

[本页PDF](#)

`阿里云镜像: -i https://mirrors.aliyun.com/pypi/simple/`

`nvidia-smi` : 针对N卡, 查看GPU型号, 显存使用情况, 使用率等

`lscpu` : 查看cpu的信息

- `P-core` : 性能核心 (支持超线程, 可以虚拟化为2个核心)
- `E-core` : 能效核心 (不支持超线程)

`lscpu | grep cache` : 显示各级缓存大小

`ls` : 查看当前目录下信息

`conda env list` : 查看已建立的虚拟环境

`conda deactivate` : 退出当前环境

`pwd` : 查看当前路径

`conda create -n videollava python = 3.10 -y` : 建立一个名为videollava的虚拟环境

`mv oldname newname` : 给文件夹命名

`conda remove --name myenv --all` : 删除名字为myenv的虚拟环境

`rm -rf your_folder_name` : 删除文件夹

`conda search qwen-vl-utils -c conda-forge` : search为寻找, 后面跟着包名, `-c` conda-forge 为社区频道

`conda install -c conda-forge qwen-vl-utils=0.0.11 -y` : `-y` 为直接同意

`uptime` : 查看系统运行时间

`uptime -s` : 查看上次运行时间

`top` : 实时监控CPU和内存使用

- 按1: 监控CPU运行情况
  - us: 用户态占用
  - sy: 系统态占用
  - id: 空闲率

`history` : 查看历史命令

`df -h` : 查看磁盘空间使用情况, 硬盘是永久存储的, 但读取速度很慢

- `tmpfs` : 临时文件系统, 数据存在RAM(内存)中
- `/dev/nvme0n1p2` : 主系统分区
- `/dev/nvme0n1p1` : 存储系统启动文件
- `efivarfs` : 虚拟文件系统

`free -h` : 查看内存使用情况, 内存是临时存储的, 断电后会丢失, 读取速度极快

- `total` : 总内存
- `used` : 已用内存(包括缓存和缓冲区)
- `free` : 完全空闲的内存
- `buff/cache` : 被内核缓存和缓冲区占用的内存
- `available` : 实际可用的内存

`nvcc --verison` : 查看cuda版本

`tensorboard --logdir=./logs --port=6006 --bind_all`

- `--logdir` : 指向你的日志目录(和`logging_dir`一致)
- `--port` : 指定端口(默认6006)
- `--bind_all` : 允许外部访问

`ps aux` : 查看进程

- `ps` : Process Status(进程状态)
- `a` : 显示所有用户的进程(而不仅是当前用户)
- `u` : 以用户为导向的格式(显示详细信息)
- `x` : 包括未关联终端的进程(如后台服务)

`kill -9 1234` : 强制杀死PID为1234的进程

`ssh -L [本地端口]:[远程主机]:[远程端口] [用户名]@[服务器地址]` : 将远程服务器端口转发至本地端口

## 防火墙相关

`sudo ufw status` : 用管理员权限查看防火墙状态

`sudo ufw allow/deny 8080` : 开放/拒绝8080端口

`sudo ufw enable/disable` : 启用/禁用防火墙

## 系统服务相关

`sudo systemctl status nginx` : 查看服务状态

`sudo systemctl start/stop/restart redis` : 启动/停止/重启服务

`sudo systemctl enable/disable redis` : 启动/取消开机自启

`tops` : 衡量处理器基本运算操作的次数, 单位为**万亿次/秒**

## 对比单位:

- 1 GOPS = 10亿次/秒
- 1 POPS = 1000万亿次/秒

## 模型参数

### 精度

名称	参数	位数	适用范围
全精度 (FP32)	<code>torch.float32</code>	32	训练, 高精度推理
半精度 (FP16)	<code>torch.float16</code>	16	推理
脑浮点 (BF16)	<code>torch.bfloat16</code>	16	Ampere架构GPU训练
TF(32)	自动启用(为FP32)	19	NVIDIA Ampere架构的矩阵运算加速
INT8	1字节/参数 ( FP32的1/4 )	8-bit整数	边缘设备部署
INT4	0.5字节/参数 ( FP32的1/8)	4-bit整数	边缘设备部署

## 变体

模型权重的存储格式或优化版本, 与精度绑定

名称	说明	关联精度
fp32	全精度权重	<code>torch.float32</code>

fp16	半精度权重	torch.float16
bf16	脑浮点权重	torch.bfloat16
tf32	计算时使用(为FP32)	自动启用

```
# 8-bit量化配置
quant_config = BitsAndBytesConfig(
    load_in_8bit=True,
    bnb_4bit_compute_dtype=torch.bfloat16
)
```

计算的时候反量化为bfloat16，存储的时候还是INT8-bit

## CUDA

### 硬件层面

- **CUDA Core**: GPU 的基本计算单元，每个核心可执行一个线程的运算
- **SM ( Streaming Multiprocessor )**：
  - GPU 的计算单元，每个 SM 包含多个 CUDA Core、共享内存、寄存器等。

### 内存层次

- **全局内存 ( Global Memory )**：所有线程可访问，带宽高但延迟高（类似 CPU 的 RAM）。
- **共享内存 ( Shared Memory )**：SM 内线程共享，低延迟（类似 CPU 缓存）。
- **寄存器 ( Registers )**：每个线程私有，速度最快。

### 软件层面

- **Kernel** ( 内核函数 )
  - 在GPU上并行执行的函数
- **线程层次**
  - **Thread**: 最小执行单元
  - **Block**: 一组线程，共享同一SM的资源
  - **Grid**: 多个Block的集合，构成一个完整的Kernel任务

### CUDA核心组件

- **CUDA驱动程序**: 提供GPU硬件的基础访问能力

- **cuDNN**: 深度学习算子优化
- **cuFFT**: 快速傅里叶变换
- **cuBLAS**: 矩阵乘法、线性代数
- **NVCC**: 编译器，把.cu文件编译成GPU可执行代码，**要匹配GPU架构！！！**