

## NVIDIA TAO Toolkit 学习摘要

TAO (Train, Adapt, Optimize) Toolkit 是 NVIDIA 推出的低代码 AI 工具链, 基于迁移学习和模型优化技术, 帮助开发者快速构建、微调并优化适用于边缘设备的轻量级模型。

```
yuerm@LAPTOP-5MK05CJF: ~$ git clone https://github.com/NVIDIA/tao_tutorials.git
Cloning into 'tao_tutorials'...
remote: Enumerating objects: 1139, done.
remote: Counting objects: 100% (25/25), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 1139 (delta 19), reused 18 (delta 18), pack-reused 1114 (from 1)
Receiving objects: 100% (1139/1139), 7.18 MiB | 164.00 KiB/s, done.
Resolving deltas: 100% (569/569), done.
yuerm@LAPTOP-5MK05CJF: ~$ cd tao_tutorials
```

图 1: 手动下载 TAO 包

获取 NGC API KEY:

```
$ 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
```

```
yuerm@LAPTOP-5MK05CJF: ~$ docker login nvcr.io
Username: $oauthtoken
Password:
Login Succeeded
yuerm@LAPTOP-5MK05CJF: ~$
```

图 2: 登录成功

```
yuerm@LAPTOP-5MK05CJF: ~$ python -m ipykernel install --user --name launcher --display-name "launcher"
Installing collected packages: wcwidth, pure-eval, ptyprocess, typing_extensions, traitlets, tornado, six, p
yzmq, pygments, psutil, prompt_toolkit, platformdirs, pexpect, parso, packaging, nest_asyncio, executing, de
corator, debugpy, asttokens, stack_data, python-dateutil, matplotlib-inline, jupyter-core, jedi, ipython-py
gments-lexer, comm, jupyter-client, ipython, ipykernel
Successfully installed asttokens-3.0.0 comm-0.2.2 debugpy-1.8.14 decorator-5.2.1 executing-2.2.0 ipykernel-6
.29.5 ipython-9.3.0 ipython-pygments-lexer-1.1.1 jedi-0.19.2 jupyter-client-8.6.3 jupyter-core-5.8.1 matplo
tlib-inline-0.1.7 nest-asyncio-1.6.0 packaging-25.0 parso-0.8.4 pexpect-4.9.0 platformdirs-4.3.8 prompt_tool
kit-3.0.51 psutil-7.0.0 ptyprocess-0.7.0 pure-eval-0.2.3 pygments-2.19.2 python-dateutil-2.9.0.post0 pyzmq-2
7.0.0 six-1.17.0 stack_data-0.6.3 tornado-6.5.1 traitlets-5.14.3 typing_extensions-4.14.0 wcwidth-0.2.13
(launcher) yuerm@LAPTOP-5MK05CJF: ~$ python -m ipykernel install --user --name launcher --display-name "laun
cher"
Installed kernelspec launcher in /home/yuerm/.local/share/jupyter/kernels/launcher
(launcher) yuerm@LAPTOP-5MK05CJF: ~$
```

图 3: 设置 Python 环境

```
yuerm@LAPTOP-5MK05CJF: ~$ sudo dpkg -i libnvidia-container1_1.17.8-1_amd64.deb
Selecting previously unselected package libnvidia-container1:amd64.
(Reading database ... 75183 files and directories currently installed.)
Preparing to unpack libnvidia-container1_1.17.8-1_amd64.deb ...
Unpacking libnvidia-container1:amd64 (1.17.8-1) ...
Setting up libnvidia-container1:amd64 (1.17.8-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.18) ...
yuerm@LAPTOP-5MK05CJF: ~$ sudo dpkg -i libnvidia-container-dev_1.17.8-1_amd64.deb
(Reading database ... 75191 files and directories currently installed.)
Preparing to unpack libnvidia-container-dev_1.17.8-1_amd64.deb ...
Unpacking libnvidia-container-dev:amd64 (1.17.8-1) over (1.17.8-1) ...
Setting up libnvidia-container-dev:amd64 (1.17.8-1) ...
yuerm@LAPTOP-5MK05CJF: ~$ dpkg -l | grep libnvidia-container
ii libnvidia-container-dev:amd64 1.17.8-1 amd64 NVIDIA container runtime library
(development files)
ii libnvidia-container1:amd64 1.17.8-1 amd64 NVIDIA container runtime library
```

图 4: 手动下载和安装 nvidia container

```
yuermv@LAPTOP-5MK05CJF: ~$ sudo dpkg -i nvidia-container-toolkit-base_1.17.8-1_amd64.deb
(Reading database ... 75197 files and directories currently installed.)
Preparing to unpack nvidia-container-toolkit-base_1.17.8-1_amd64.deb ...
Unpacking nvidia-container-toolkit-base (1.17.8-1) over (1.17.8-1) ...
Setting up nvidia-container-toolkit-base (1.17.8-1) ...
yuermv@LAPTOP-5MK05CJF: ~$ dpkg -l | grep nvidia-container-toolkit
rc  nvidia-container-toolkit      1.17.8-1      amd64      NVIDIA Container toolkit
ii  nvidia-container-toolkit-base  1.17.8-1      amd64      NVIDIA Container Toolkit Bas
e
yuermv@LAPTOP-5MK05CJF: ~$
```

图 5: 手动下载和安装 nvidia container toolkit

```
yuermv@LAPTOP-5MK05CJF: ~$ dpkg -l | grep libnvidia-container-dev
ii  libnvidia-container-dev:amd64  1.17.8-1      amd64      NVIDIA container runtime library (development files)
yuermv@LAPTOP-5MK05CJF: ~$
```

```
yuermv@LAPTOP-5MK05CJF: ~$ sudo dpkg -i libnvidia-container-tools_1.17.8-1_amd64.deb
Selecting previously unselected package libnvidia-container-tools.
(Reading database ... 75197 files and directories currently installed.)
Preparing to unpack libnvidia-container-tools_1.17.8-1_amd64.deb ...
Unpacking libnvidia-container-tools (1.17.8-1) ...
Setting up libnvidia-container-tools (1.17.8-1) ...
yuermv@LAPTOP-5MK05CJF: ~$ dpkg -l | grep libnvidia-container-tools
ii  libnvidia-container-tools      1.17.8-1      amd64      NVIDIA container runtime library (command-line tools)
yuermv@LAPTOP-5MK05CJF: ~$
```

```
yuermv@LAPTOP-5MK05CJF: ~$ sudo dpkg -i libnvidia-container1-dbg_1.17.8-1_amd64.deb
Selecting previously unselected package libnvidia-container1-dbg:amd64.
(Reading database ... 75202 files and directories currently installed.)
Preparing to unpack libnvidia-container1-dbg_1.17.8-1_amd64.deb ...
Unpacking libnvidia-container1-dbg:amd64 (1.17.8-1) ...
Setting up libnvidia-container1-dbg:amd64 (1.17.8-1) ...
yuermv@LAPTOP-5MK05CJF: ~$ dpkg -l | grep libnvidia-container1-dbg
ii  libnvidia-container1-dbg:amd64  1.17.8-1      amd64      NVIDIA container runtime library (debugging symbols)
yuermv@LAPTOP-5MK05CJF: ~$
```

图 6: 安装 nvidia container 包

```
4. classification_tf1
5. classification_tf2
6. deformable_detr
7. detectnet_v2
8. dino
9. dssd
10. efficientdet_tf1
11. efficientdet_tf2
12. faster_rcnn
13. grounding_dino
14. mask_grounding_dino
15. mask2former
16. lprnet
17. mask_rcnn
18. ml_recog
19. multitask_classification
20. ocdnet
21. ocrnet
22. optical_inspection
23. retinanet
24. segformer
25. ssd
26. trtexec
27. unet
28. yolo_v3
29. yolo_v4
30. yolo_v4_tiny

format_version: 3.0
toolkit_version: 5.5.0
published_date: 08/26/2024
(launcher) yuermv@LAPTOP-5MK05CJF: ~/tao_tutorials$ chmod +x setup/quickstart_launcher.sh
```

图 6: 成功安装 TAO Launcher 并添加权限

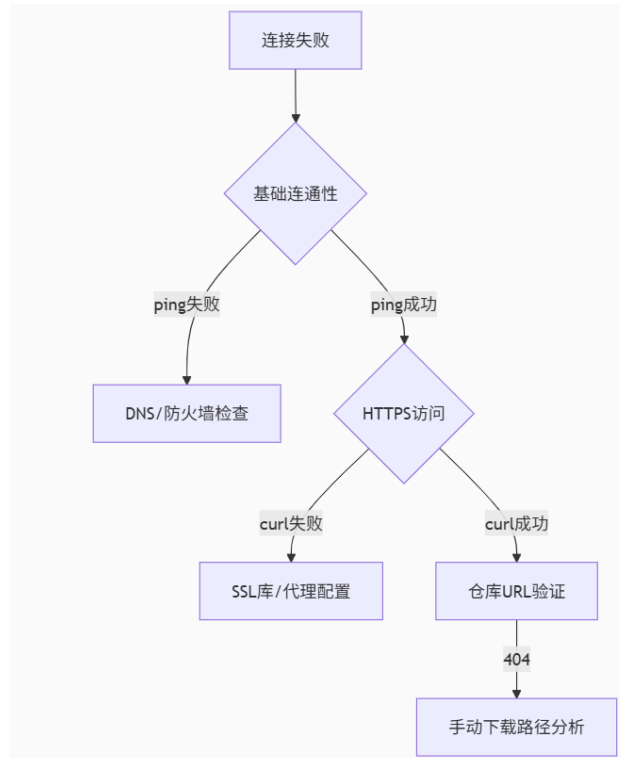


图 7：网络问题诊断框架

TAO 需要特定的 CUDA、TensorRT 和 Python 依赖组合，Docker 容器可确保环境一致，避免与主机系统冲突。TAO Toolkit 依赖 CUDA 进行 GPU 加速计算，所有训练和推理任务均通过 CUDA 调用 GPU 算力，必须安装与 TAO 版本匹配的 CUDA。TAO 生成的模型可通过 TensorRT（基于 CUDA）优化，显著提升边缘设备（如 Jetson）的推理速度。TAO Launcher 是一个基于 Python 的轻量级命令行界面，充当基于 PyTorch、TensorFlow 和 TensorRT 构建的 TAO 容器。

NVIDIA TAO Toolkit 的核心流程始于从 NGC (NVIDIA GPU Cloud) 下载预训练模型，用户可通过 Jupyter Notebook 快速启动迁移学习任务。首先在 NGC 模型库中选择适合的预训练模型（如 ResNet 或 YOLOv4），然后使用 TAO 提供的 Jupyter Notebook 模板加载模型并配置训练参数，包括数据路径、学习率和训练轮数等。在 Notebook 中，用户可以用少量标注数据对模型进行微调，TAO 会自动处理数据增强和分布式训练等复杂过程。训练完成后，通过内置工具对模型进行剪枝和量化优化，最终导出为 TensorRT 格式，直接部署到 Jetson 等边缘设备。整个过程在 Docker 容器中完成，确保环境一致性，同时支持实时监控训练指标和可视化模型性能，显著降低了从模型开发到边缘部署的技术门槛。