您面临的情况是深度学习环境中非常**典型**的多版本兼容性问题。

您完全可以在系统已经有 **CUDA 12.3** 的情况下，**安装并配置 CUDA 11.5**，而无需卸载现有版本。TensorFlow 2.10 要求 CUDA Toolkit **11.2 到 11.8**，其中 **11.5** 是一个兼容的选择。

以下是从零配置并切换到目标 CUDA 11.5 版本的详细步骤：

1. 🔍 确认兼容性要求

TensorFlow 2.10 的官方要求如下：

- **CUDA Toolkit: 11.2 - 11.8** (推荐 11.2, 11.8 或 11.5)
- **cuDNN: 8.1 - 8.6** (推荐 8.1 或 8.6)

我们选择 **CUDA 11.5** 和相应的 **cuDNN 8.3** 版本。

---

2. ⬇️ 安装旧版 CUDA Toolkit (11.5)

步骤 2.1：下载 CUDA 11.5

访问 NVIDIA 的 CUDA Toolkit Archive，找到 **CUDA Toolkit 11.5 Update 2**（或任何 11.5 版本），选择您的操作系统并下载安装程序。

步骤 2.2：安装

在安装时，请注意以下关键点：

- **Linux/Windows：** **取消** 勾选安装 **NVIDIA 驱动程序**。由于您的系统已经有 CUDA 12.3 对应的新驱动（通常兼容旧版 CUDA），我们只需要安装 Toolkit 本身。
- **安装路径：** 允许其安装到默认路径，例如在 Linux 上是 `/usr/local/cuda-11.5`，在 Windows 上是 `C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.5`。

步骤 2.3：下载 cuDNN 8.3

访问 cuDNN Archive，找到与 **CUDA 11.5** 兼容的 **cuDNN v8.3.x** 版本并下载（通常你需要注册开发者账号）。

步骤 2.4：配置 cuDNN

将下载的 cuDNN 压缩包解压，并将其中的文件（`bin`, `include`, `lib` 文件夹下的内容）**复制** 到您刚才安装的 **CUDA 11.5** 路径下（即 `/usr/local/cuda-11.5` 或 `...\CUDA\v11.5`）。

---

3. 🐍 配置独立的 Python 虚拟环境

为了确保 TensorFlow 2.10 只使用 CUDA 11.5，我们必须在虚拟环境中进行隔离。

步骤 3.1：创建并激活环境

Bash

```
# 使用 conda 创建虚拟环境（推荐）
conda create -n tf210_gpu python=3.9
conda activate tf210_gpu
```

步骤 3.2：安装 TensorFlow

在激活的环境中，安装 TensorFlow 2.10：

Bash

```
pip install tensorflow==2.10
```

---

4. ⚙️ 环境变量设置：指向 CUDA 11.5

这是最重要的一步，它告诉 TensorFlow 在哪里找到正确的 CUDA 版本。您需要将 **CUDA 11.5** 的路径添加到环境变量中。

对于 Linux / macOS (推荐方法)

在激活的虚拟环境中，设置环境变量：

Bash

```
# 确保在 tf210_gpu 环境中运行
export PATH=/usr/local/cuda-11.5/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda-11.5/lib64:$LD_LIBRARY_PATH
```

> **提示：** 如果您使用的是 `/usr/local/cuda` 软链接进行全局切换，您可以将软链接从 `cuda-12.3` 指向 `cuda-11.5`，但这会影响系统上其他使用 12.3 的程序。**推荐只在虚拟环境中设置上述临时变量。**

对于 Windows

您可能需要调整系统环境变量或在运行脚本中设置临时的环境变量，指向 `C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.5`。

- `PATH` 变量中，确保 `...\CUDA\v11.5\bin` 路径排在 `...\CUDA\v12.3\bin` 之前。

---

5. ✅ 验证和测试

在激活了虚拟环境并设置了环境变量后，运行以下 Python 代码验证：

Python

```
import tensorflow as tf
```

```
# 检查 TensorFlow 是否识别 GPU
gpus = tf.config.experimental.list_physical_devices('GPU')
if gpus:
    print(f"GPUs detected: {len(gpus)}")
    # 尝试运行一个简单的张量操作
    a = tf.constant([[1.0, 2.0], [3.0, 4.0]])
    b = tf.constant([[1.0, 1.0], [0.0, 1.0]])
    c = tf.matmul(a, b)
    print(c.numpy())
else:
    print("No GPU detected.")

# 检查使用的 CUDA 版本（通过查看加载的库）
# 如果一切配置正确，TensorFlow 2.10 将会使用 CUDA 11.5
```

如果代码成功运行并显示检测到 GPU，则说明您已成功地在 **CUDA 12.3** 系统上，为 **TensorFlow 2.10** 配置了 **CUDA 11.5** 环境。

下一步： 既然您在学习深度学习，想了解一些关于 **TensorFlow 2.x 模型保存和加载**的最佳实践吗？