# 一、ComfyUI配置指南

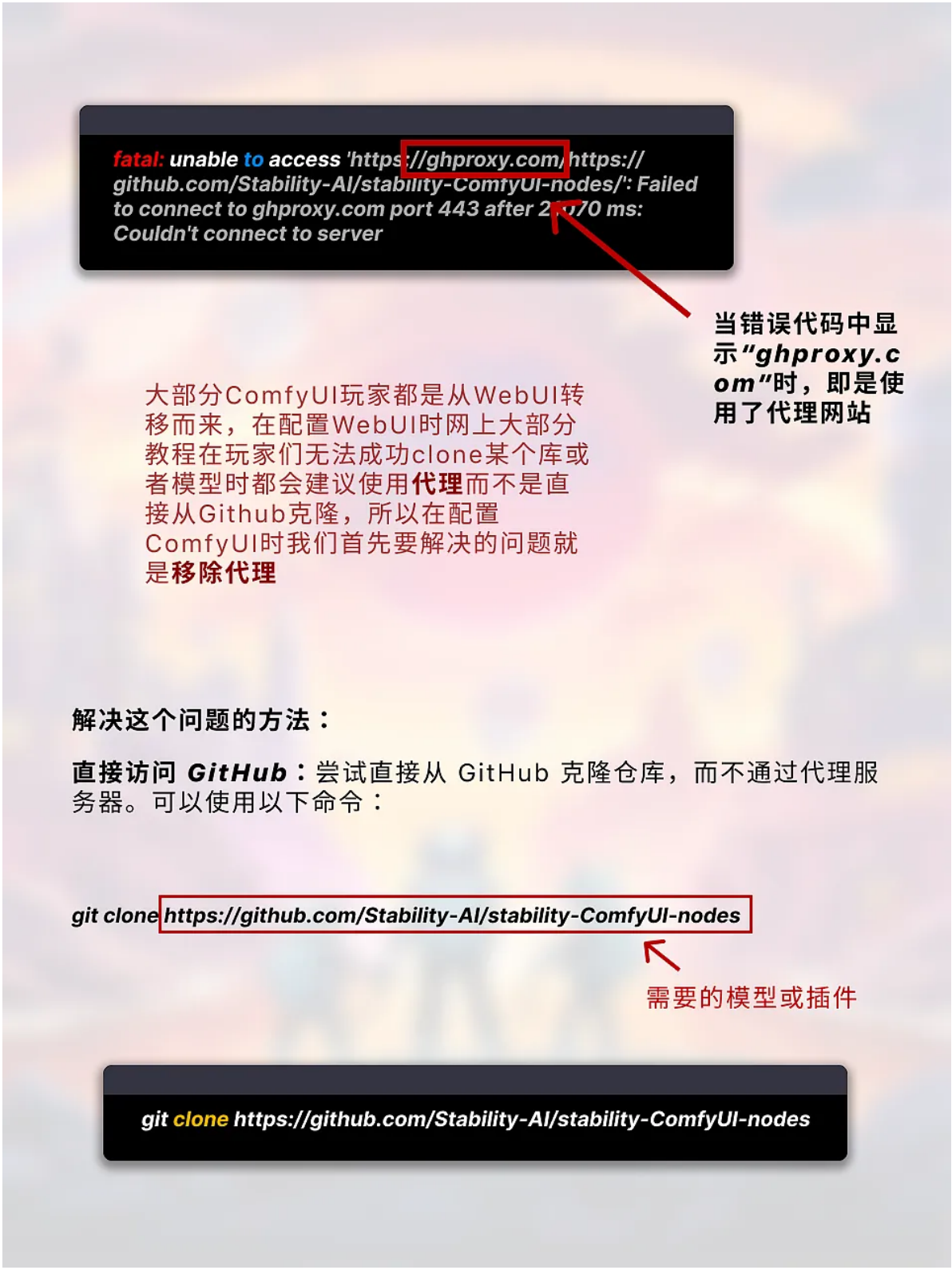
## 前言

SD是固定了底层架构的ConfyUI，像文生图，图生图都是confyui里已经固定的工作流ConfyUI利于专业团队使用，多人协作，节点模式利于模块化拼装和使用，在视频领域类似于达芬奇Fusiom或者Nuke，和3D领域的Blender材质界面都是用节点的思路构建的，这篇给大家讲解的是ComfyUI安装过程中的报错指南、环境配置和脚本更新相关问题

## 一、报错指南


## 1.1 代理错误

大部分ComfyUI玩家都是从WebUI转移而来，在配置WebUI时网上大部分教程在玩家们无法成功clone某个库或者模型时都会建议使用代理而不是直接从Github克隆，所以在配置ComfyUI时我们首先要解决的问题就是移除代理




方案一：移除全局代理设置

**如果依然无法下载，就需要进一步检查**

### 1. 检查全局代理设置

首先，检查 Git 的全局代理设置。在命令行中运行以下命令：

```
git config --global --get http.proxy
```

这些命令将显示 Git 是否有为 http 和 https 设置全局代理。

### 2. 移除全局代理设置

如果发现有设置代理，可以通过以下命令来移除它们：

```
git config --global --unset http.proxy
```

这会从全局 Git 配置中移除代理设置

### 3. 检查仓库级别的代理设置

还应该检查特定仓库的代理设置。在仓库的目录中运行：

```
git config --get http.proxy
```

如果这些命令返回了代理地址，就可以使用以下命令移除它们：

```
git config --unset http.proxy
```

### 4. 尝试克隆仓库

方案二：用命令行进一步检查


## 1.2 检查代理

代理设置通常在网络或浏览器设置中进行配置。可以检查浏览器的设置来确定是否使用了代理。

对于 Chrome：在地址栏输入 chrome://settings/，然后搜索“代理”或进入“高级设置”查找网络设置。

对于 Firefox：在地址栏输入 about:preferences，然后在“网络设置”中查看代理配置。

对于 Edge：在地址栏输入 edge://settings/，然后搜索“代理”设置。

**当然，如果不熟悉 *Git* 命令行，还有一些简单的方法来检查是否有代理设置影响网络连接**

### *1.* 查看浏览器代理设置

代理设置通常在网络或浏览器设置中进行配置。可以检查浏览器的设置来确定是否使用了代理。

- 对于 Chrome：在地址栏输入 chrome://settings/，然后搜索“代理”或进入“高级设置”查找网络设置。
- 对于 Firefox：在地址栏输入 about:preferences，然后在“网络设置”中查看代理配置。
- 对于 Edge：在地址栏输入 edge://settings/，然后搜索“代理”设置。

### *2.*检查操作系统的代理设置

还可以在操作系统级别检查代理设置。

- Windows：转到“设置” -> “网络和互联网” -> “代理”。
- macOS：转到“系统偏好设置” -> “网络” -> 选择网络服务（例如 Wi-Fi 或以太网） -> “高级” -> “代理”。

## 1.3 整包下载

Tips：使用这种方法下载的仓库不会包含 Git 版本历史。如果需要保持仓库的更新，将需要定期手动下载最新的 ZIP 文件，或解决 Git 克隆命令的问题。

**当然还有最简单的方法，就是直接去*Github*下载整个文件夹**

**1. 打开仓库的 *GitHub* 页面：**

在网页浏览器中，打开要下载的仓库的 GitHub 页面。例如，如果要下载的仓库是 https://github.com/Stability-AI/stability-ComfyUI-nodes，则在浏览器中直接访问这个链接。

**2. 找到“*Code*”按钮：**

在仓库页面上，找到并点击“Code”（代码）按钮。这个按钮通常位于仓库文件列表的上方。

**3. 下载 *ZIP*：**

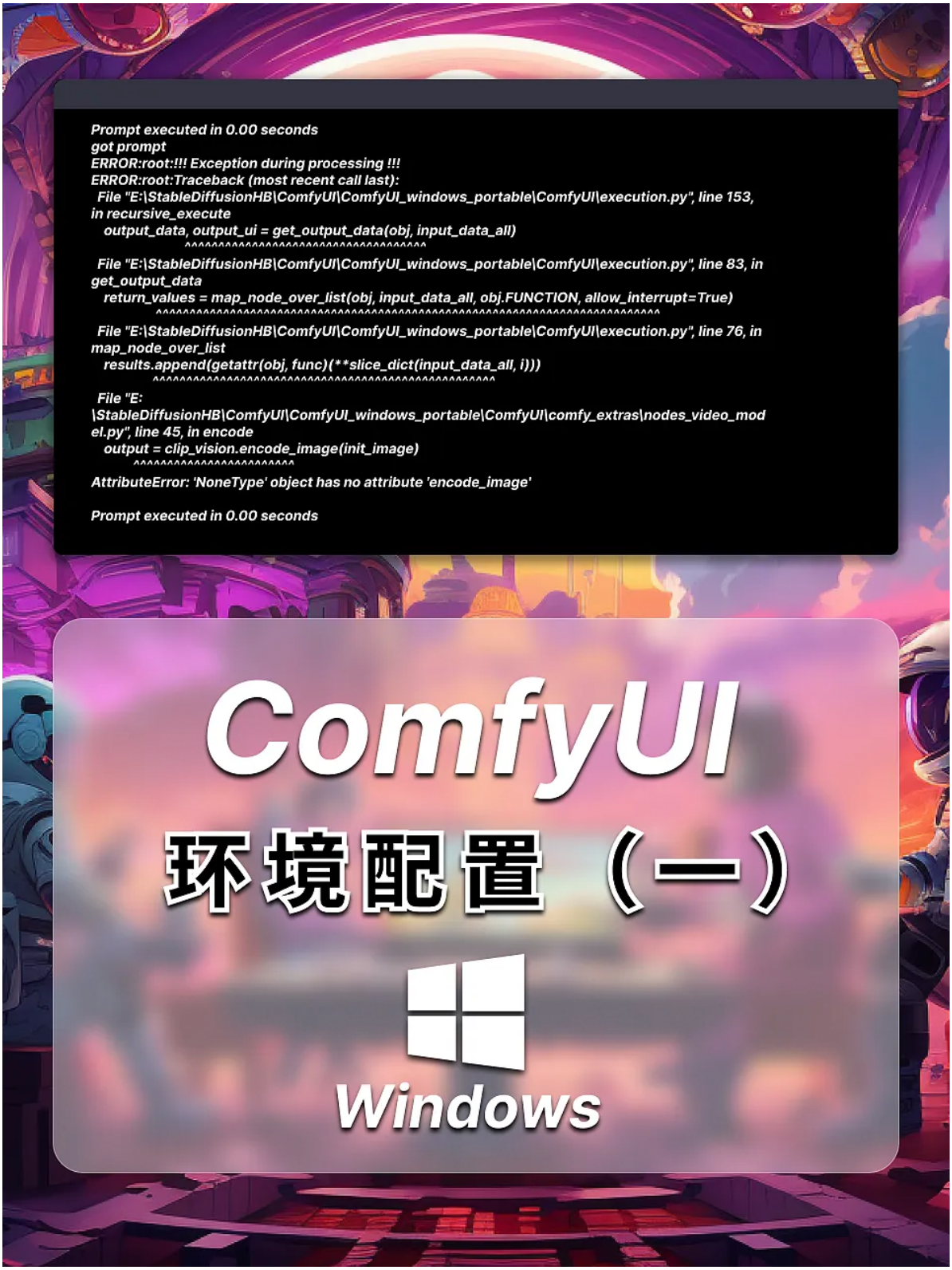
在弹出的小菜单中，选择“Download ZIP”（下载 ZIP）。这将下载仓库的当前状态作为一个 ZIP 文件。

**4.解压文件：**

下载完成后，找到 ZIP 文件并解压它。解压后，您将得到仓库中的所有文件和文件夹。

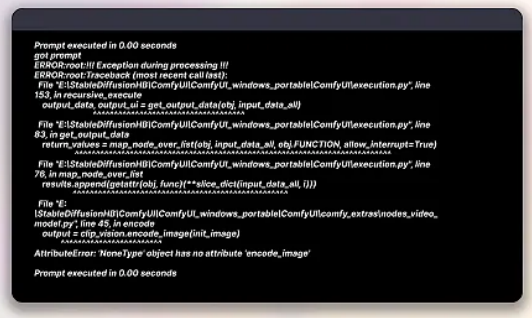
Tips：使用这种方法下载的仓库不会包含 Git 版本历史。如果需要保持仓库的更新，将需要定期手动下载最新的 ZIP 文件，或解决 Git 克隆命令的问题。

# 二、环境配置




## 1.1 乱码错误

运行 ComfyUI 相关的 Python 代码时常出现的异常： AttributeError: 'NoneType' object has no attribute 'encode_image' 指示在尝试调用 encode_image 方法时，clip_vision 对象是 None 类型，而不是预期的对象类型。




运行 ComfyUI 相关的 Python 代码时常出现的异常：***AttributeError: 'NoneType' object has no attribute 'encode_image'*** 指示在尝试调用 encode_image 方法时，clip_vision 对象是 None 类型，而不是预期的对象类型。

**问题引起的原因：**

**对象未正确初始化：**clip_vision 对象可能没有被正确初始化。缺少必要的库或模块，或者初始化代码存在问题。

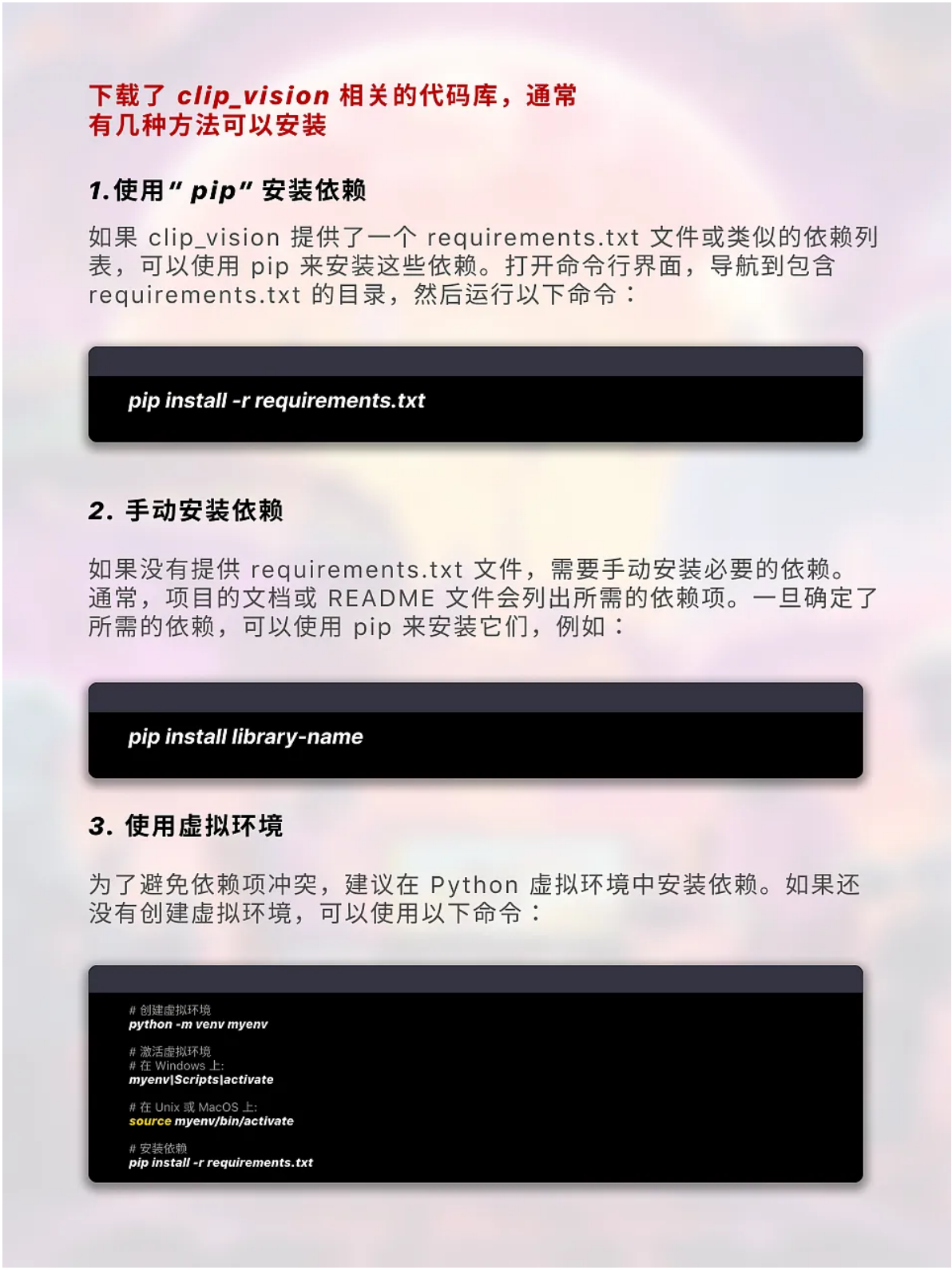
用大家能理解的说法就是缺少依赖项，像市面上大部分整合包启动器，列如铁锅炖虽然有180G的文件大小，但是只自带模型和插件，运行的环境和依赖项依然需要我们自行安装

**解决方法：**

下载clip_vision，并安装依赖即可

## 1.2 安装依赖

如果 clip_vision 提供了一个 requirements.txt 文件或类似的依赖列表，可以使用 pip 来安装这些依赖。打开命令行界面，导航到包含 requirements.txt 的目录，然后运行以下命令：




方法一

## 1.3 更新pip

更新 pip :pip 是 Python 的包管理器，用于安装和管理 Python 包。ComfyUI使用特定的 Python 环境，所以需要确保在正确的环境中更新 pip

**更新 *pip* :**pip 是 Python 的包管理器，用于安装和管理 Python 包。ComfyUI使用特定的 Python 环境，所以需要确保在正确的环境中更新 pip


**但*pip*版本不影响*ConfyUI*运行，个人是不建议更新，下面是更新 *pip* 的步骤：**

**1.打开命令行界面：**

- 在 Windows 上，可以搜索“命令提示符”或“CMD”并打开它。
- 在 macOS 或 Linux 上，可以打开“终端”。

**2.运行更新命令：**

- 根据的日志，使用特定的 Python 环境来更新 pip。一般情况下，命令如下：

```
D:\ComfyUI\Blender_ComfyUI\Blender_ComfyUI\python_embeded\python.exe -m pip install --upgrade pip
```

- 这个命令告诉 Python 运行 pip 模块，并调用 install 命令来更新 pip 本身。

**3.等待命令执行完毕：**

- 更新过程会在命令行中显示进度。一旦完成，pip 将被更新到最新版本。

**4.验证更新：**

- 更新完成后，可以运行以下命令来验证 pip 的版本：

```
D:\ComfyUI\Blender_ComfyUI\Blender_ComfyUI\python_embeded\python.exe -m pip --version
```

- 这将显示当前安装的 pip 版本，可以确认它是否已经更新到最新版本。

## 1.4 模型缺失

Tips：确保所有缺失的模块都已经被正确安装。可以使用 Python 的包管理器 pip 安装这些缺失的模块。运行这些安装命令时，确保使用的是与 ComfyUI 相关联的 Python 环境

**环境配置日志解决：**

***1.“No module named 'qrcode”***

需要安装 qrcode 模块。运行 pip install qrcode 来安装。

```
pip install qrcode
```

***2.” No module named 'librosa“***

需要安装 librosa 模块，用于音频处理。运行 pip install librosa 来安装。

***3. ” No module named 'omegaconf“***

需要安装 omegaconf 模块。可以通过运行 pip install omegaconf 来安装

***4. “No module named 'numexpr”***

需要安装 numexpr 模块。可以通过运行 pip install numexpr 来安装。

***5. “No module named 'timm”***

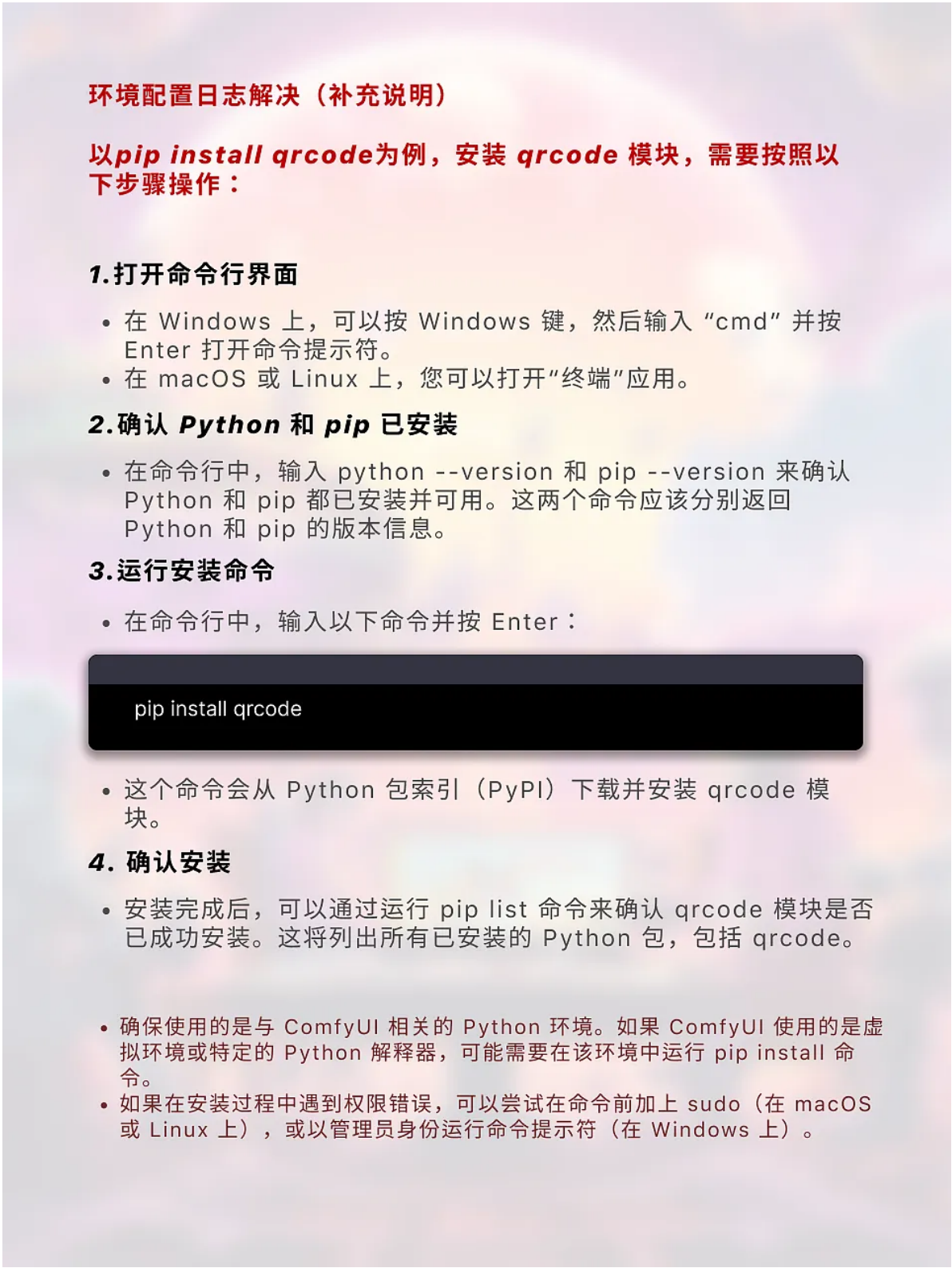
需要安装 timm 模块。可以通过运行 pip install timm 来安装

Tips：确保所有缺失的模块都已经被正确安装。可以使用 Python 的包管理器 pip 安装这些缺失的模块。运行这些安装命令时，确保使用的是与 ComfyUI 相关联的 Python 环境

## 1.5 补充说明

确保使用的是与 ComfyUI 相关的 Python 环境。如果 ComfyUI 使用的是虚拟环境或特定的 Python 解释器，可能需要在该环境中运行 pip install 命令。

如果在安装过程中遇到权限错误，可以尝试在命令前加上 sudo（在 macOS 或 Linux 上），或以管理员身份运行命令提示符（在 Windows 上）。




## 2.1 安装缺少依赖库

例如，pip install requests==2.25.1 会安装 requests 库的 2.25.1 版本。

**安装缺少的依赖库：**

**1.确认需要安装的库：**

- 从你提供的日志中找出缺少的库名称。例如，如果日志中显示 No module named 'xyz'，那么 xyz 就是需要安装的库。

**2. 打开命令行界面：**

- 在 Windows 上，你可以使用命令提示符（CMD）或 PowerShell。
- 确保你的命令行界面使用的是与 ComfyUI 相同的 Python 环境。

**3. 安装库：**

- 在命令行中，使用以下命令来安装库：

```
pip install library_name
```

- 将 library_name 替换为你需要安装的库的名称。

**4. 安装特定版本的库：**

- 如果需要安装特定版本的库，可以使用以下命令：

```
pip install library_name==version_number
```

- 例如，pip install requests==2.25.1 会安装 requests 库的 2.25.1 版本。

## 2.2 依赖库安装补充说明

Tips：如果你在安装过程中遇到任何困难，不要犹豫查看相关库的官方文档，或者在开源社区（如 GitHub、Stack Overflow）搜索相关的讨论和解决方案。

**安装缺少的依赖库：**

**5.升级 *pip*:**

- 在安装依赖库之前，建议升级 pip 以确保可以访问最新的包。使用以下命令升级 pip：

```
pip install --upgrade pip
```

**6. 处理安装错误:**

- 如果在安装过程中遇到错误，细阅读错误信息。它通常会提供有关问题原因和可能的解决方案的线索。
- 对于一些特殊的库，可能需要额外的依赖或配置。在这种情况下，查看该库的官方文档会很有帮助。

**7. 验证安装:**

- 安装完成后，可以在命令行中运行 pip list 来查看已安装的库列表，确认你的库是否已正确安装。

**8. 重启应用:**

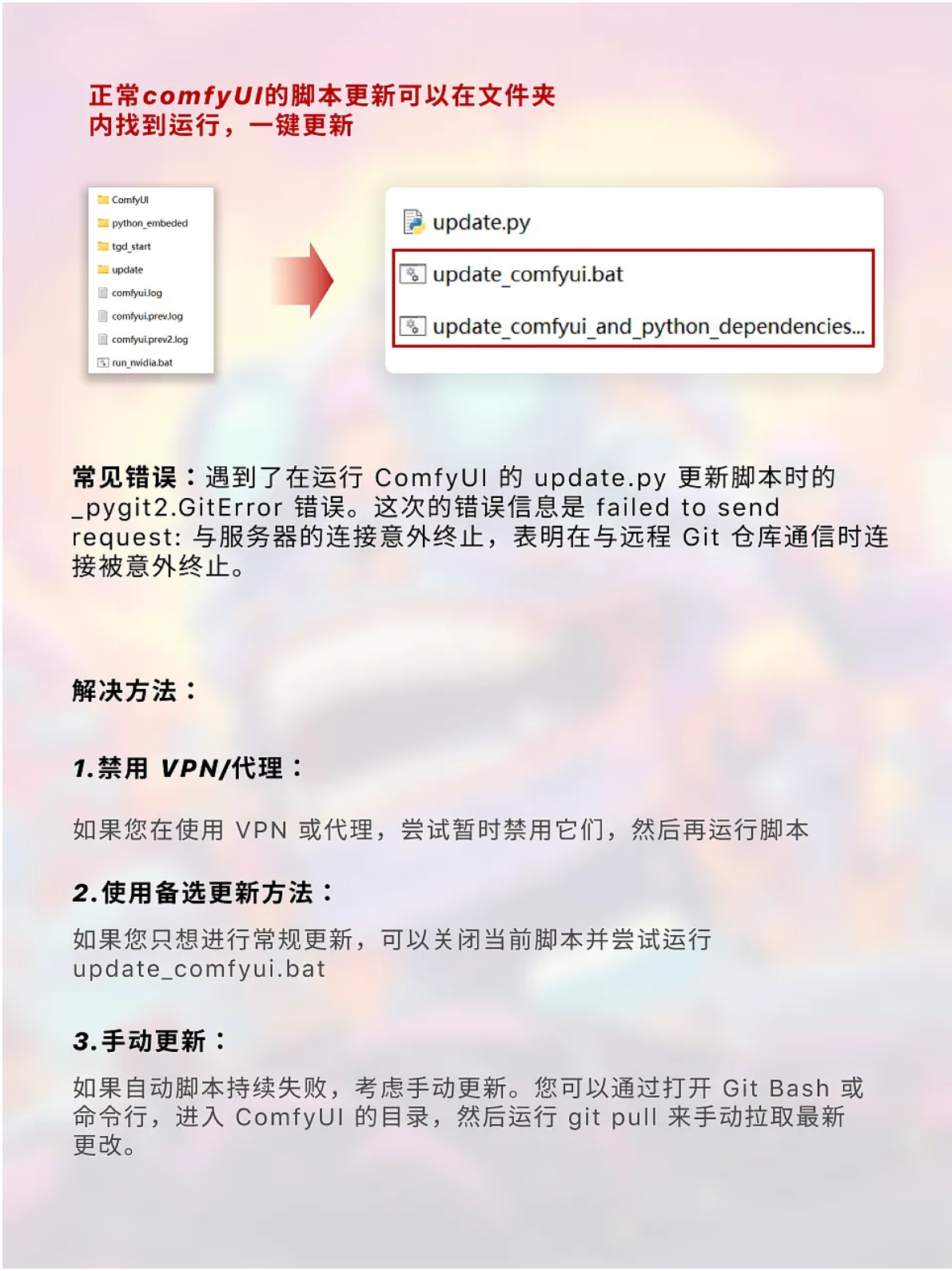
- 安装完依赖库后，重新启动 ComfyUI 以使更改生效。

Tips：如果你在安装过程中遇到任何困难，不要犹豫查看相关库的官方文档，或者在开源社区（如 GitHub、Stack Overflow）搜索相关的讨论和解决方案。

## 三、脚本更新


## 1.1 常见错误

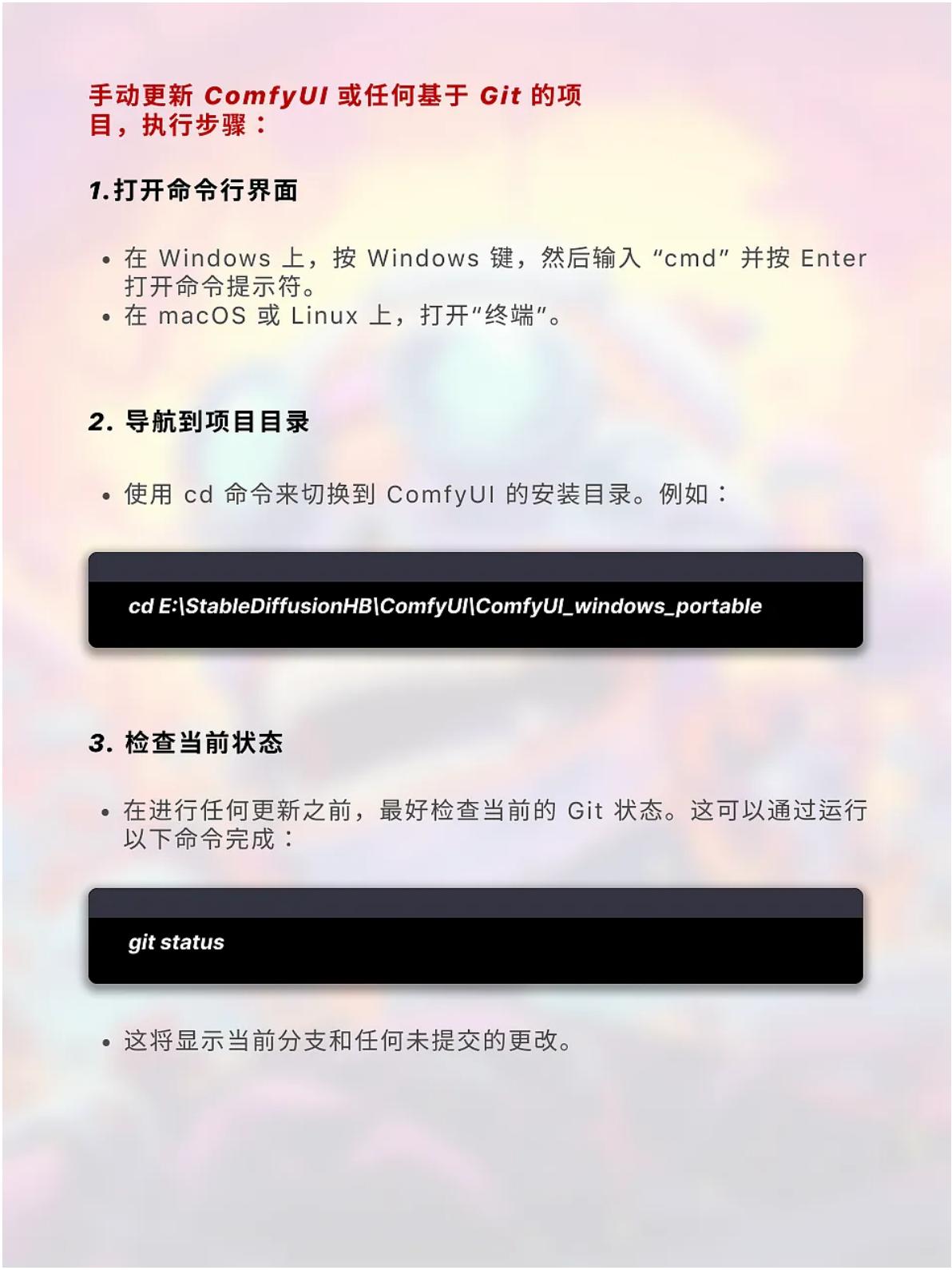
常见错误：遇到了在运行 ComfyUI 的 update.py 更新脚本时的 _pygit2.GitError 错误。这次的错误信息是 failed to send request: 与服务器的连接意外终止，表明在与远程 Git 仓库通信时连接被意外终止。




## 1.2 手动更新

在 Windows 上，按 Windows 键，然后输入 “cmd” 并按 Enter 打开命令提示符。

在 macOS 或 Linux 上，打开“终端”。




## 1.3 手动更新补充

手动更新 ComfyUI 或任何基于 Git 的项目，执行步骤:

**手动更新 *ComfyUI* 或任何基于 *Git* 的项目，执行步骤：**

**4.拉取最新更改**

- 使用以下命令从远程仓库拉取最新的更改：

```
git pull
```

- 这个命令将会合并远程仓库中的更改到当前的工作分支

**5. 解决可能的冲突**

- 如果在拉取过程中出现代码冲突，Git 将提示您解决这些冲突。您可能需要手动编辑文件来解决这些冲突

**6. 安装依赖**

- 如果项目有依赖项更新，您可能需要使用 pip 安装或更新这些依赖。通常，这可以通过以下命令完成：

```
pip install -r requirements.txt
```

- 确保在项目的相应环境中运行此命令，特别是如果您使用了虚拟环境
- 在手动更新前，确保备份的工作，以防万一更新过程中出现问题

## 1.4 手动跟新补充（二）

如使用 git pull 命令无法更新，可以尝试以下手动更新的方法：

**如使用 *git pull* 命令无法更新，可以尝试以下手动更新的方法：**

**1. 下载最新版本的源代码**

- 访问 ComfyUI 的 GitHub 仓库页面：https://github.com/comfyanonymous/ComfyUI。
- 找到“Code”按钮，点击它，然后选择“Download ZIP”。

**2. 解压下载的文件**

- 将下载的 ZIP 文件解压到一个新的文件夹中

**3. 替换旧文件**

- 复制解压后的所有文件和文件夹
- 前往您之前使用 ComfyUI 的目录（E:\StableDiffusionHB\ComfyUI\ComfyUI_windows_portable\ComfyUI）
- 粘贴并覆盖旧文件。这个步骤将用下载的最新文件更新您的 ComfyUI 安装

**4. 检查依赖项**

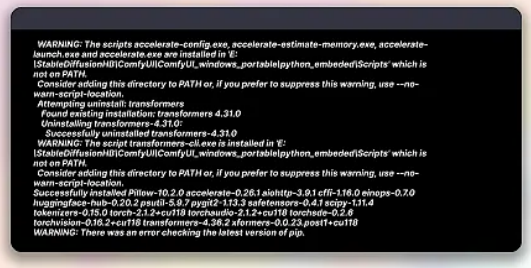
- 如果 ComfyUI 有任何依赖项或要求特定的安装步骤，请确保按照官方文档指示进行操作
- 可能需要在 ComfyUI 目录中运行如 pip install -r requirements.txt 的命令来安装 Python 依赖项

## 1.5 常见警告信息修正

通过这种方式，可以直接在命令行中运行这些脚本，不需要指定完整的路径。

要添加目录到 PATH，你需要打开系统属性（右键点击“此电脑”或“我的电脑”，选择“属性”），然后进入“高级系统设置”。在弹出的窗口中，选择“环境变量”，并在“系统变量”区域找到并编辑 PATH 变量，添加 E:\StableDiffusionHB\ComfyUI\ComfyUI_windows_portable\python_embeded\Scripts。

**常见的警告信息修正**




**1. 将脚本所在目录添加到 *PATH*:**

- 通过这种方式，可以直接在命令行中运行这些脚本，不需要指定完整的路径。
- 要添加目录到 PATH，你需要打开系统属性（右键点击“此电脑”或“我的电脑”，选择“属性”），然后进入“高级系统设置”。在弹出的窗口中，选择“环境变量”，并在“系统变量”区域找到并编辑 PATH 变量，添加 E:\StableDiffusionHB\ComfyUI\ComfyUI_windows_portable\python_embeded\Scripts。

**2. 忽略这个警告:**

- 如果不需要在命令行中直接运行这些脚本，可以选择忽略这个警告。
- 使用 --no-warn-script-location 选项可以在安装过程中避免这类警告。

Tips：最后，关于 "WARNING: There was an error checking the latest version of pip." 的警告，这可能是因为网络连接问题或者 pip 服务器的问题。一般来说，这不会影响软件的正常使用。如果你想确保 pip 是最新版本，可以尝试运行 python -m pip install --upgrade pip 来更新它。如果这个警告持续出现，并且你遇到了相关问题，那么可能需要检查你的网络连接或者稍后再尝试

# 四、后记

最后，关于 "WARNING: There was an error checking the latest version of pip." 的警告，这可能是因为网络连接问题或者 pip 服务器的问题。一般来说，这不会影响软件的正常使用。如果你想确保 pip 是最新版本，可以尝试运行 python -m pip install --upgrade pip 来更新它。如果这个警告持续出现，并且你遇到了相关问题，那么可能需要检查你的网络连接或者稍后再尝试。

ComfuUI节点式工作流最大的作用和效用对于个人而言优点就是更利于分享，一键导入别人现成工作流，或是把某一个模块分享出去，今后会开始持续分享ComfyUI的相关学习知识，大家的关注点赞就是我最大的动力！

# 二、ComfyUI基础入门

## 前言

对于comfyui来说，所有的插件都是额外的一些节点，不同的节点为我们提供了不同的功能，当我们想调用更多的功能时，在现在框架下没有这些功能的话就需要下载额外的支持这些功能节点

## 一、软件安装篇

### 1.1 ComfyUI优势


SD是固定了底层架构的ConfyUI，像文生图，图生图都是confyui里已经固定的工作流

ConfyUI利于专业团队使用，多人协作，节点模式利于模块化拼装和使用，在视频领域类似于达芬奇Fusiom或者Nuke，和3D领域的Blender材质界面都是用节点的思路构建的

## 1.2 ComfyUI配置要求


打开ComfyUI的Github，下载ComfyUI压缩包并解压就可在本地使用

## 1.3 ComfyUI模型路径设置


文件后缀名yaml后面example为样本文件，并没起作用，如需文件起作用，则需要删除.example后缀

## 1.3.1 ComfyUI模型路径设置


Tips：只需更改对应根目录和control目录即可，其他自行核对下

如果电脑种已经安装Stablediffusion则可以把模型路径设置到SD文件下，节省空间，这样就不用下载两份模型或复制多份模型产生额外空间占用，【如果没安装SD则跳过此部分】

注意：此处Controlnet目录是不对的，Controlnet模型目录一般是在根目录下拓展文件夹extensions下的Controlnet里modes文件夹中，所以需要把路径指定到对应目录来1.4

### 1.3.2 ComfyUI模型路径设置


如果没安装SD则默认路径就在ComfyUI根目录Modus中

### 1.3.3 ComfyUI启动


稍稍等待一段时间，当跳出链接和网页时时则代表启动成功

# 二、插件安装篇

## 1.1 ComfyUI插件安装


找到ComfyUI根目custom_nodes

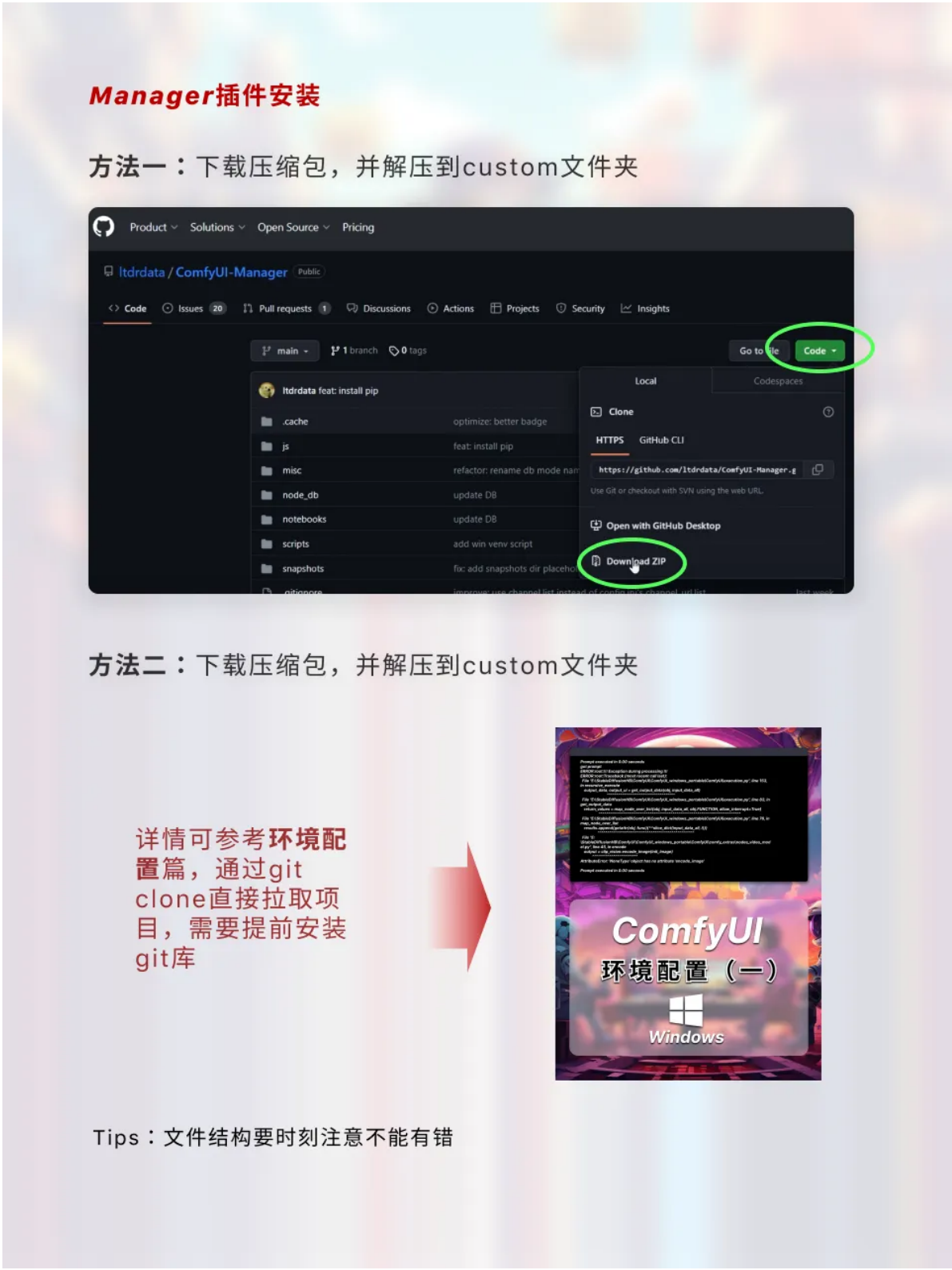
对于comfyui来说，所有的插件都是额外的一些节点，不同的节点为我们提供了不同的功能，当我们想调用更多的功能时，在现在框架下没有这些功能的话就需要下载额外的支持这些功能节点

安装Manager插件：

Manager插件时方便管理整个插件的组，包括检测当前模板是否有丢失的插件，同时自动更新插件、整合插件等，可以在其中搜索你想要的任意插件

插件地址：https://github.com/ltdrdata/ComfyUI-Manager

## 1.2 Manager插件安装

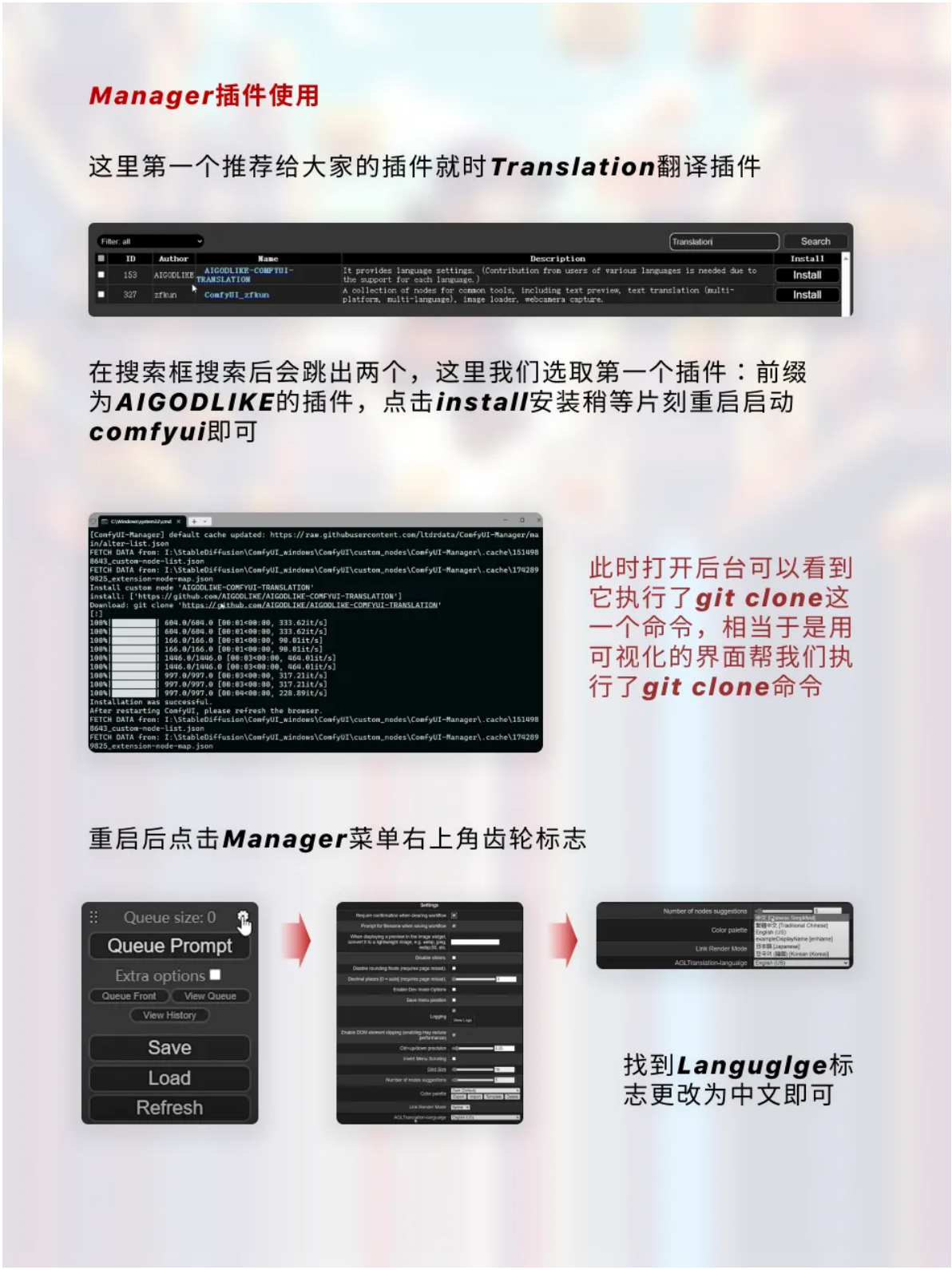



Tips：文件结构要时刻注意不能有错

## 1.3 Manager插件使用


成功运行comfyui后在窗口的右边会悬浮Manager窗口

## 1.4 Manager插件推荐




在搜索框搜索后会跳出两个，这里我们选取第一个插件：前缀为AIGODLIKE的插件，点击install安装稍等片刻重启启动comfyui即可

这里第一个推荐给大家的插件就时Translation翻译插件

打开后台可以看到它执行了git clone这一个命令，相当于是用可视化的界面帮我们执行了git clone命令

## 1.5 Manager插件使用


此时页面已经变为中文界面，如果要切换语言只需要点击Manager菜单上切换语言即可，英文界面点击Switch Locale同理

到这里关于ComfyUI的基础界面配置就完成了，下面就是一个简单的ComfyUI的提示词生成工作流

## 三、结尾

这就是我入门笔记程序安装、插件安装以及程序的基本内容了，下一篇将根据ComfyUI核心的生成部分以及SD扩散模型的基本运算结构来分解讲解

# 三、ComfyUI工作流节点/底层逻辑详解

# 前言

我们所有训练的图形都是512x512是真实空间的图像，也就是Pixel space，是每一个像素生成的图像，同时我们也叫它像素空间，当我们训练时，其实是对图片进行了一次多重多维度的压缩，压缩以后的图片像素只有64x64，机器会在这个空间（潜在空间）里进行训练和运算，再从这个空间（潜在空间）中合成我们所有想要的信息，最后转换成肉眼能看到的真实空间，也就是像素空间的完整图像

# 一、ComfyUI 基础概念理解

## 1.1 任何工作流的核心枢纽都是采样器


所以任何工作流的第一步都是创建采样器

e.g.SD中采样方法DPM++2M Karras在ComfyUI中则需要选择采样器：DPM++2M；调度器Karras。两者分开选取

## 1.2 不同采样器最终结果差别不会太大


lcm采样器通常是配和实时绘画工作流使用

如果想要提高图片质量，通常会使用DPM++采样器，但对于提升图片质量DPM++2M是不如带sde后缀的采样器，sde的效果是增加了图像的发挥性和想象力，对于生成的图像会有更多的变化，出现额外惊喜，Uni PC是后期推出的采样器，优点类似于lcm采样器，可以在十部之内出不错的图像

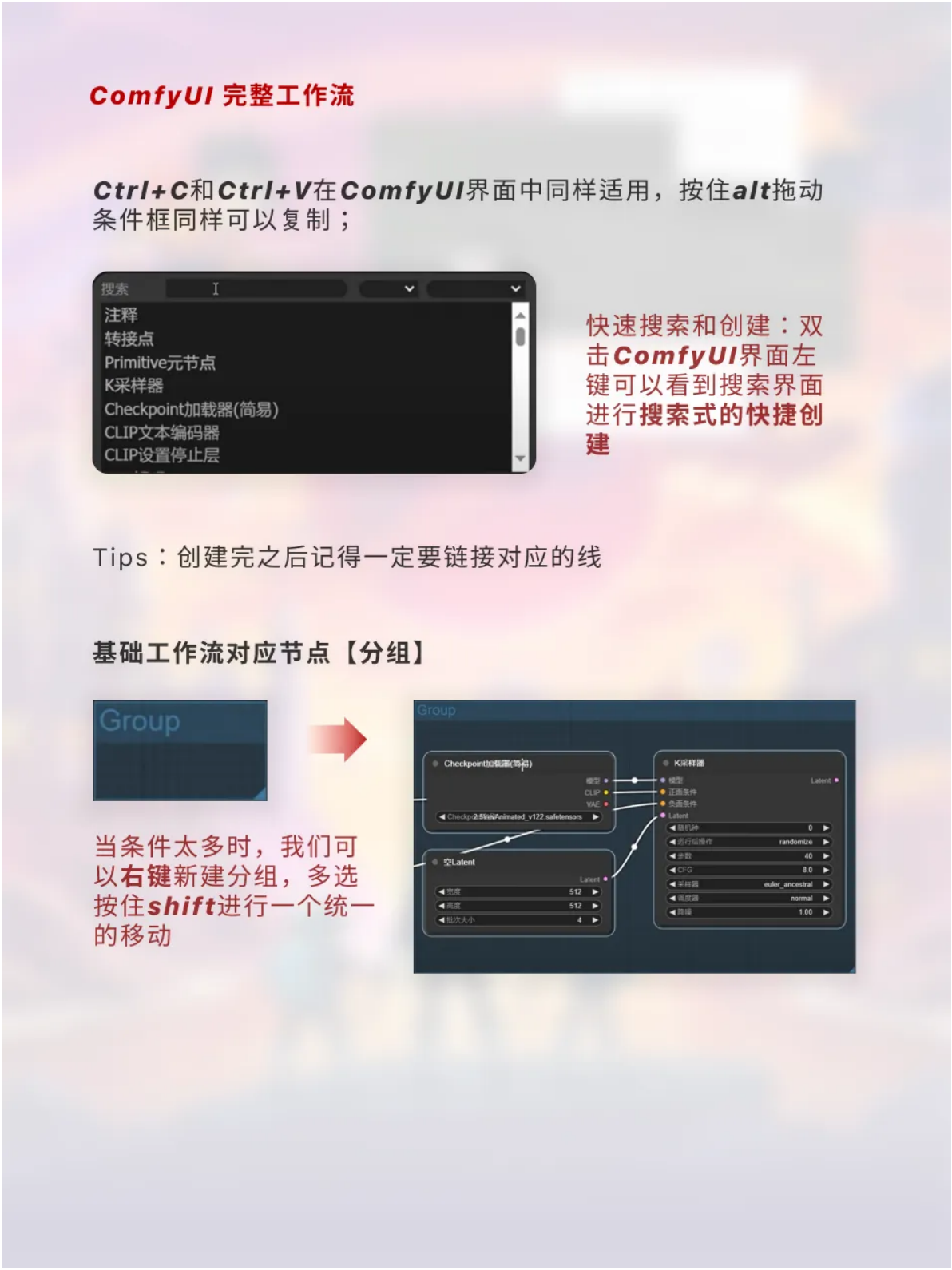
## 1.3 调度器和采样器不同，常用的数量并不多


sgm_uniform也是定向性降噪，需要配合lcm实时绘画搭配使用

降噪幅度就是根据我们采样步数来设置降噪多少，默认的文生图降噪就是1即可

## 1.4 ComfyUI 完整工作流


正负面条件则为关键词，CLIP编码器中的文本都会输入到对应的条件里面去（负面提示词同理）




Tips：创建完之后记得一定要链接对应的线

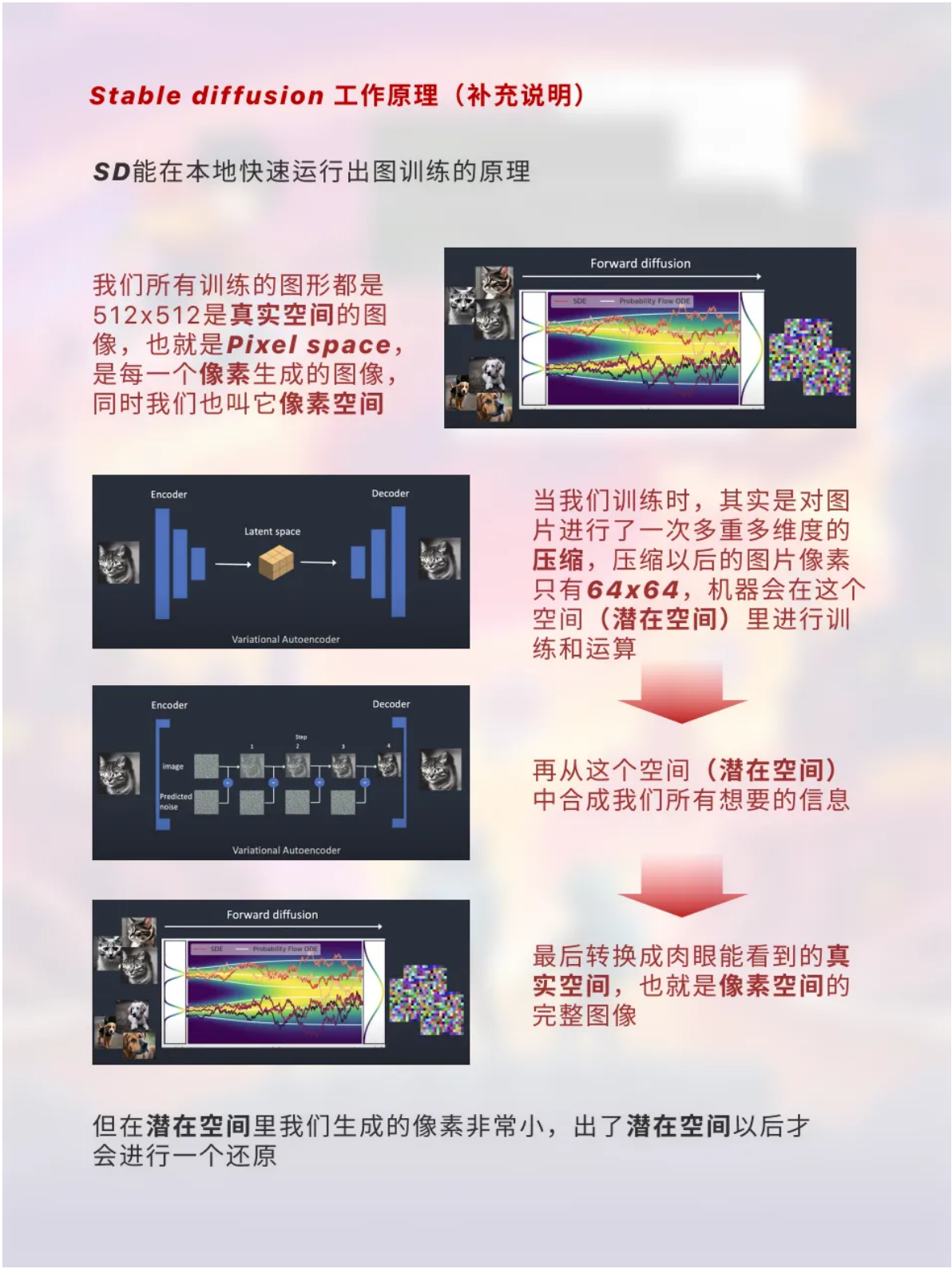
Ctrl+C和Ctrl+V在ComfyUI界面中同样适用，按住alt拖动条件框同样可以复制；

快速搜索和创建：双击ComfyUI界面左键可以看到搜索界面进行搜索式的快捷创建；

当条件太多时，我们可以右键新建分组，多选按住shift进行一个统一的移动

## 二、Stable diffusion 工作原理（补充说明）

## 1.1 SD能在本地快速运行出图训练的原理




最后转换成肉眼能看到的真实空间，也就是像素空间的完整图像

我们所有训练的图形都是512x512是真实空间的图像，也就是Pixel space，是每一个像素生成的图像，同时我们也叫它像素空间；当我们训练时，其实是对图片进行了一次多重多维度的压缩，压缩以后的图片像素只有64x64，机器会在这个空间（潜在空间）里进行训练和运算；再从这个空间（潜在空间）中合成我们所有想要的信息；最后转换成肉眼能看到的真实空间，也就是像素空间的完整图像；但在潜在空间里我们生成的像素非常小，出了潜在空间以后才会进行一个还原；

# 1.2 基础工作流对应节点【Latent】补充说明（一）


空的潜在空间多大也就直接决定了我们图片的大小

Latent即潜在空间，两个空间的转换；潜在空间Latent space，真实空间Pixel space，在工作流中经常会对空间进行转换；切换英文可能更方便大家理解，也就是Empty Latent Image一个空的潜在空间图像；

# 1.3 基础工作流对应节点【Latent】补充说明（二）


模型传递后的参数也是处于潜在空间之中

紧接上一篇中，大家知道了采样器，Checkpoint加载器和空Latent是处在潜在空间的；

Tips：最后记得VAE解码器中的VAE需要链接到Checkpoint模型加载器上的VAE

## 1.4 基础工作流对应节点【VAE解码】补充说明


如果模型没有VAE或者特定的挂载了一些VAE，就需要重新脱出一个VAE加载器（load VAE）单独读取一个VAE模型；

Tips：通常的VAE模型在我们从潜在空间转换为真实空间后，它的色彩、细节、光影都会有一点点变化，这就是VAE进行调整的一个过程；

## 1.5 基础工作流对应节点【文本编码器】


有了纯文字后我们还需要一个encoding的文本编码再传入采样器才能成立；

文本编码器之所以叫CLIP text encoding而不是直接叫做正/负项提示词是因为节点做了两个操作：比如说我们输入1 girl，1 girl在其中是作为字符串（str）类型的文本，有了这些字符串后不能直接输入给模型；模型需要的是字符串编码以后的数据，因为它处理的信息十分多，从机器的角度它是多模态处理，就是图像和文字是两种不同的处理方式；因此需要把文字编码成能解释如右图中内容的信息，才能将两种不同的信息同时运算；

Tips：这里提到原理是因为接下来，对文字处理时，我们也会遇到两种不同类型的文字信息，一种是encoding编完码以后的，另一种就是string纯文字信息，当我们了解原理后就不会把纯文字节点去链接到采样器上；

## 1.6 基础工作流


基础工作流

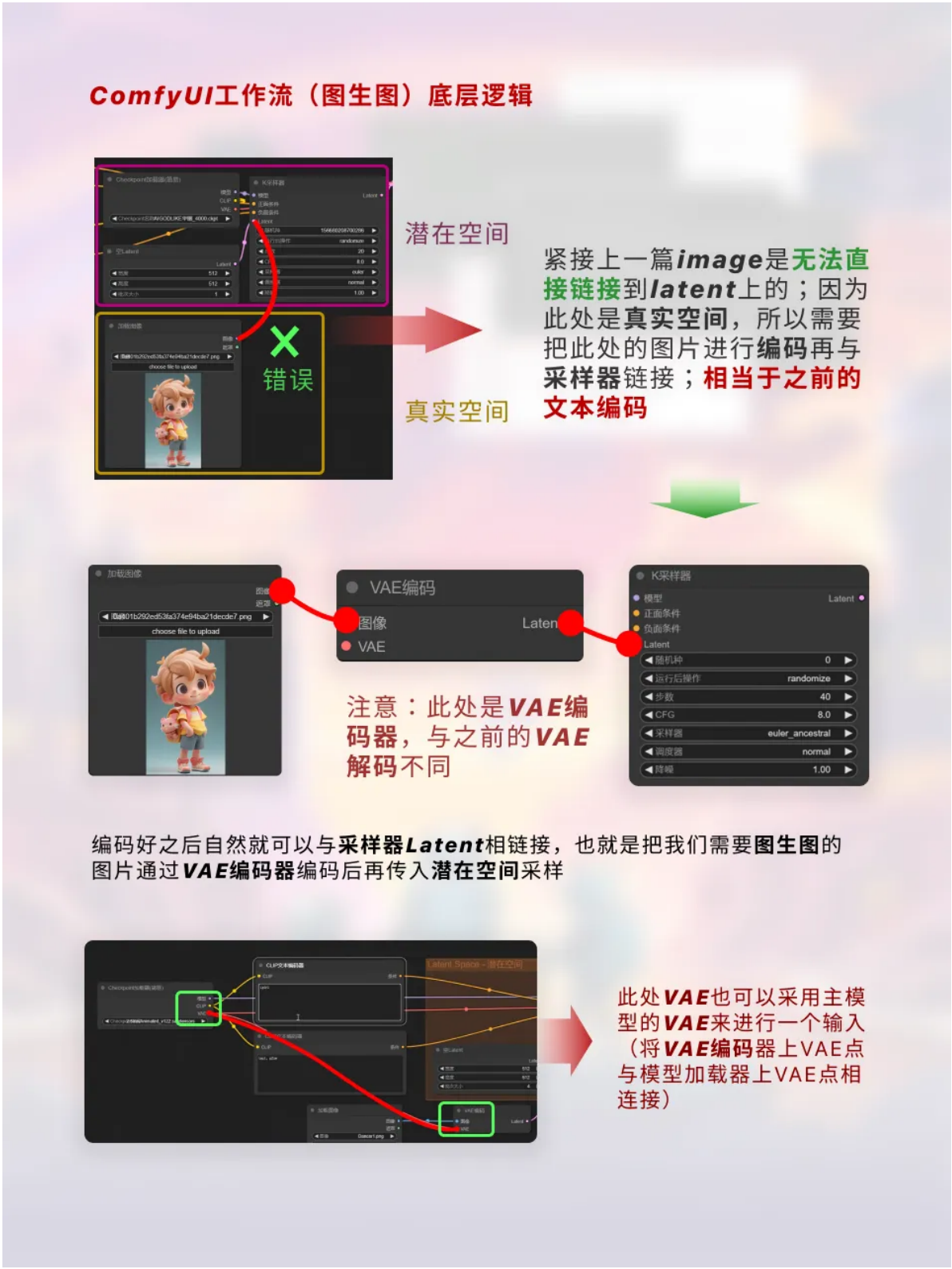
Tips：到这里我们一套基础的工作流就搭建完成了，最后记得要把CLIP文本编辑器的CLIP点与Checkpoint加载器的CLIP点相连接

# 三、工作流底层逻辑

## 1.1 ComfyUI工作流（图生图）底层逻辑


在了解完一个基础工作流后，我们就可以尝试用自己的理解去复刻SD当中的图生图工作流；基于上一篇章的学习，这里我们就知道了图生图应该在（空Latent）中做文章；这里我们图生图给采样器输入的就不能是一个空的图像，而变为一个指定图像，用指定图像去采样生成新图像，因此要把图像给到浅空间；输出有图像（IMAGR)输出和遮罩（MASK）输出，图像输出就是Pixel的真实空间的像素输出；遮罩输出一般不用管，是作为黑白蒙版的信息输出，在一些control net中才会偶尔用到

## 1.2 ComfyUI工作流（图生图）底层逻辑




注意：此处是VAE编码器，与之前的VAE解码不同

紧接上一篇image是无法直接链接到latent上的；因为此处是真实空间，所以需要把此处的图片进行编码再与采样器链接；相当于之前的文本编码；编码好之后自然就可以与采样器Latent相链接，也就是把我们需要图生图的图片通过VAE编码器编码后再传入潜在空间采样；此处VAE也可以采用主模型的VAE来进行一个输入（将VAE编码器上VAE点与模型加载器上VAE点相连接）；

## 1.3 ComfyUI工作流（图生图）底层逻辑


再K采样器中底部的降噪之前默认是1，也就是完全根据我们的步数进行百分百降噪；

我们是图生图工作流，既然有了图像的输入降噪就不是100%，我们可以给少一些改成0.4，也就是40%的重绘幅度，就会更像我们给出的图像（我们给出的图像是需要有具体分辨率，尽量与我们输出的分辨率相等，图像过大内存会崩溃，当然如果使用第三方插件来缩小/裁切也是可行的）；到这里我门就可以点击添加提示词队列来进行我们的图生图工作流程

## 1.4 ComfyUI工作流（图生图）底层逻辑


当我们一切就绪开始运行这个工作流后可以发现生成的图片与原图是十分接近的

这就是我们了解了空间问题以后，就能很容易的通过节点的自由链接实现图生图的简单功能，当然这个工作流还不是太完善，通常图生图工作流在SD中还需要进行一个提示词的反推和图片的修剪，再加一个imparting（图像的重绘）其实这些功能最后在ComfyUI中都会整合成一个一个的模块

# 四、必备插件补全

## 1.1 ComfyUI工作流插件安装


第一个呢就是提示词汉化插件，这与我们之间manager的翻译插件不同。类似于我们再web-UI中的补齐汉化和各种实时翻译大全的自定义节点；第二个呢就是一个自定义脚本，也就是大家常称呼的瑞士军刀。集合了很多WebUI中实用的小功能，非常全面：embedding补全、二级菜单、以及网络对齐、模型资料查看等等

## 1.2 安装步骤


接下来给大家演示 下安装步骤：

插件下载操作可参考我之前发的Stable Diffusion的报错指南链接，其中下载步骤相同，下载完两个压缩包之后直接拖入comfyUI根目录下并分别解压，此时我们就可以直接云运行，重新打开comfyUI

## 1.3 更新提示


Tips：如果一只卡在检测界面就是网络配置问题，需要科学上网

只是后就需要用到之前的手动下载；但依旧有一些节点文件再解压以后还是需要二次在网上配置下载，这时候就必须要用到科学上网了

## 1.4 结尾


当然也可以在manager中安装，在manager中安装就简单的多，只需要直接搜索即可，之前有提过，也可以直接复制完整的github地址，通过URL安装，在弹出的窗口中输入链接，点击确定就可拉取（注意两种方法都需要配置网络）

# 五、结尾

在了解完一个基础工作流后，我们就可以尝试用自己的理解去复刻SD当中的图生图工作流；

记得关注，我会持续为大家带来最新的ComfyUI教程哦。

# 四、ComfyUI节点技巧进阶/多模型串联

## 前言

在之前安装完汉化插件后，在新建节点中就会出现Alek节点组，一般我们用第一个简化版就够用，当我们深入学习后或者有了更好的翻译引擎比如deep的api接口，则可以选用高级版去探索


**本篇涵盖内容：**

- 文本编辑拓展说明
- CLIP设置停止层应用
- AlekPet辅助功能介绍
- ComfyUI Alek节点组详解
- Primitive元节点匹配与调用
- 字符串操作（function）详解
- KSampler（采样器）输入类型
- 单个/多个Lora模型连接/串联方式
- 如何单独运用翻译文本text节点输出

## 一、节点进阶详解

## 1.1 ALEK节点组


重点还是以上三类文本翻译

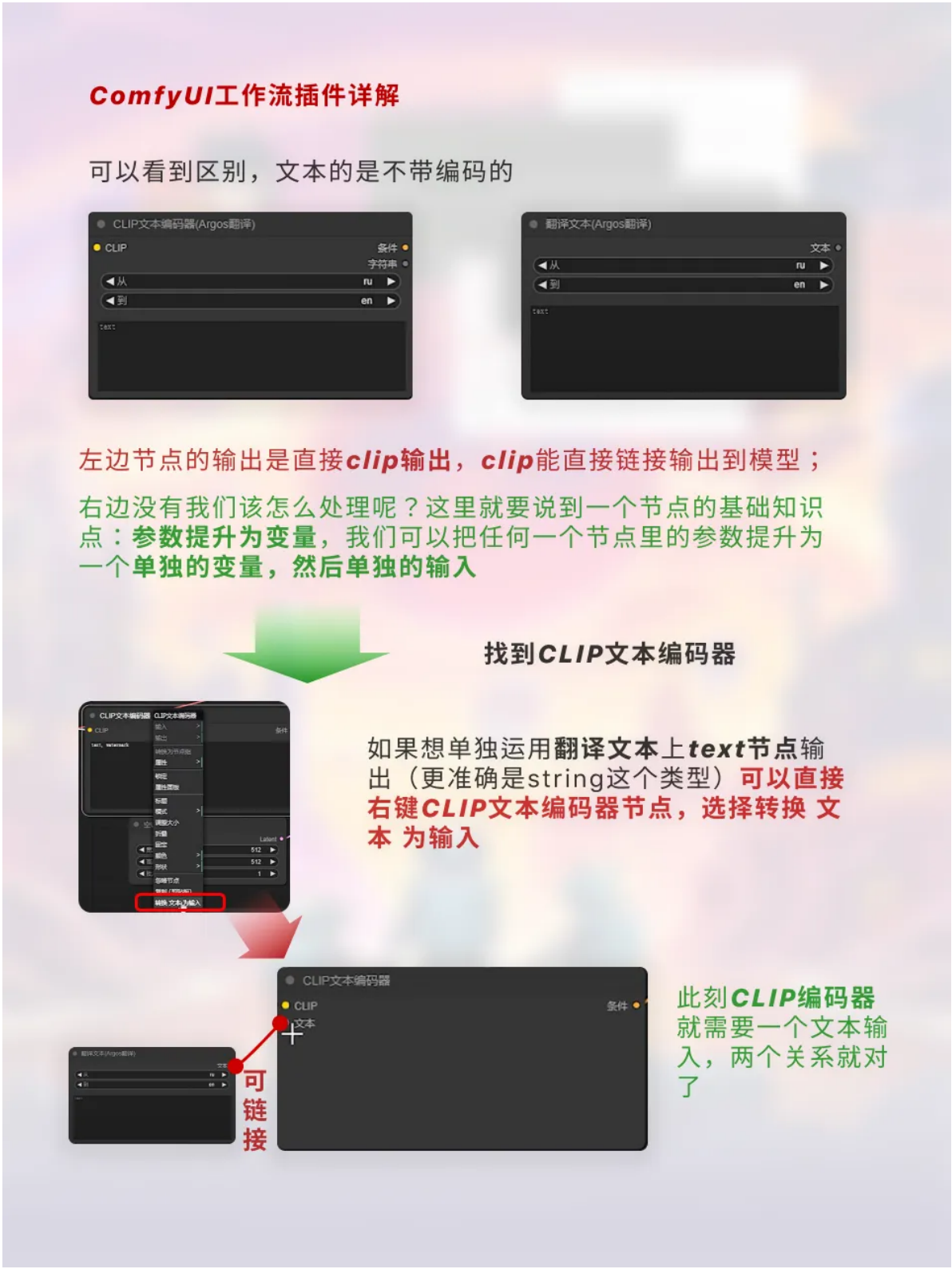
在安装完汉化插件后，在新建节点中就会出现Alek节点组，这就是插件提供给我们的插件包，我们所有下载的插件但凡是节点都会是一些列的节点，有的拓展包比较大，有的比较小；

像Alek拓展包就是比较轻量化的节点，我们把条件和文本的节点一一罗列出来，至于拓展是附带的一些小功能类似预览文本，图像则是像绘图等可以再之后放入我们的Ctrl net或图像处理的流程中，这里就不过多赘述了；

## 1.2 文本编码器


与工作流中CLIP文本编码器相同，负责接收我们输入文本，和把文本编码成Token；一般我们用第一个简化版就够用，当我们深入学习后或者有了更好的翻译引擎比如deep的api接口，则可以选用高级版去探索；

## 1.3 翻译节点

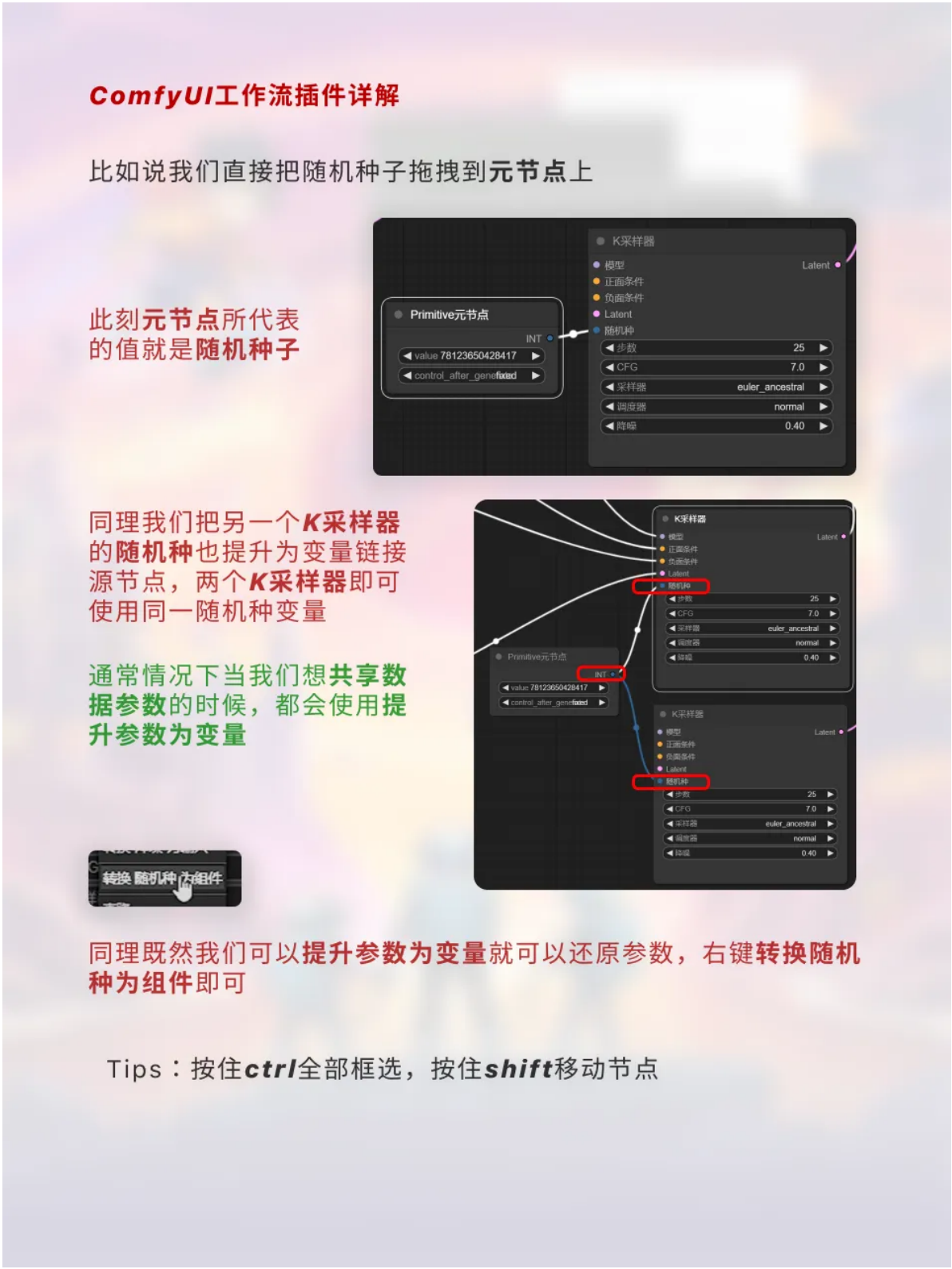



可以看到区别，文本的是不带编码的，左边节点的输出是直接clip输出，clip能直接链接输出到模型；右边没有我们该怎么处理呢？这里就要说到一个节点的基础知识点：参数提升为变量，我们可以把任何一个节点里的参数提升为一个单独的变量，然后单独的输入；如果想单独运用翻译文本上text节点输出（更准确是string这个类型）可以直接右键CLIP文本编码器节点，选择转换 文本 为输入

## 1.4 提升变量

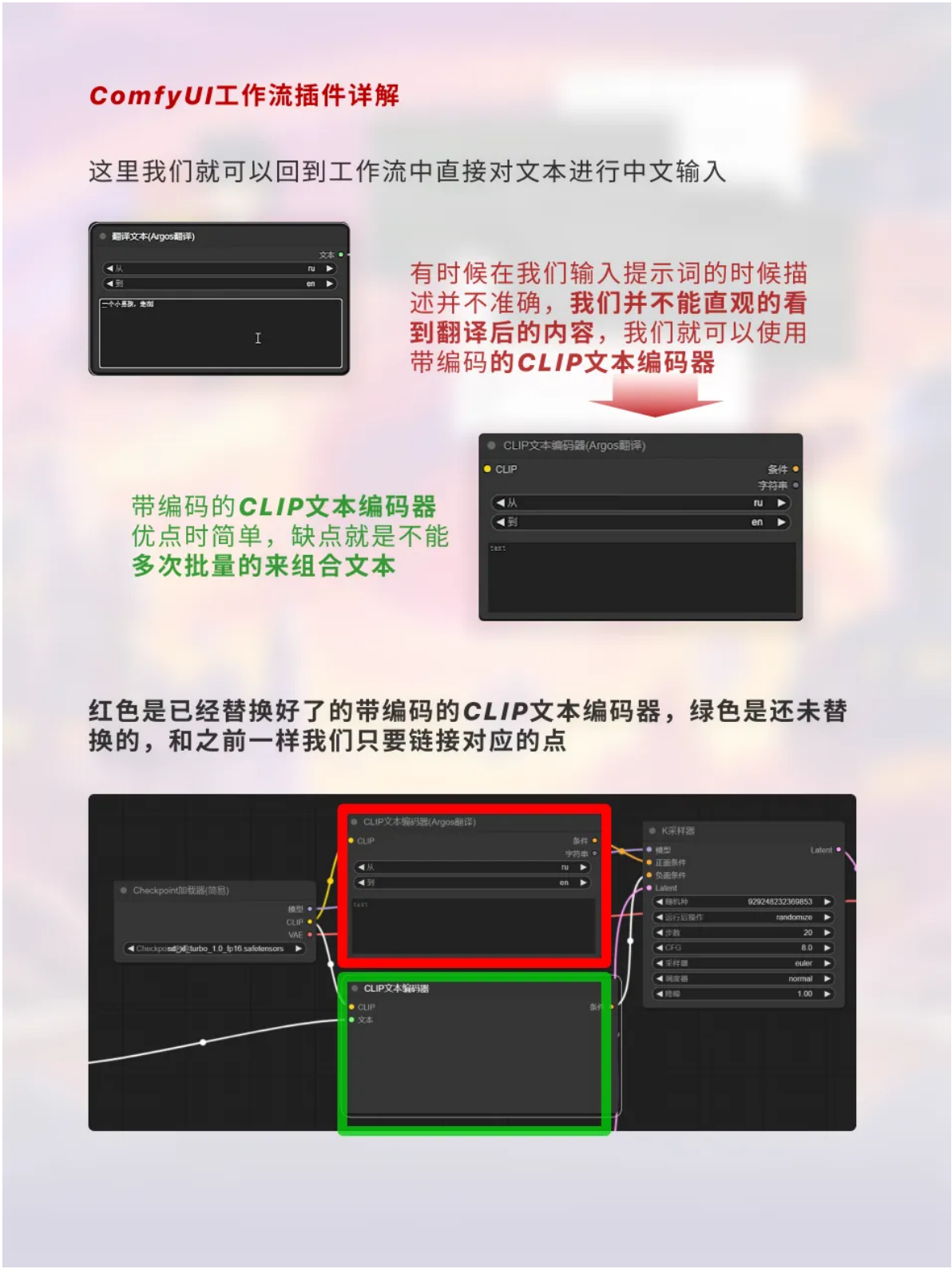

同样的方法我们也可以在正向提示词进行一个转换，这种提升变量的方式呢我们在节点里是经常用到的；比如说我们有两个K采样器，两组生成内容不一样（两种不同的提示词）但我们希望两组的随机种子是一样的固定某一个值，此时我们就可以用到提升变量把这个值提出来；

## 1.5 元节点




比如说我们直接把随机种子拖拽到元节点上，此刻元节点所代表的值就是随机种子；同理我们把另一个K采样器的随机种也提升为变量链接源节点，两个K采样器即可使用同一随机种变量；通常情况下当我们想共享数据参数的时候，都会使用提升参数为变量；同理既然我们可以提升参数为变量就可以还原参数，右键转换随机种为组件即可；

## 1.6 文本输入




有时候在我们输入提示词的时候描述并不准确，我们并不能直观的看到翻译后的内容，我们就可以使用带编码的CLIP文本编码器；带编码的CLIP文本编码器优点时简单，缺点就是不能多次批量的来组合文本；

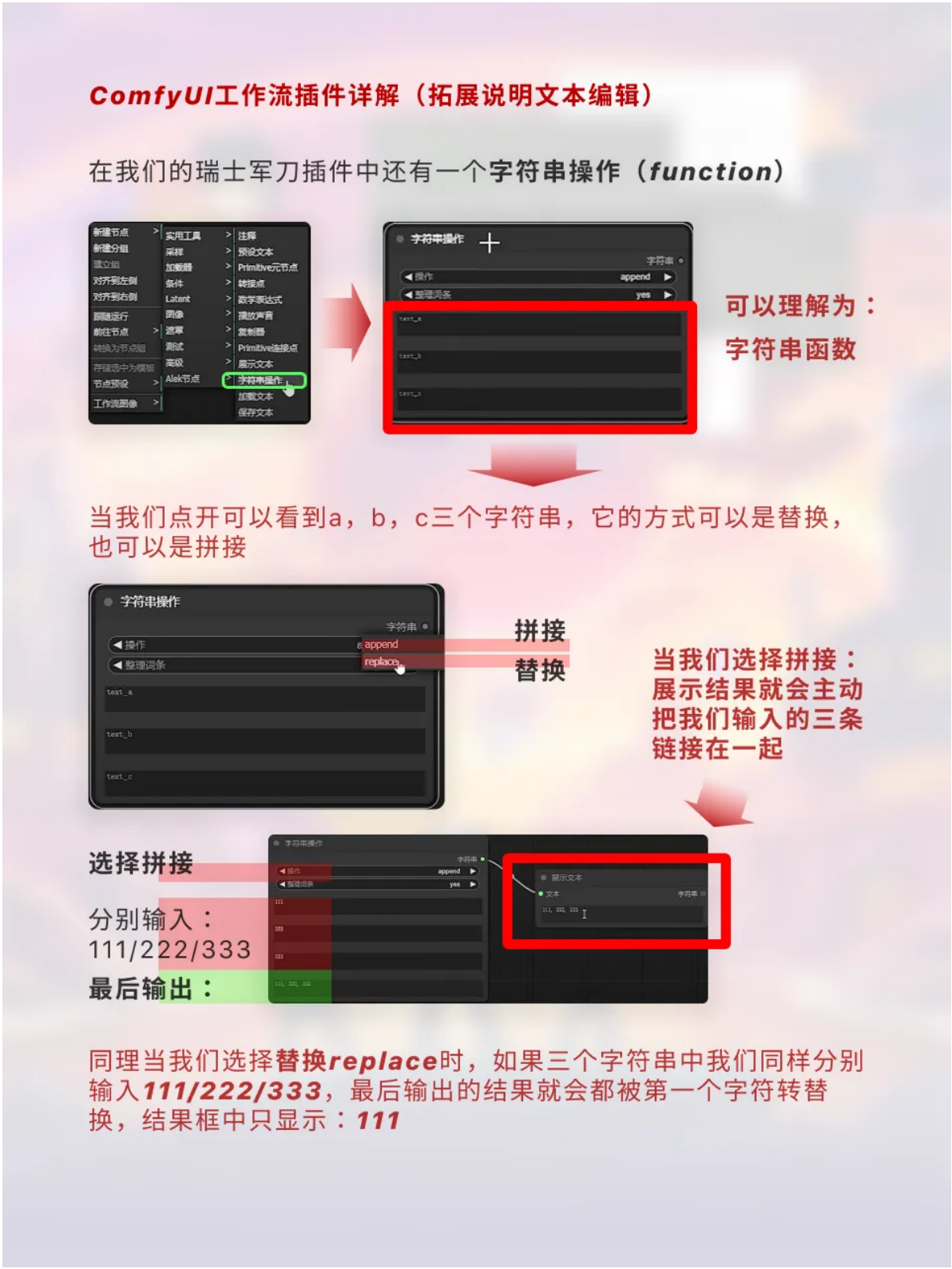
## 1.7 流程总结


这样我们在文本框种输入的同时就编码翻译进去了，如果想查看原英文只需要再加一个展示文本即可；展示文本和预览文本功能相同；这就是我们一个翻译文本的基本运用；

Tips：当我们不需要某个节点的时候就可以ctrl+m停止该节点运行

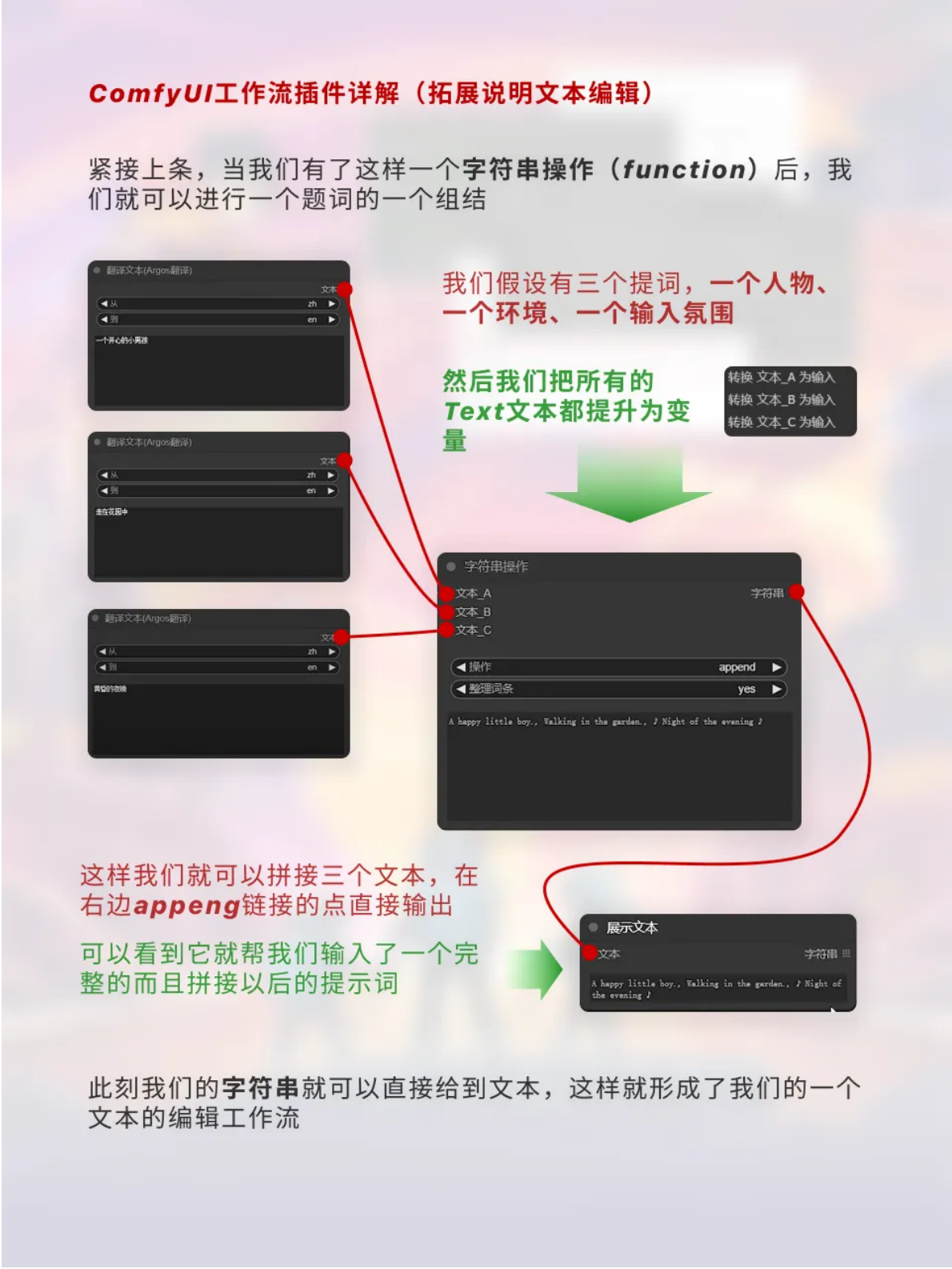
# 二、提词技巧精通

## 1.1 字符串操作（function）




在我们的瑞士军刀插件中还有一个字符串操作（function）；可以看到图片最后当我们选择替换replace时，如果三个字符串中我们同样分别输入111/222/333，最后输出的结果就会都被第一个字符转替换，结果框中只显示：111；

## 1.2 提词组结




紧接上条，当我们有了这样一个字符串操作（function）后，我们就可以进行一个题词的一个组结；

此刻我们的字符串就可以直接给到文本，这样就形成了我们的一个文本的编辑工作流；

## 1.3 补充说明


当然现阶段图片可能有些不美观，因为我们此时的模型还是需要加一些质量描述词/修饰词等，直出图片还是不如SDXL模型人性化

## 1.4 自定义语句


文字补全功能需要点击自定义语句，让它读取下本身的TXT文件，插件就会加载网上读取的默认文字补全；

## 1.5 信息查看插件


包括主模型，右键我们也可以看到查看信息，点击查看信息就会读取civitiai的模型数据内容，也可直接点进civitai看模型详细信息；同样是菜单栏中：我们可以自由选择图像面板位置，按照下面图片依次点击，菜单左边就会出现历史记录；

Tips：当我们需要图像重绘时，也可以直接从历史区域拖拽；也可以在comfyui空白界面通过拖拽图像直接出现该图像的生成工作流

## 1.6 自动排列


当安装了插件后我们在右键子菜单中可以看到Checkpoint加载器（pysss）选项

## 1.7 吸附网络

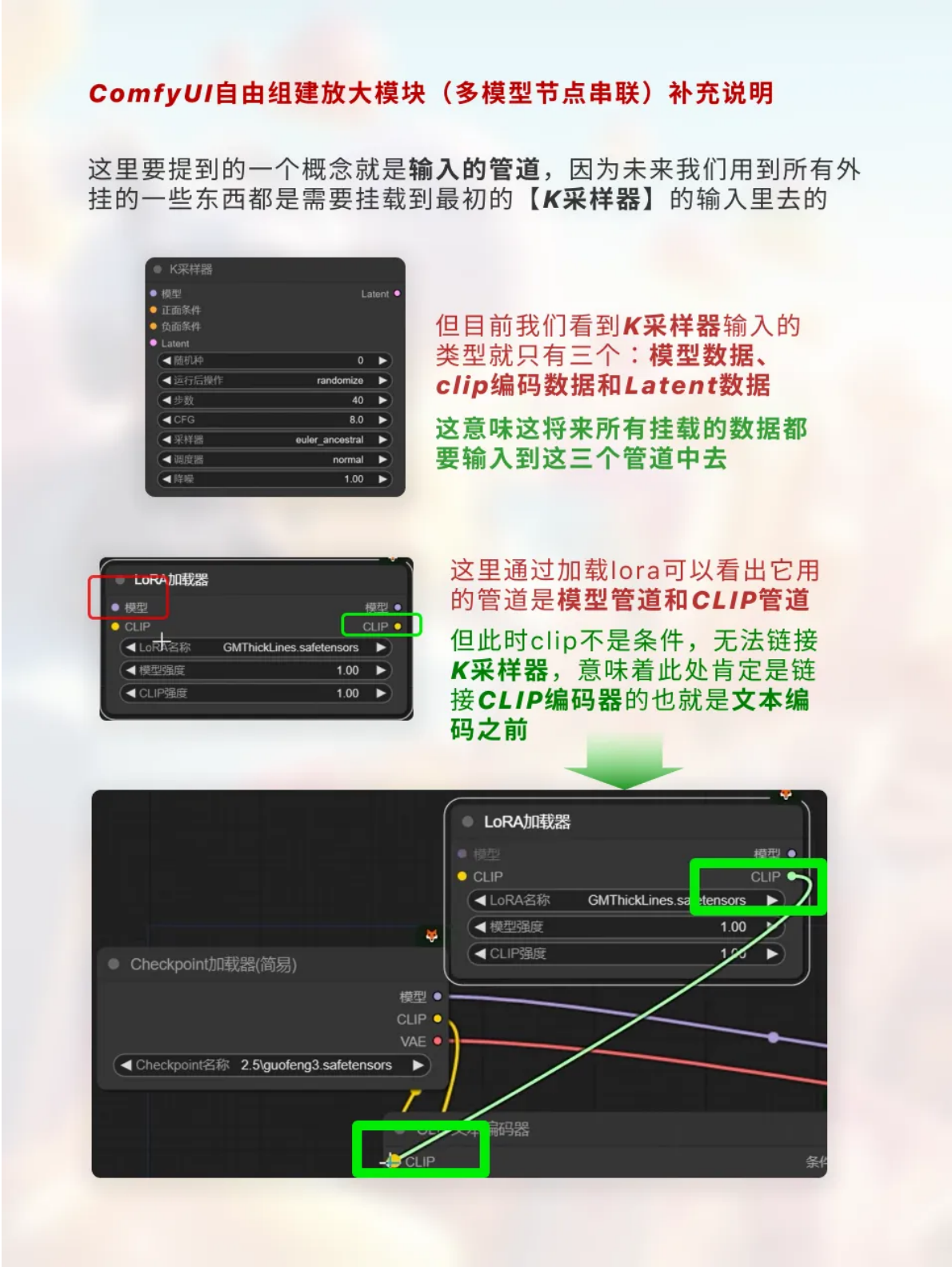

还要介绍的一个功能就是吸附网格，在移动时可以吸附任意点，相当于自动对齐；其余小操作大家也可以自行探索，这里就不过多赘述了，后面在用到时会顺带跟大家展示一下；

## 三、多模型节点串联

## 1.1 Lora加载器

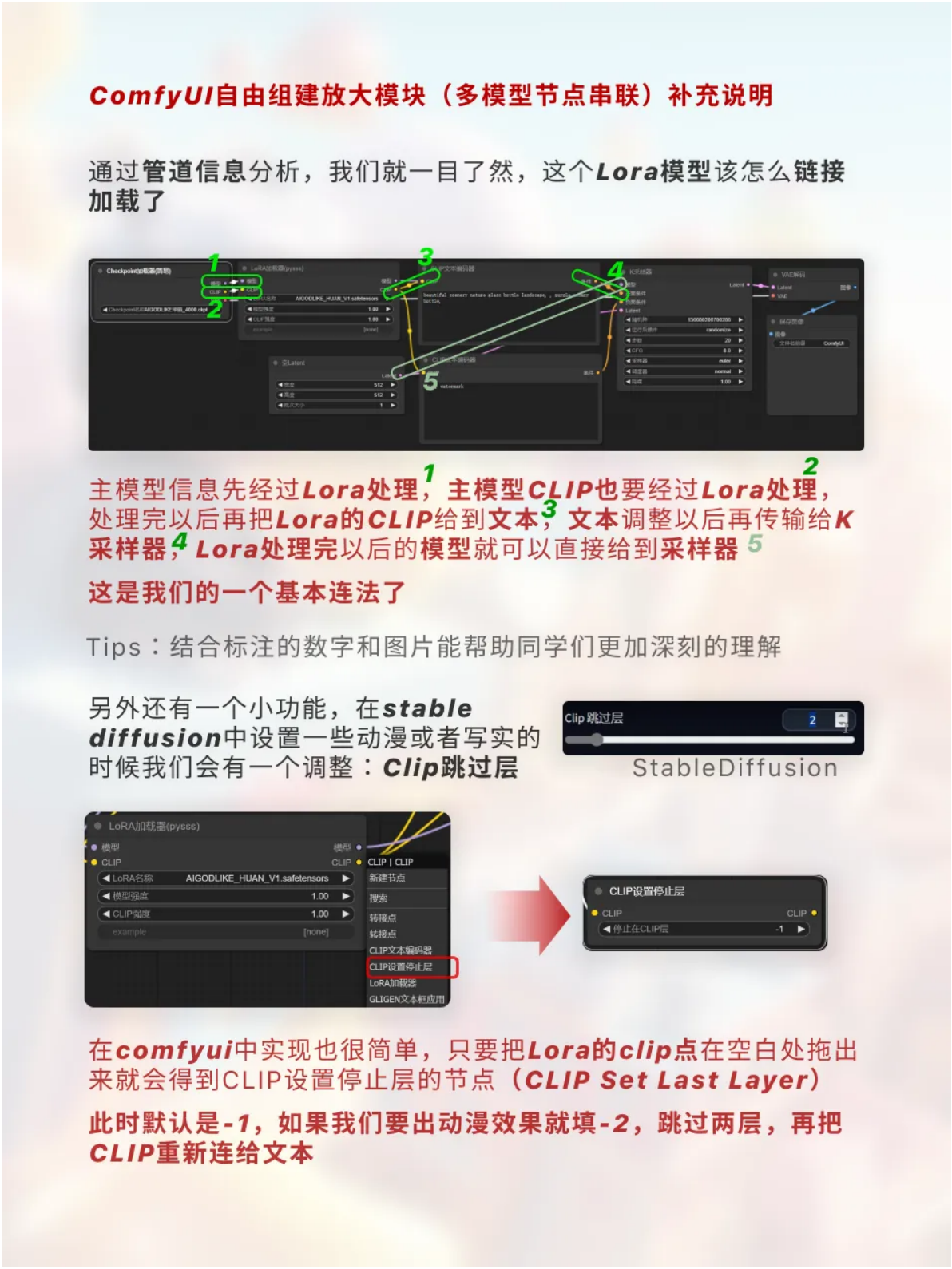

SD中我们常用的一个功能高清修复，其本质就是图片的放大重绘重采样，和我们图生图中是一致的，这一篇讲解的就是如何在comfyUI中的多重放大工作流和Lora模型的节点串联；首先我们依然是加载一个默认工作流，这里我们再来说一个小功能，comfyUI的功能是特别细碎的，有些东西没办法组织成体系，所以我们会在工作流之中参插教给大家；我们在放大工作流上额外实现的一个功能就是Lora模型的一个加载（因为生成图像所以未来效果更好，我们这里添加一个Lora）；通过之前节点输出输入的学习我们现在应给也基本可以判断出Lora（加载器）在当前模型输入中的哪一条线路；

## 1.2 管道输入概念




这里要提到的一个概念就是输入的管道，因为未来我们用到所有外挂的一些东西都是需要挂载到最初的【K采样器】的输入里去的；但目前我们看到K采样器输入的类型就只有三个：模型数据、clip编码数据和Latent数据，这意味这将来所有挂载的数据都要输入到这三个管道中去；这里通过加载lora可以看出它用的管道是模型管道和CLIP管道；但此时clip不是条件，无法链接K采样器，意味着此处肯定是链接CLIP编码器的也就是文本编码之前；

## 1.3 CLIP设置停止层




通过管道信息分析，我们就一目了然，这个Lora模型该怎么链接加载了；主模型信息先经过Lora处理，主模型CLIP也要经过Lora处理，处理完以后再把Lora的CLIP给到文本，文本调整以后再传输给K采样器，Lora处理完以后的模型就可以直接给到采样器，这是我们的一个基本连法了；

在comfyui中实现也很简单，只要把Lora的clip点在空白处拖出来就会得到CLIP设置停止层的节点（CLIP Set Last Layer）此时默认是-1，如果我们要出动漫效果就填-2，跳过两层，再把CLIP重新连给文本；

## 1.4 多模型串联


如果我们还想添加Lora模型，比如一张图需要3-5个Lora模型，我们只需要复制Lora用串接的形式链接；此时我们的CLIP从模型过了四遍最后输出给文本，最后一个模型再输出给到我们的K采样器（不一定是K采样器，但总之任何采样器都是需要模型输入的）；

## 1.5 补充说明


## 补充说明

在Lora加载器中，我们可以看到模型强度和CLIP强度两个参数输出的强度选项；通常我们在webui中两个强度时相关的（一样数值），这样就可以共用一个参数，不需要设置两遍；

这里如果我们想让参数一致就需要用到之前的知识点参数提升变量，再新建元节点，用元节点读取它们的信息，这样一个参数就可以同时控制模型和CLIP两个参数；

# 四、后记

下面一篇会给大家更新放大组件，ComfyUI的工作流并不唯一，没有一个答案是固定死的，没有完全觉得的答案也没有完全绝对的工作流，一切都靠我们思路的发散。

# 五、ComfyUI遮罩修改重绘/Inpenting模块详解

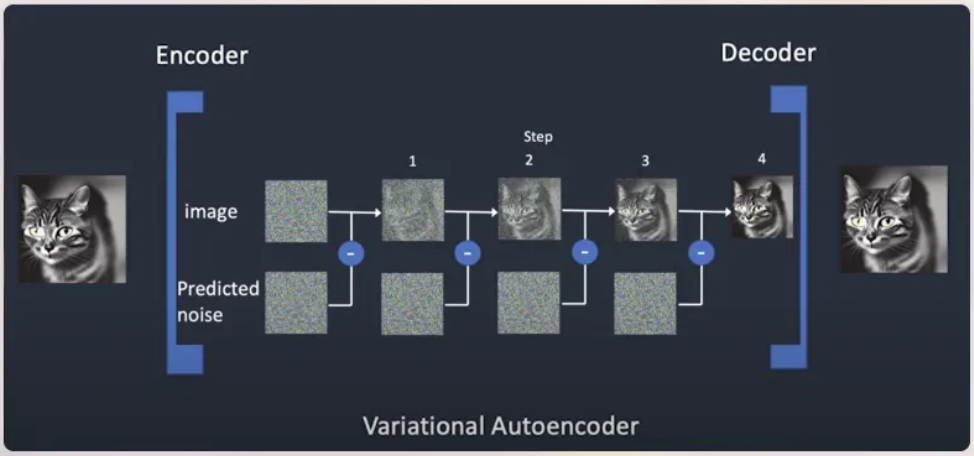
## 前言

紧接上一篇说完我们的就说到inpainting模块了，和之前的流程略有不同，那么重绘的逻辑呢自然是在我们已有的图片上，已有的图片和我们放大也是一个逻辑一样有两种方法，一种就是浅空间里，一种就是浅空间外

### 1.1 SD图片放大逻辑

**ComfyUI工作流裁剪到重绘-图生图完整搭建（inpainting模块）**

紧接上一篇说完我们的就说到**inpainting模块**了，和之前的流程略有不同，那么重绘的逻辑呢自然是在我们已有的图片上，已有的图片和我们放大也是一个逻辑一样有两种方法，**一种就是浅空间里，一种就是浅空间外**




通常在**ComfyUI**中我们的优势就是能**对浅空间进行修改**，而**SD**就相对死板，没有办法对浅空间进行更多细致的操作；

**浅空间的好处就是不需要编码，不需要对图像进行多次的转换**

（转入转出的来回转换）

**第二个好处就是因为同处浅空间，多次生成后的图片和之前的原图混合度会更好，这是浅空间最大的一个优势**

**浅空间的缺点就是无法依次呈现的太丰富细节和太大的图像，因为它永远是在GPU的内存中运算**

Tips：这就是为什么浅空间不能放太大

通常在ComfyUI中我们的优势就是能对浅空间进行修改，而SD就相对死板，没有办法对浅空间进行更多细致的操作；浅空间的好处就是不需要编码，不需要对图像进行多次的转换（转入转出的来回转换）

第二个好处就是因为同处浅空间，多次生成后的图片和之前的原图混合度会更好，这是浅空间最大的一个优势；

## 1.2 浅空间之外图像加载


我们会发现我们是没有办法把解码的图像加载过来的（想要加载过来我们在一些插件里是需要使用一个循环流的，涉及第三方插件，这里就不过多赘述）；加载以后我们同样需要进行一个浅空间转换，一样是进行一个编码采样；

## 1.3 遮罩绘制


紧接上篇，唯一不同的就是这个地方我们需要进行一个重绘（比如我们想替换背景）就需要在遮罩中绘制一下；这里我们来简单的把背景画一下，退出后就可以直接看到我们遮罩的画的范围；

## 1.4 VAE内补编码器


上面可以看到我们的遮罩并不精细，这里我要用到的就不是编码了，我们拖出来新的节点，选择VAE内补编码器；它其实是对于一个遮罩部分的一个编码，它会识别遮罩；与我们原本VAE区别就是多了一个遮罩信息点；我们继续删除之前VAE编码器，将VAE连接过来，右边的Latent直接给到采样器（这里就相当于inpainting二次采样）；

## 1.5 提词应对场景


提示词我们可以直接套用负面提词，正向提词如果想改变可以单独出一个CLIP；点击生成后我们会看到进行到这一步出来的结果是比较糟糕的；最大的问题是出在我们的文本（大部分朋友都会出现的问题就是出现两个头）；

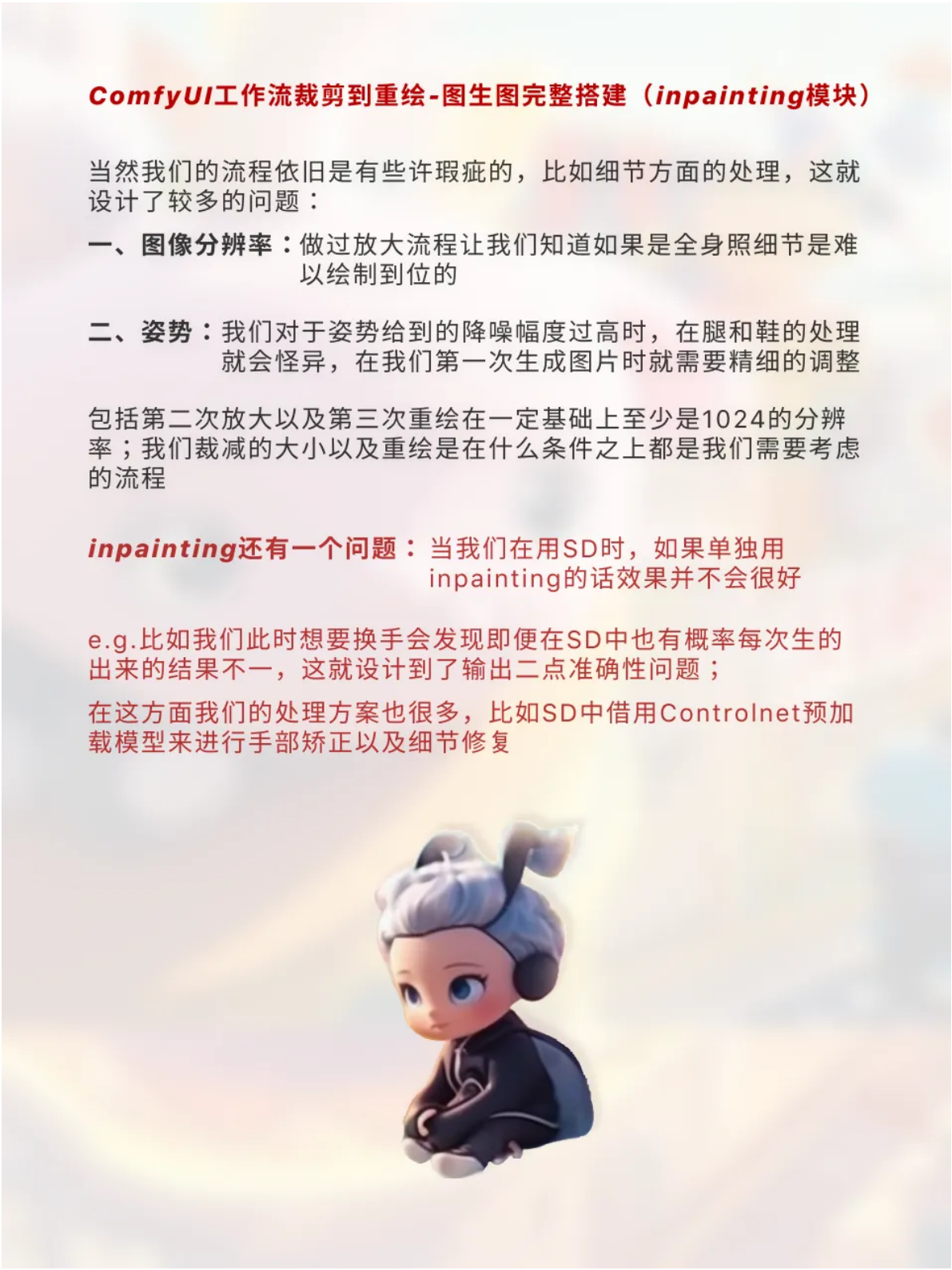
## 1.6 提词应对场景（补充说明）


紧接上一篇，因此对遮罩内要修改的东西，我们就应该只填写背景的内容，删除小男孩的内容，而不是整张画的描述；再次生成我们就可以发现背景成功被我们替换成了所设置的天空；这就是我们在潜在空间之外现有图像的一个转换，相当于图像Pixel信息再转换给了潜在空间，然后重新出图；

## 1.7 潜在空间之外


这里我们重新绘制一个遮罩，然后节点拖出来选择设置Latent噪波遮罩；这个意思就是潜在空间的遮罩会被遮罩的地方重新生成罩波，也就是重新进行二次采样；与之前的区别就是这次是在laten空间里进行一个转换；接下来就相对简单，我们只要依次点对点连接（结合我们的提示词，把上一个采样器的latent直接输出过来，再把加了遮罩的latent直接输入第二个采样器）

## 1.8 细节补充说明




补充说明

当然我们的流程依旧是有些许瑕疵的，比如细节方面的处理，这就设计了较多的问题：

# 一、图像分辨率：

做过放大流程让我们知道如果是全身照细节是难以绘制到位的

# 二、姿势：

我们对于姿势给到的降噪幅度过高时，在腿和鞋的处理就会怪异，在我们第一次生成图片时就需要精细的调整；

包括第二次放大以及第三次重绘在一定基础上至少是1024的分辨率；我们裁减的大小以及重绘是在什么条件之上都是我们需要考虑的流程；

## 结尾

Inpenting模块中在我们想要换手会发现即便在SD中也有概率每次生的出来的结果不一，这就设计到了输出二点准确性问题；在这方面我们的处理方案也很多，比如SD中借用Controlnet预加载模型来进行手部矫正以及细节修复；更多的方案就需要大家自行探索了！

感谢大家的关注，持续分享ComfyUI的内容教学！

# 六、ComfyUI超实用SDXL工作流手把手搭建

## 前言

Stable Diffusion XL 是 Stable Diffusion v1.5 的进阶版本，它与之前的版本相比，主要有以下几个特点：

- 能生成更高质量的图片
- 图片的细节更丰富
- 能理解更复杂的 prompt
- 支持生成更大尺寸的图片

这一期我们要分享的是SDXL完整工作流，说到SDXL和我们之前的1.5模型是最大的区别就是体量的不同，造成整个生成过程当中有两个方面需要考虑和调节；

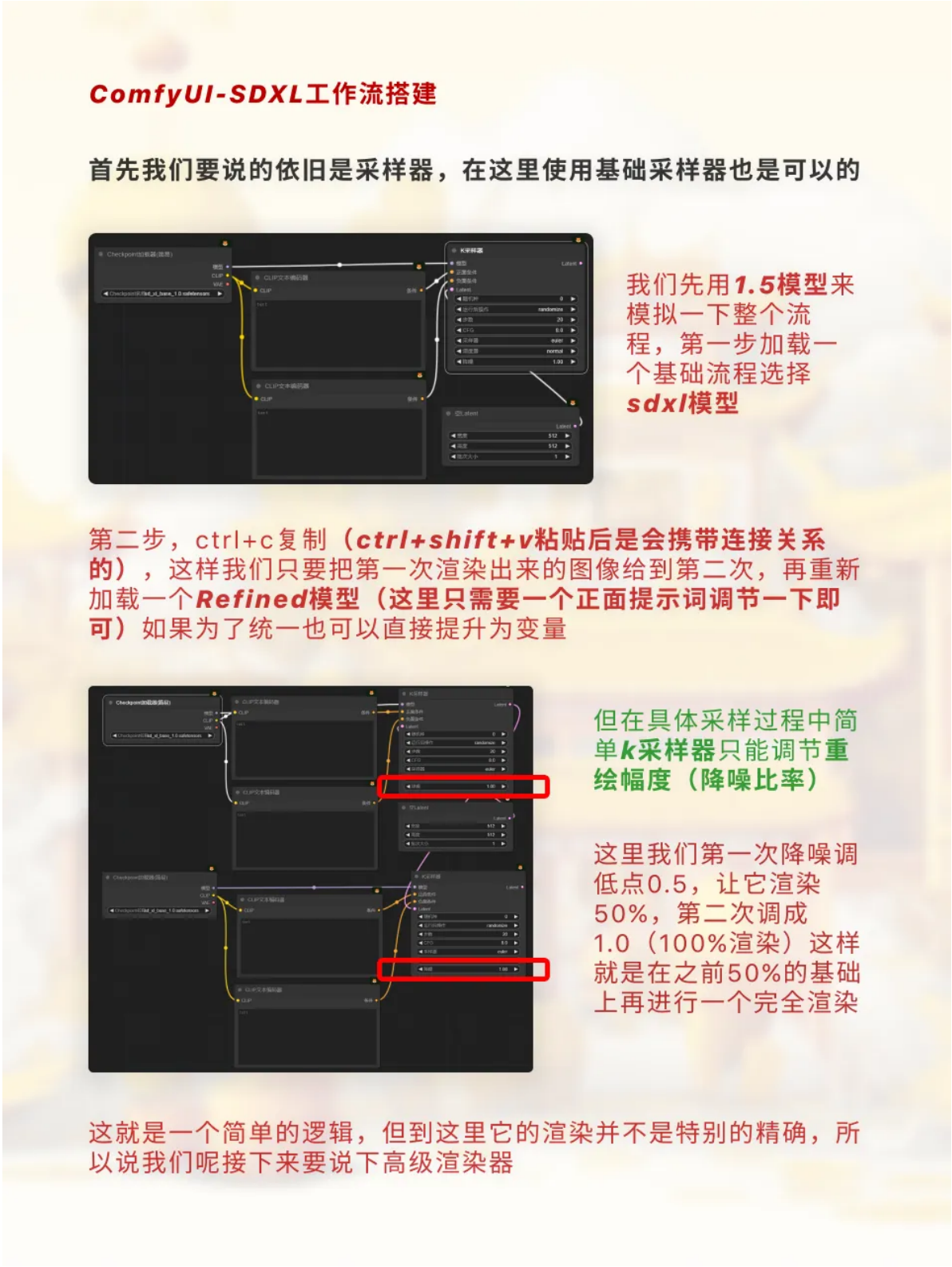
## 一、Refined模型

### 1.1 基础介绍


基础介绍

- 标准的SDXL有两个模型，一个base模型一个Refined模型，需要设计到双重采样或者二次采样；
- 因为整个空间（CLIP空间）不一样，所以在提词方面可以做出很细微的切分和分类

## 1.2 采样器




采样器

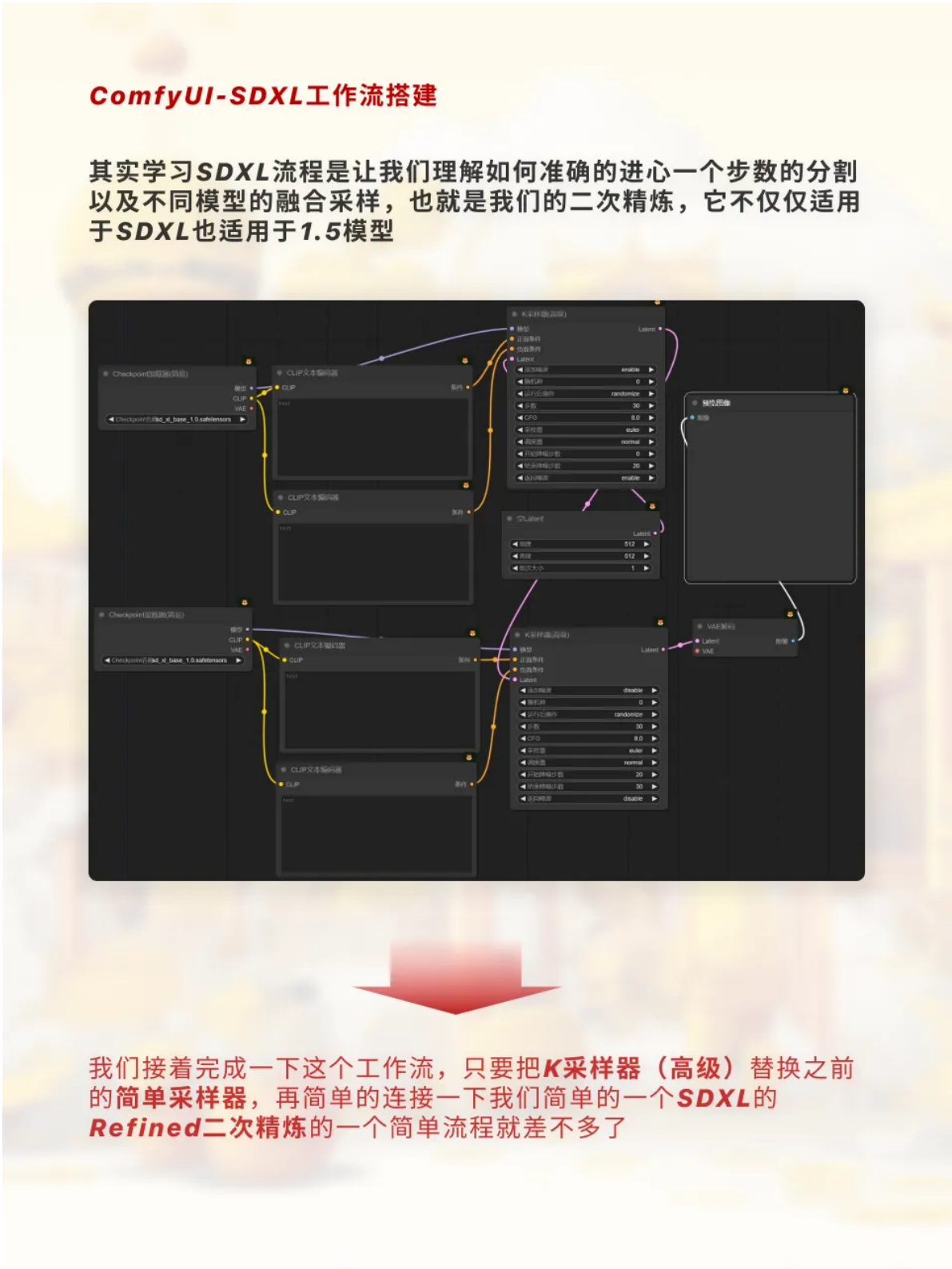
- 第二步，ctrl+c复制（ctrl+shift+v粘贴后是会携带连接关系的），这样我们只要把第一次渲染出来的图像给到第二次，再重新加载一个Refined模型（这里只需要一个正面提示词调节一下即可）如果为了统一也可以直接提升为变量
- 这就是一个简单的逻辑，但到这里它的渲染并不是特别的精确，所以说我们呢接下来要说下高级渲染器

## 1.3 噪波添加


噪波添加

- K采样器（高级）是比较精确的一个分布渲染
- 噪波是默认添加的（如果要二次渲染则不要添加）
- 这里我们复制一份，左边是第一次渲染，右边是第二次渲染，就是连续两次渲染的过程

## 1.4 采样器替换




采样器替换

- 其实学习SDXL流程是让我们理解如何准确的进心一个步数的分割以及不同模型的融合采样，也就是我们的二次精炼，它不仅仅适用于SDXL也适用于1.5模型
- 我们接着完成一下这个工作流，只要把K采样器（高级）替换之前的简单采样器，再简单的连接一下我们简单的一个SDXL的Refined二次精炼的一个简单流程就差不多了

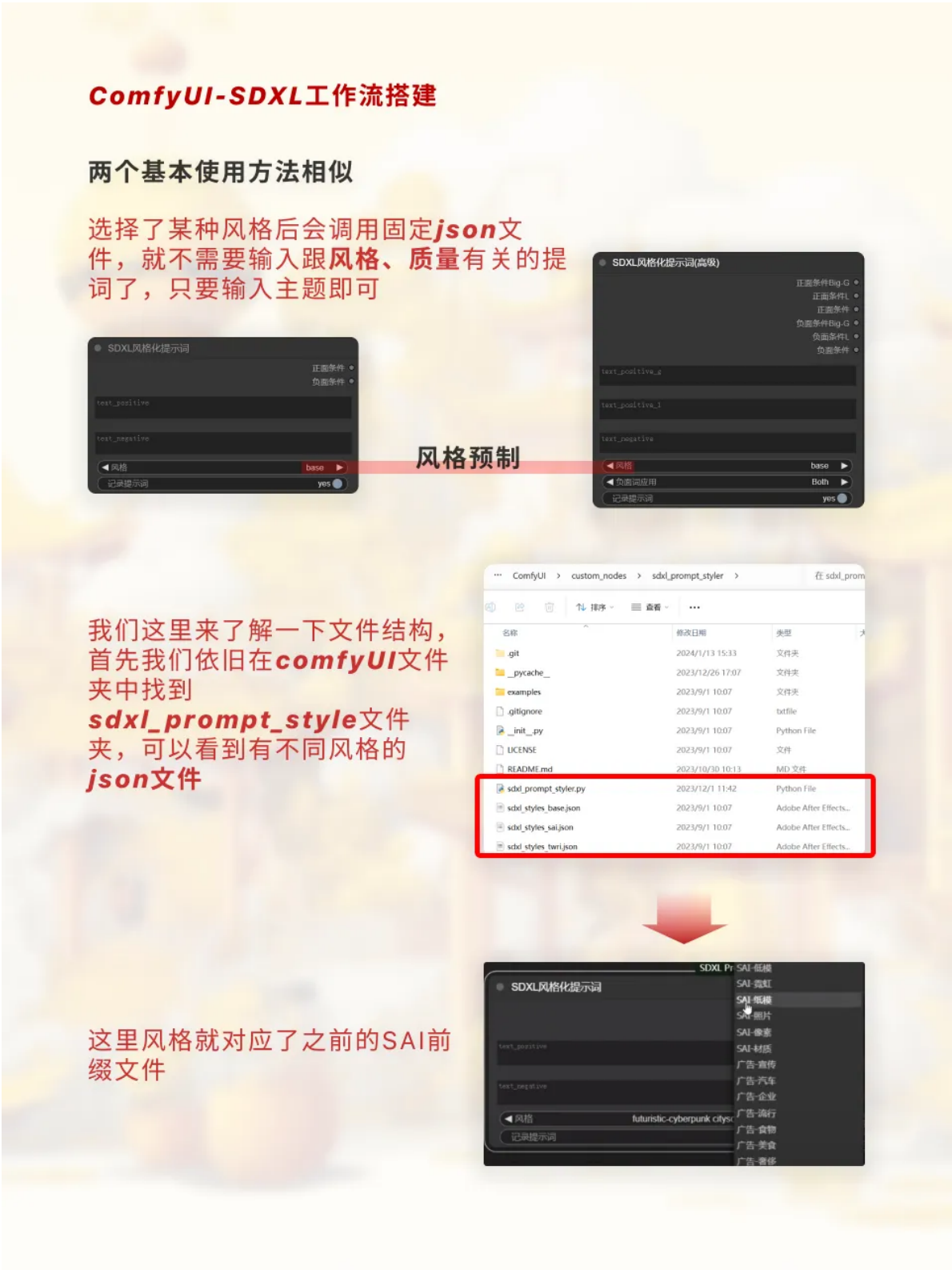
# 二、SDXL风格化提示词

## 2.1 SDXL Prompt Styler插件


SDXL Prompt Styler插件

- 下面我们要说的就是对于提词而言我们需要有个什么样的加工方式，但在这之前要继续跟大家分享一个风格化插件（外部UI里提供现成风格）
- 在ComfyUI中也有这门一个插件，搜索SDXL或者styler
- 此时我们会看到两个Styler，第一个是我们标准的SDXL Prompt Styler，第二个是拓展版本，会比第一个多一些小功能，但基本道理和使用方法都相同，这里我们直接安装即可

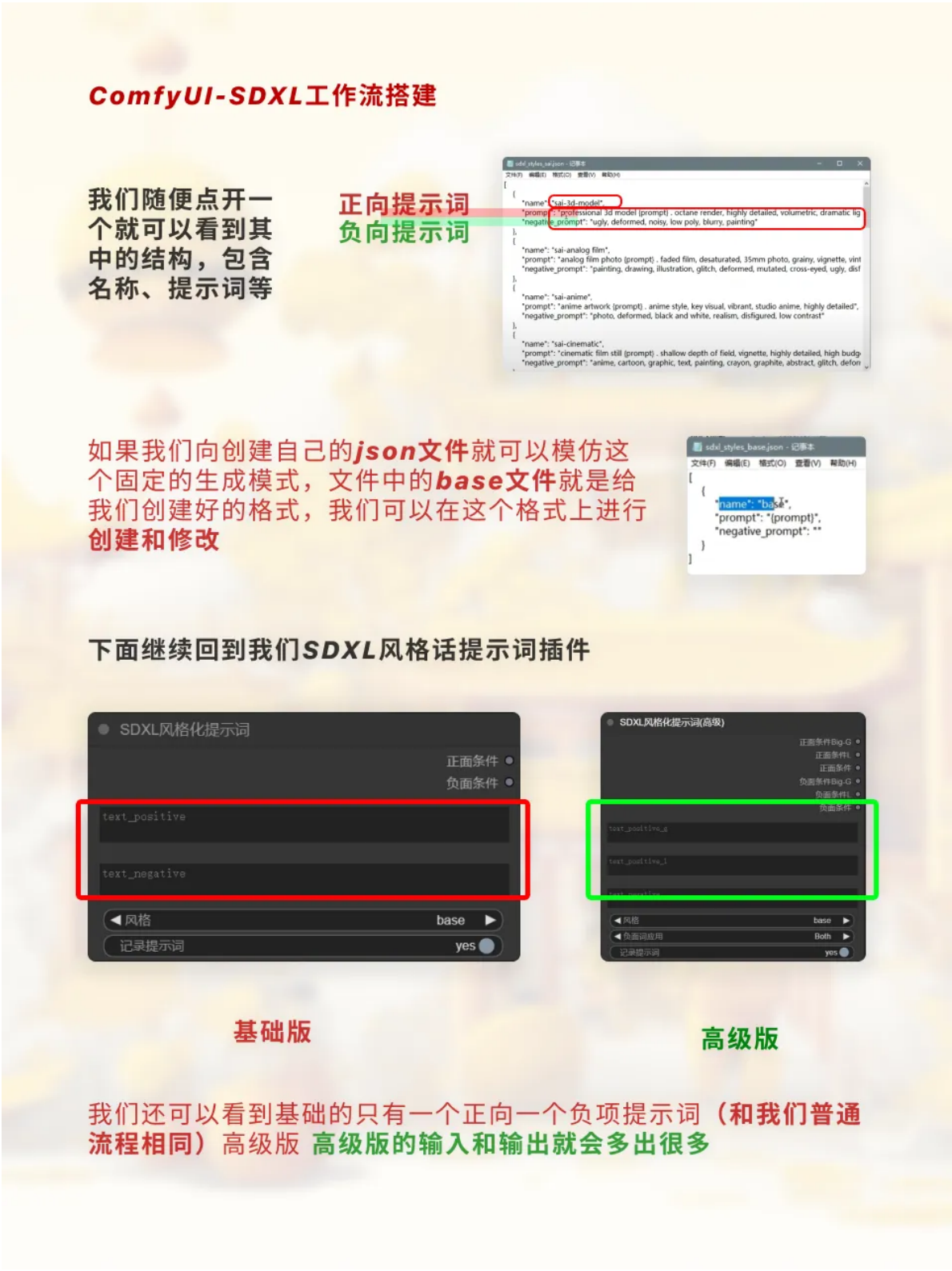
## 2.2 json文件结构




json文件结构

- 选择了某种风格后会调用固定json文件，就不需要输入跟风格、质量有关的提词了，只要输入主题即可
- 我们这里来了解一下文件结构，首先我们依旧在comfyUI文件夹中找到sdxl_prompt_style文件夹，可以看到有不同风格的json文件

## 2.3 SDXL风格化提示词插件




SDXL风格化提示词插件

- 我们随便点开一个就可以看到其中的结构，包含名称、提示词等
- 如果我们向创建自己的json文件就可以模仿这个固定的生成模式，文件中的base文件就是给我们创建好的格式，我们可以在这个格式上进行创建和修改
- 我们还可以看到基础的只有一个正向一个负项提示词（和我们普通流程相同）高级版，高级版的输入和输出就会多出很多

## 2.4 SDXL提词结构


SDXL提词结构

- 关于高级版这里我们就涉及到一个SDXL的提词结构了
- 我们先看向提词G和L还有负面条件G和负面条件L，它们可以清晰的帮我们分出画面中想要的东西从而细致的调节一些东西
- 先说正面输出的G和L：

  L相当于我们的Tag模式（我们先要确认图像中有什么比如一个人、一棵树）主要形容的是名词，就是一个Tag需要详细描述的概念

G则更适合详细描述概念，可以简单的理解，L是名词的话，G就是形容词（L如果是一个女生，G就是形容这个女生的五官发型等；如果L是牛仔裤，G就是牛仔裤颜色等）

G更适合形容概念

- 这里我们先生成一张图像，可以看到其中有车、房屋、人这些就是画面中的tag，当我们需要具体形容样式颜色就需要用到SDXL风格化提示词（高级）

## 2.5 高级工作流实现


高级工作流实现

- 对于负面提示词来说道理相同，但我们通常不会特别清晰描述负面提示词，大多数是沿用原先的负面，或者根据画面调整排除我们不想要的内容

- 当我们尝试把正面/负面条件接入时会发现无法接入，这里是因为翻译问题，转为因为会发现显示的是text_positive，也就是说是文本，只不过是文本形式的正向条件，还没进行CLIP编码
- 我们需要处理的就是把CLIP文本编码器提升为变量，这时候就可以正常连接了到这里我们的工作流其实就基本完成了，下一篇会加入风格化插件

# 三、SDXL工作流搭建

## 3.1 风格化插件替换


风格化插件替换

- 这里我们用很简单的提词就能得到不错的图像，同样是一艘船一个是带了风格的，一个是没带风格的，差距就是额这么明显，下图是选用了电影风格

- 接着我们说到SDXL风格化提示词(高级)流程

## 3.2 SDXL风格化提示词(高级)流程


SDXL风格化提示词(高级)流程

- 既然我们采样器这里已经用了高级了，那么我们就把流程整体升级一下
- 我们先删除原来的文本编码，然后替换成高级的，这里我们可以考到两个针对SDXL的CLIP文本编码
- CLIP其实也算是视觉编码的一个空间，CLIP空间是实际图像分辨率2-4倍

## 3.3 数学表达式


数学表达式

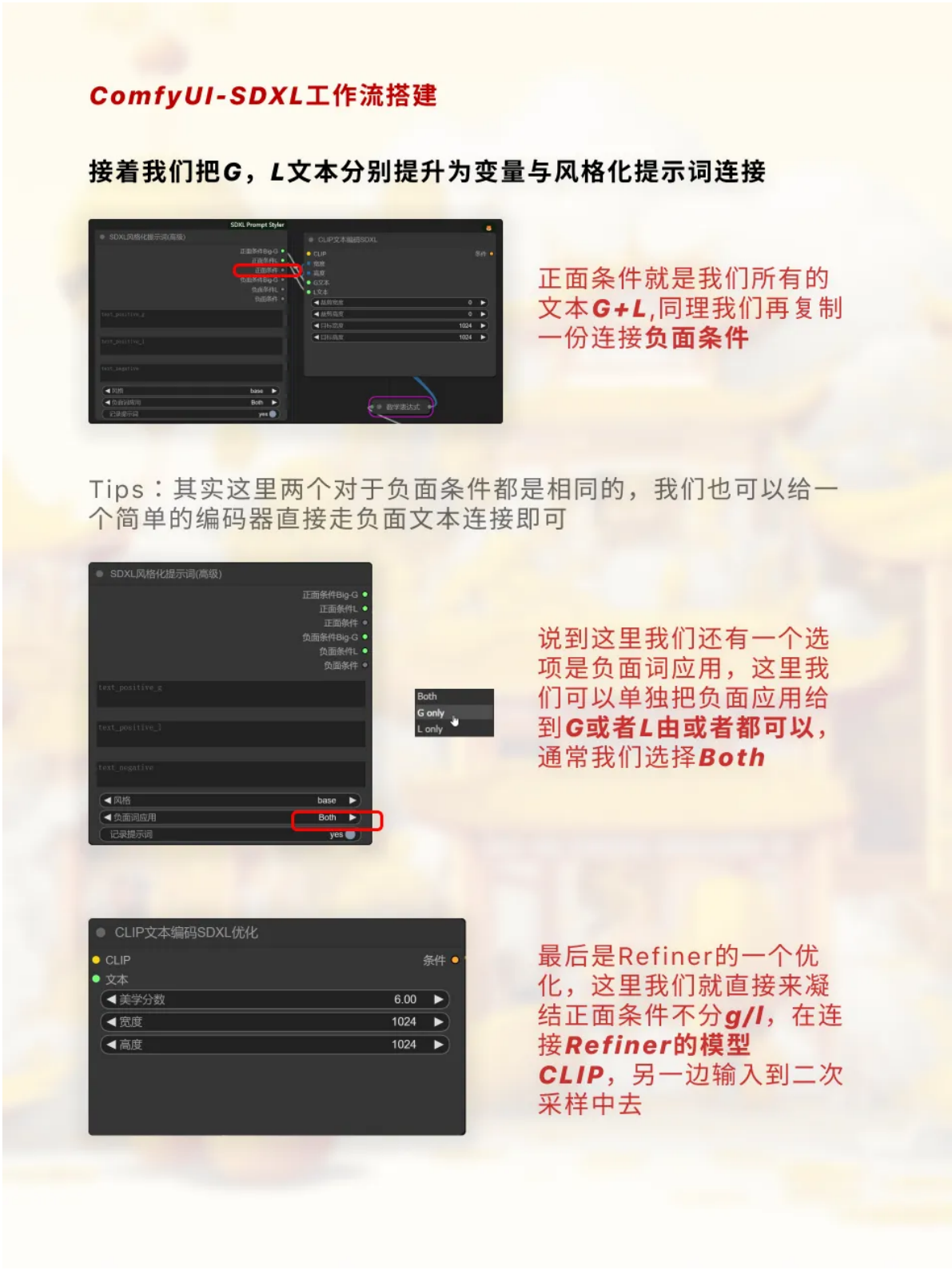
- 我们延展的来做一个节点运算，因为我们要根据Latent数字来进行倍数放大，首先从空Latent中拖出一个数学表达式节点
- 这里我们就可以把Latent数据传输到表达式中，在表达式中可以根据数据取它的宽度
- 例如我们取它的宽度（1024），在当中填写4倍：a.width*4取出的值就是4096
- 这里算是ComfyUI相关底层的知识了，我们只需要大概了解皆可，大部分过程中是很少涉及的，和程序有点相似，因为节点就是一个半开发的框架，为了大家更多的去理解，方便未来去流程化处理

## 3.4 数学表达式数据传递使用


数学表达式数据传递使用

- 这里我们直接把表达式的整数传给CLIP文本编码即可，这样我们就定义了CLIP文本本身的一个参数，我们乘2乘4都可以任意
- 这里目标宽高度，我们就可以以同样的方式传递给采样器，当然也可以默认1024省事的处理下，之后再遇到全流程自动化再规范逻辑

# 3.5 变量与风格化提示词连接




变量与风格化提示词连接

- 正面条件就是我们所有的文本G+L,同理我们再复制一份连接负面条件
- 其实这里两个对于负面条件都是相同的，我们也可以给一个简单的编码器直接走负面文本连接即可
- 说到这里我们还有一个选项是负面词应用，这里我们可以单独把负面应用给到G或者L由或者都可以，通常我们选择Both
- 最后是Refiner的一个优化，这里我们就直接来凝结正面条件不分g/l，在连接Refiner的模型CLIP，另一边输入到二次采样中去

## 3.6 完整SDXL高级工作流


完整SDXL高级工作流

- 到这里我们SDXL的一套高级工作流基本完成了且都采取了高级节点和详细定制

## 3.7 SDXL工作流整合


SDXL工作流整合

- 我们把G和L提升为变量，并添加了翻译插件，进行了一个单独节点的翻译和输入
- 最后采样给了两个统一的变量，一个步数变量，一个转换变量，因为采样器的步数和转换步数都是一致共用的，只不过连接的接口不一样

# 后记

这就是我们对SDXL整个模型所需要的一些完整设置，以及它所涉及的相关内容讲解。

这部分内容是想让大家明白整个工作流该如何构建，理清思路和逻辑。

大家也可以积极尝试去让工作流有更多的一些变化。