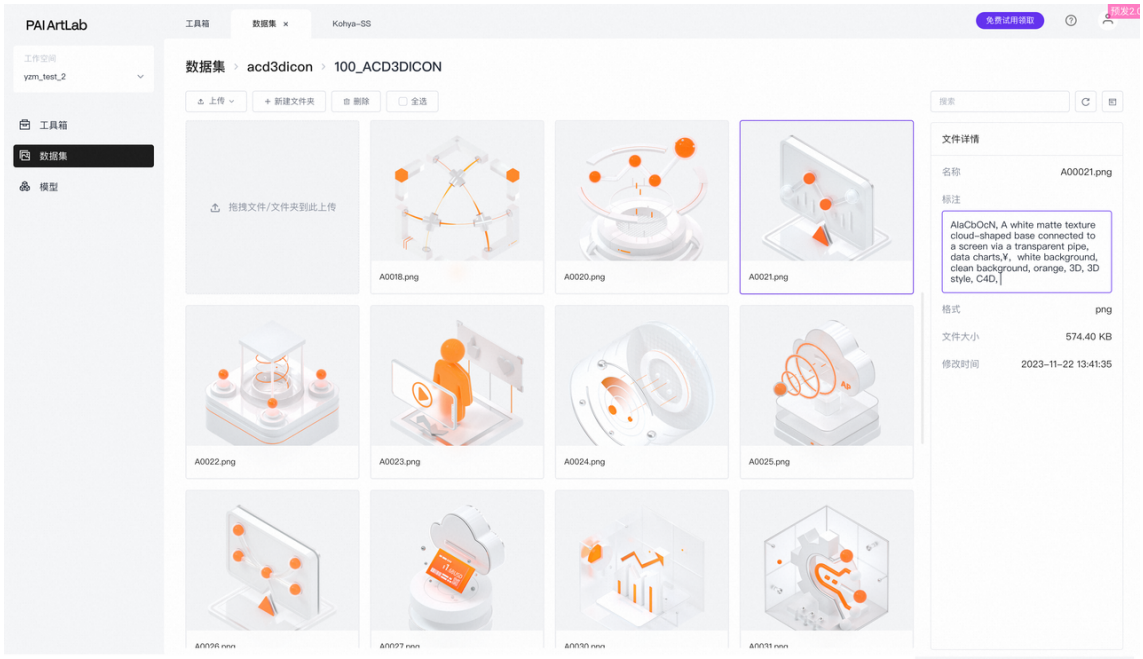
- 在最下方的日志里可以观察到自己打标的进度，也可以看到打标完成的提示。



- **1.6.6.返回数据集功能模块检查图像标注**
  - 这时候再返回到数据集里面，可以看到刚才上传好的图片已经有对应的标注文件了。

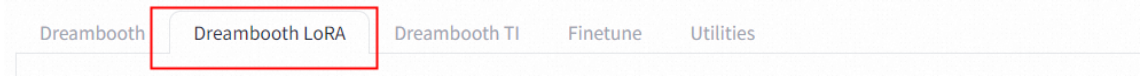


- **1.6.7.数据集功能模块也可以手动调整图像标注**
  - 大家如果想要微调标注可以去借助其他打标插件，也可以通过平台手动微调。

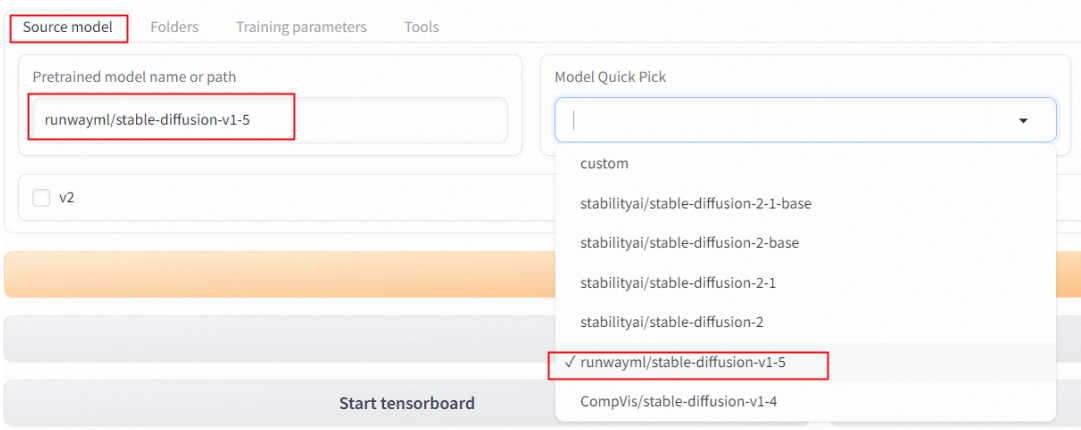


2.Kohya实操：LoRA模型训练

- **2.1.先选择底模**
  - 底模型是LoRA训练时采用的基础模型，建议采用runwayml/stable-diffusion-v1-5（这个模型去huggingface下载，名字是v1-5-pruned-emaolny.safetensors类似这样的名字）或者用sd_xl_base_1.0.safetensors等别人验证过的模型。如果你网上随意下载一个checkpoint大模型就拿来做底模型大概率会报错。

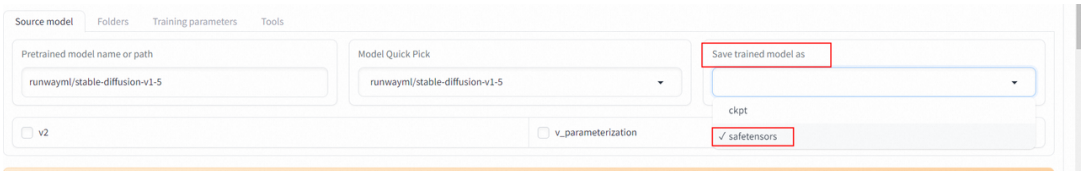


  - SourceModel页面操作
    - 选择训练模型



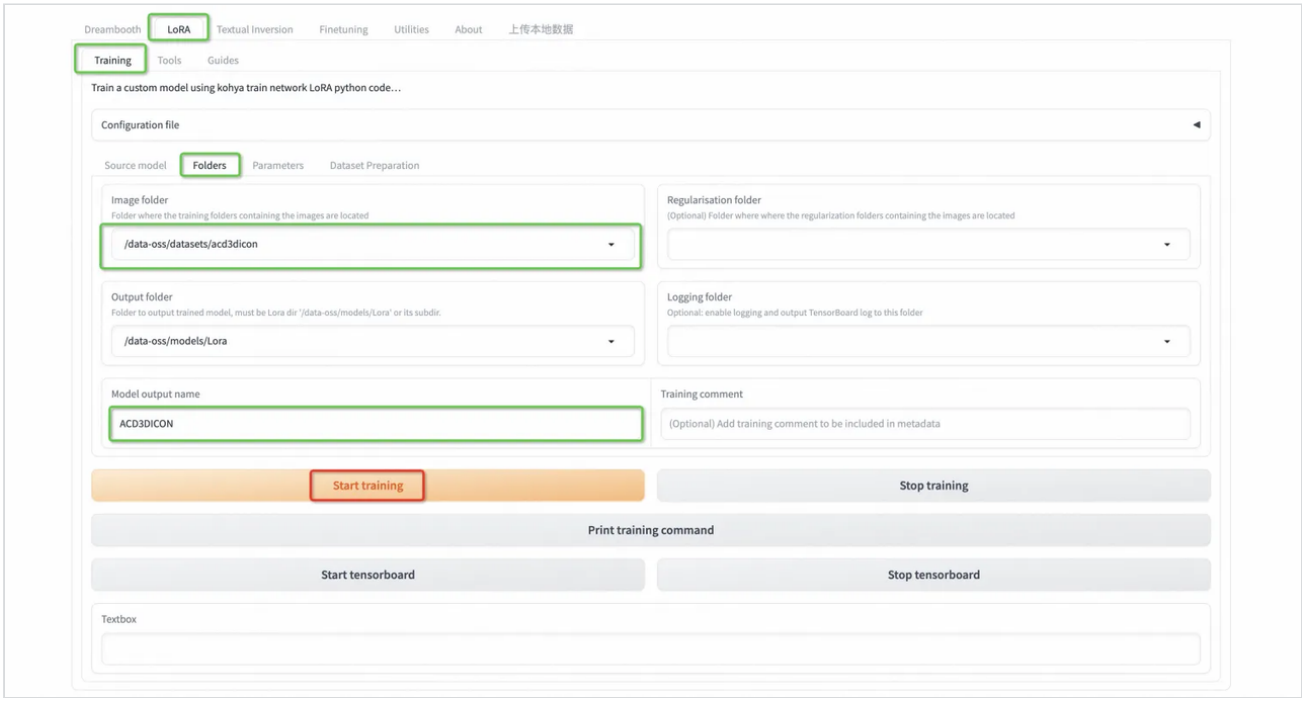
进入“Source Model（源模型)”并选择预训练的模型

- 选择模型类型



本例中选用safetensors，相比较checkpoint来说更具有安全性

- **2.2.选择模型类型和数据集**

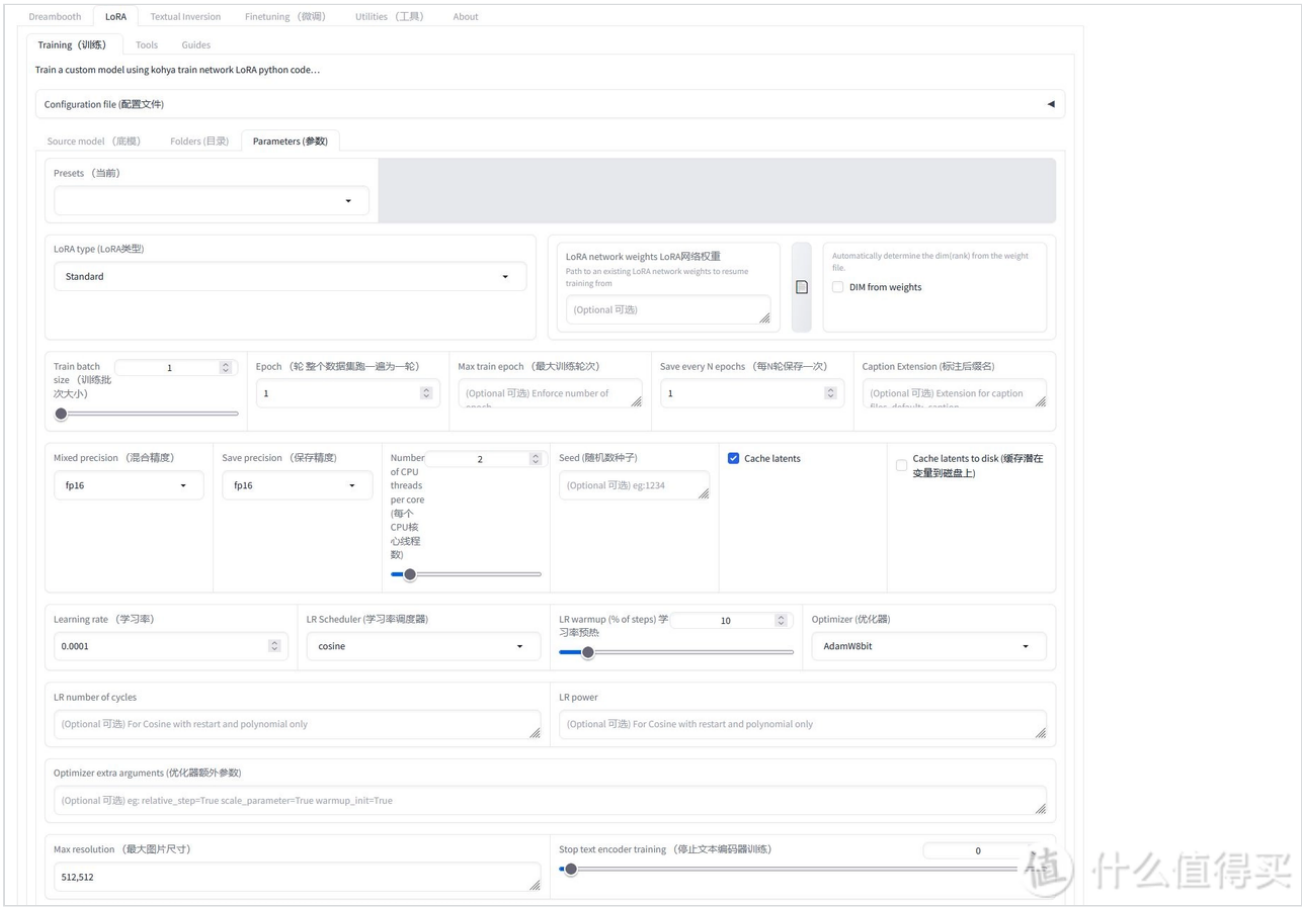


  - 2.2.1.选择Lora、Training、Folders
  - 2.2.2.选择我们上传了数据集文件夹的数据集（这里注意，数据集文件打标的时候，我们要选到数据集下面图片的文件夹；做模型训练的时候，我们要选择放置数据集文件夹的数据集）
  - 2.2.3.给一些训练参数, 在这个环节不展开讲解（在教程末端可查阅）
  - 2.2.4.点击Start training

- **2.3.训练进度与模型存储**

  - 我们依旧在页面最下端的日志里可以看到模型训练的进度，也可以看到模型训练完成的提示

3.Training Parameters 部分常用训练参数介绍说明



- **3.1.参数介绍**
  - 首先模型训练总步数：
    - 图片数量 * repeat 数量 * 设置的epoch / batch_size = 训练总步
      - eg.10张图*20步*10个循环/2并行数=1000步
  - 标紫色是比较常调的参数

参数	功能	设置说明
repeat	这个在文件夹命名时已经赋予了，代表会读取多少次这个图像	文件夹命名
LoRA Type	选择LoRA类型，这里大家直接保持默认选择standard即可。	这里不同的LoRA类型在文档最上方有讲解
LoRA network weights	LoRA网络权重，如果要接着训练则选用最后训练的LoRA	选填
batch size	训练批量大小	根据显卡性能，12G显存最大为1,8G最大为2
epoch	训练轮数——将所有数据训练一次为一轮	自行计算。一般： Kohya中总训练次数=训练图片数量x重复次数x训练轮数/训练批量大小。 WebUI中总训练次数=训练图片数量x重复次数。 使用类别图像时，在Kohya或在WebUI中总训练次数都会乘2；在Kohya中模型存储次数会减半。

深度学习，训练数据的时候，有个既简单又常见的技术点，需要我们弄清楚。

repeats（步数）： 10_文件夹名称，即文件夹里每张图学机器学10次。

batch_size（一批次的大小）：例如，批次大小为2，则同时学习2张图像，1个batch包含的样本数目通常设置为2的n次幂，68，128，256。这小部分样本即为“一批数据”

较小的批次大小：会导致训练更慢，需要更少的repeat数，机器学习会提升对图像精度的学习

较大的批次大小：会导致训练更快，需要更多的repeat数，机器学习会降低对图像精度的调整，但会捕捉多张图像的泛的特征。（这里建议大家设置较大的批次大小和repeat数）

epoch（轮次）：一个轮次就是把所有的数据集过一遍。也就是“一轮训练”

iteration（迭代次数）：每跑完一个batch都要更新参数，这个过程叫一个iteration。也就是“一次训练”一个epoch里，迭代的次数（Iterations）就是batch的数量

总结：Epoch 表示完成一次对整个数据集的训练，Batch 表示把大规模数据分成小批次进行训练，Iteration 表示每小批次进行训练更新参数的过程。

举例1

比如，总共10000张图片，batch_size的大小设置为100，训练的轮次epoch设置为10，那么这批跑完10个epoch，一共迭代了1000次

iterations = （10000/100）*10 = 1000（次）

举例2

比如，总共10000张图片，batch_size的大小设置为256

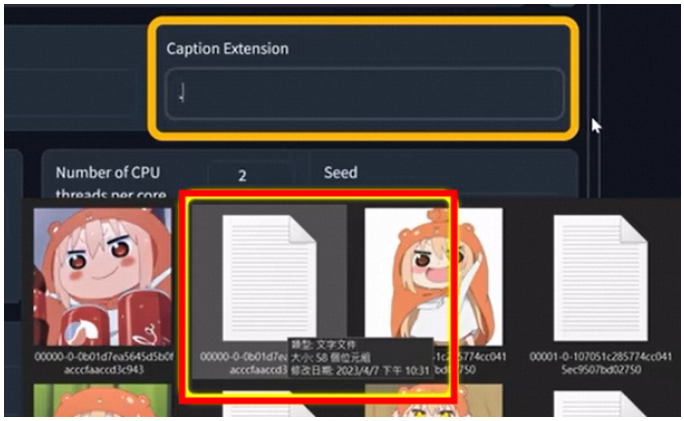
每个epoch训练的图片数量：10000

训练集的bath数：/256=39+1（40）

每个epoch需要完成的batch数：40

每个epoch的iteration数：40

训练10轮，模型权重更新数：40*10=400

Caption Extension	打标文件扩展名	选填，如图  训练图集中注解/提示文件的格式，.txt
Mixed precision	混合精度	视显卡性能决定。默认可选no、fp16、bf16。30以上显卡可选bf16
Save precision	保存精度	同上
Number of CPU threads per core	CPU每核线程数	主要为显存，根据所购实例和需求调整
LR Scheduler	学习率调度器	下拉列表，按需选择cosine或cosine with restart等函数
LR Warmup（% of steps）	学习预热步数	按需调节，默认为10，可以选不要（0）
Max Resolution	最大分辨率	根据图片情况
Network Rank (Dimension)	模型复杂度	一般默认可设64能适应大部分场景
Network Alpha	网络Alpha	建议可以设小，rank和alpha设置影响最终输出lora大小

Convolution Rank (Dimension) & Convolution Alpha	卷积度	Lora对模型的微调涵盖范围。根据LoRA Type不同做调整。 Kohya官方建议： LoCon：dim<=64,alpha=1(或更低) LoHA：dim<=32，alpha=1
clip skip	文本编码器跳过层数	Clip跳过，二次元选2，写实模型选1——动漫模型训练最初就有跳过一层，如使用训练素材也是二次元图像，再跳一层=2
Learning rate	学习率	推荐：1e-4（1除以10的4次方） 1e-4=1/10000=0.0001 1e-5=1/100000=0.00001 刚开始训练时：学习率以0.01 ~ 0.001 为宜 loss值太高提高学习率，loss值太低就降低学习率 学习率高，可能很快的跑到终点，但可能学的太快，囫囵的学，导致过拟合（过度浮夸）。 学习率低，可以更精致更精细的学，但可能学的太慢，效率太低，导致不拟合（跟数据集不像），花的时间还久。
Unet lr	学习率	推荐：1e-4 学习率高，学习速度快 设置Unet会覆盖lr值
Text encord lr	学习率	推荐：5e-5 学习率高，学习速度快 调整至Unet lr的十分之一左右，有助于学习文本编码器（对图像对应的文本描述学习更敏感）
Optimizer	优化器	按需选择，默认AdamW8bit，DAdaptation自动操作

Network Dimension	学习精细度	精细度越高，模型的细节越好 但也不是越高越好。提升维度有助于细节学习但模型收敛速度慢训练时间久，也容易过拟合，高分辨率的图需要更高的纬维度。 Network Dimension=128时 输出文件大小140MB Network Dimension=64时 输出文件大小70MB Network Dimension=32时 输出文件大小40MB 二次元场景：32（画风越华丽精度可以越高） 人物：32-128 实物风景：大于等于128（对象越复杂DIM越高）
Optimazer	优化器	不同优化器适用的模式不一样
	AdamW8bit	推荐：1e-4
	DAdaptation	推荐：1 优化器自动调整学习率。学习率选项指定的值不是学习率本身，而是D-Adaption决定的学习率的应用率，所以通常指定1.0
	Lion	采用DA决定的最优化率1/3

- **3.2.新手建议参数**

我们T4/P100的机器是16G		
分辨率	显存	batch size
512*512	12G	6
	6G	4
1024*768	24G	4

新手建议		
图片	30张	图像质量高
Repeat	二次元：7-15 人像：20-30 实物：30-100	次数多学习效果好
Epoch	10	
Unet lr	1e-4	如果初始loss下不去，提高学习率，如果太低，降低学习率
Network rank	128/64（Alpha）	128细节更好
Loss	一般在0.08左右	

- **3.3.loss值**

  - 在机器学习LoRA模型的训练过程中，损失值(loss)是一个重要的指标，它衡量的是模型预测结果与实际结果之间的差异，间接用于评估模型的训练效果。理想情况下，随着训练过程的进行，损失值应该逐渐减小，这意味着模型正在逐步学习

和适应训练数据，提高其预测准确性。 一般而言，**当LOSS值在0.08~0.1之间时，可以认为模型的训练效果较好**，LOSS值为0.08时较佳。

- 简单理解，lora学习是loss从高到低的过程，假设我epoch=30，那么如果我需要的loss区间是0.09-0.07，我可能能获得第20轮-第24轮的lora，不至于因为轮次太小，比如两轮间loss直接从0.1变成0.06，错过我想要的loss区间。

炼丹是玄学，没有一个固定的模版，还是建议大家自己多多尝试。