链滴

矩池云 | GPU 分布式使用教程之 Pytorch

作者：matpool
原文链接：https://ld246.com/article/1669948587928
来源网站：链滴

GPU 分布式使用教程之 Pytorch

Pytorch 官方推荐使用 DistributedDataParallel(DDP) 模块来实现单机多卡和多机多卡分布式计算。DP 模块涉及了一些新概念，如网络（World Size/Local Rank），代码修改（数据分配加载），多种动方式（torchrun/launch），使用前请参考官方文档以及更多学习资料。

选择机器

- 单机多卡分布式：租用同个计算节点的多张卡即可。
- 多机多卡分布式：需要先申请开通 分布式集群 功能， 点击这里申请开通，在租用时，请选择带如图所示图标的机器。没有这个图标的机器不支持加入分布式网络。

单机多卡

1）租用机器： 为实现Pytorch的单机多卡分布式，首先，您需要按正常流程租用GPU，如单节点 4 卡 A2000，选择Pytorch镜像，如Pytorch 1.12镜像。

租用的时候 GPU 数设置成 4，即表示 4 卡，对应显存、内存等配置也会翻倍。

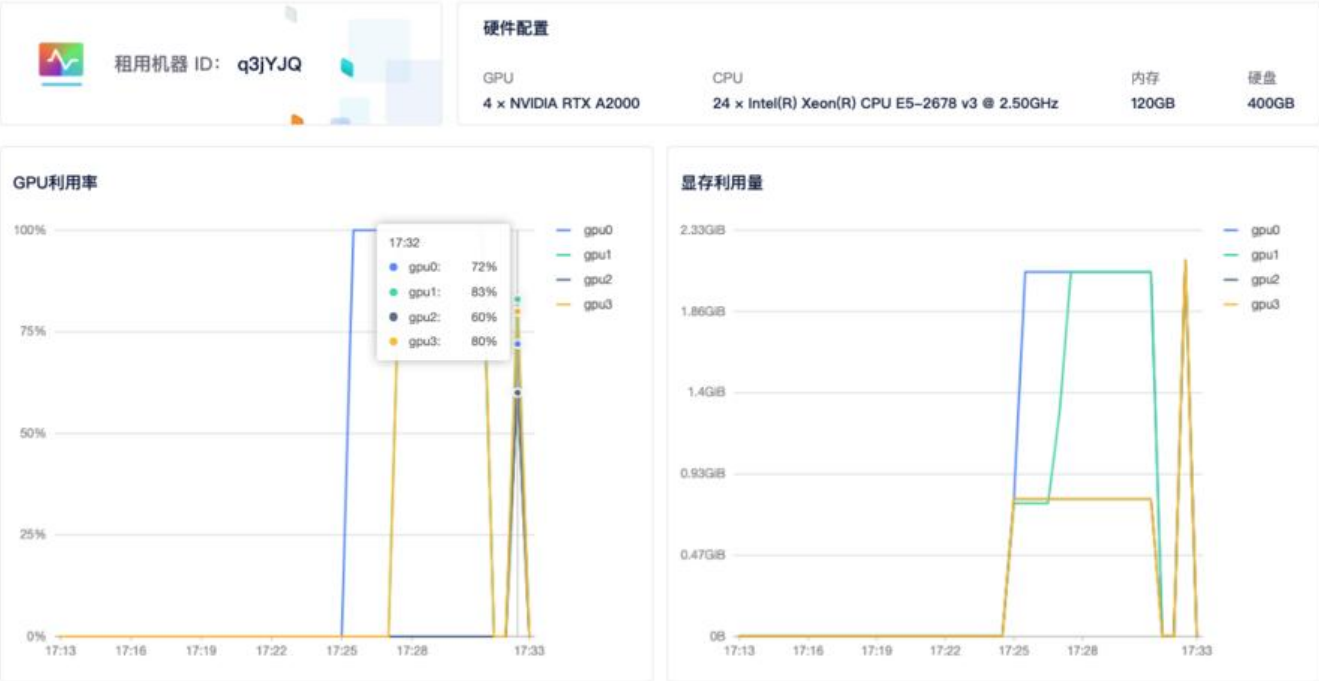
2）适配代码： 分布式需对脚本进行相应修改，可参考官方文档。此处使用开源demo.py

3）运行代码： 进入运行脚本所在目录，输入命令行，如：

```
cd /mnt/test/multi-card/torch
python -m torch.distributed.launch --nproc_per_node=4  mnmc_ddp_launch.py
```

这里使用的是 launch 启动方式，也可使用torchrun以及其他启动方式。--nproc_per_node 指定每节点的GPU数量，mnmc_ddp_launch.py 为执行脚本文件（如需下载 cifar10 数据集，修改downloa =True)。

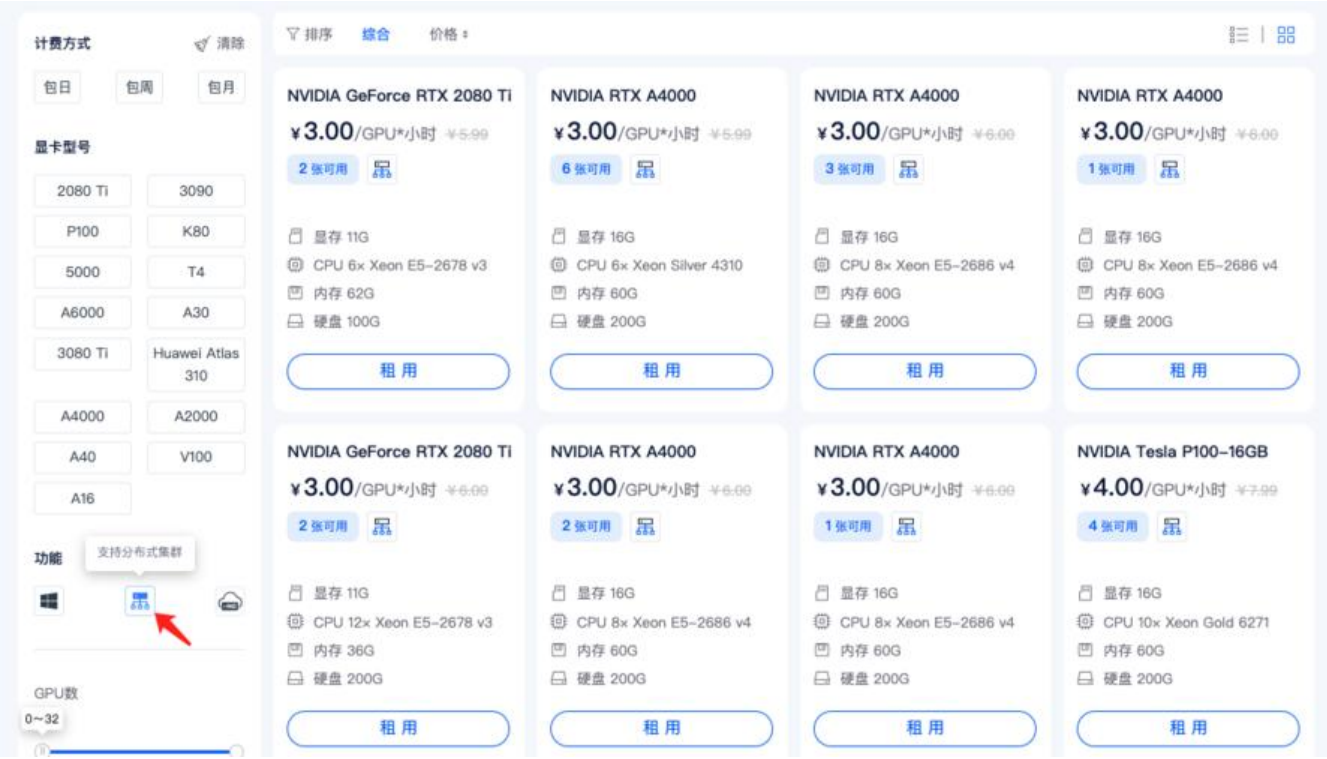
4）查看GPU使用情况： 租用界面点击 详情按钮即可查看 GPU、CPU使用情况。从截图中可以看到 4 个显卡都有使用到。



多机多卡

多机多卡使用需要先申请开通 分布式集群 功能，点击这里申请开通

1）租用机器： 首先，您需要按正常流程租用 GPU，主机市场筛选栏选择 支持分布式集群 筛选，然选择自己需要的机器租用即可。



如两个计算节点，租用两台 A2000 4 卡，共计 8 卡。选择相同的Pytorch镜像，如Pytorch 1.12。

注意： 多机多卡中每个节点的 GPU 卡数应该一样，才能都使用上，机器类型也最好一样。

2）创建集群： 进入 【个人中心】 — 【我的租用】 — 【分布式集群】。

分布式集群需要先进行申请，申请通过后，点击【添加集群】- 【添加机器】—【确定】。

3）添加机器： 点击集群页面添加机器按钮，勾选要加入集群的机器，点击确定，即可将租用机器添到集群。

添加机器成功后，系统会给每个节点分配集群 IP，当状态为已连接时，代表机器间可相互通信。

IP	机器ID	机器备注	机器型号	带宽	镜像	状态	添加时间	操作
192.168.1.9/24	q82Jz4	–	NVIDIA RTX A2000 x4	2.5Gbps	Pytorch 1.12	已连接	11–16 13:41	释放 移除
192.168.1.10/24	qK9aml	–	NVIDIA RTX A2000 x4	2.5Gbps	Pytorch 1.12	已连接	11–16 13:41	释放 移除

4）添加机器： 登录任一节点。因秘钥由您掌握，故需由您按以下步骤完成节点间的ssh连通:

```
ssh-keygen -t rsa # 一路默认，生成公私钥
ssh-copy-id root@其他节点IP  #分发给其他节点，输入对应秘钥。IP可在我的集群页面查看，如192168.1.1
```

5）添加以下环境变量： 在每一个节点，使用 ifconfig 命令查询节点网卡名称，如 meth01，meth0。登陆各个节点添加相同环境变量（可用 ssh 登录）

```
(myconda) root@PYMQoe:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.0.3  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:ac:11:00:03  txqueuelen 0  (Ethernet)
        RX packets 419  bytes 252761 (252.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 381  bytes 5811408 (5.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

meth920: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1450
        inet 192.168.1.4  netmask 255.255.255.0  broadcast 192.168.1.255
        ether 02:00:01:00:00:04  txqueuelen 1000  (Ethernet)
        RX packets 114  bytes 18072 (18.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 77  bytes 17054 (17.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
export NCCL_SOCKET_IFNAME=meth919,meth920
export GLOO_IFACE=meth919,meth920
export NCCL_DEBUG=INFO #可选，如需获得额外的nccl信息
```

可以将以上内容添加到 ~/.bashrc文件中（meth917 meth918记得改成自己的网卡名称）。

6）适配代码： 分布式需对脚本进行相应修改，可参考官方文档。此处使用开源demo.py

6）运行程序： 登录主节点，进入运行脚本所在目录，输入命令行，如：

```
cd /mnt/test/multi-card/torch
python -m torch.distributed.launch --nproc_per_node=2 --nnodes=2 --node_rank=0 --master addr="192.168.1.2" --master_port=12345 mnmc_ddp_launch.py
```

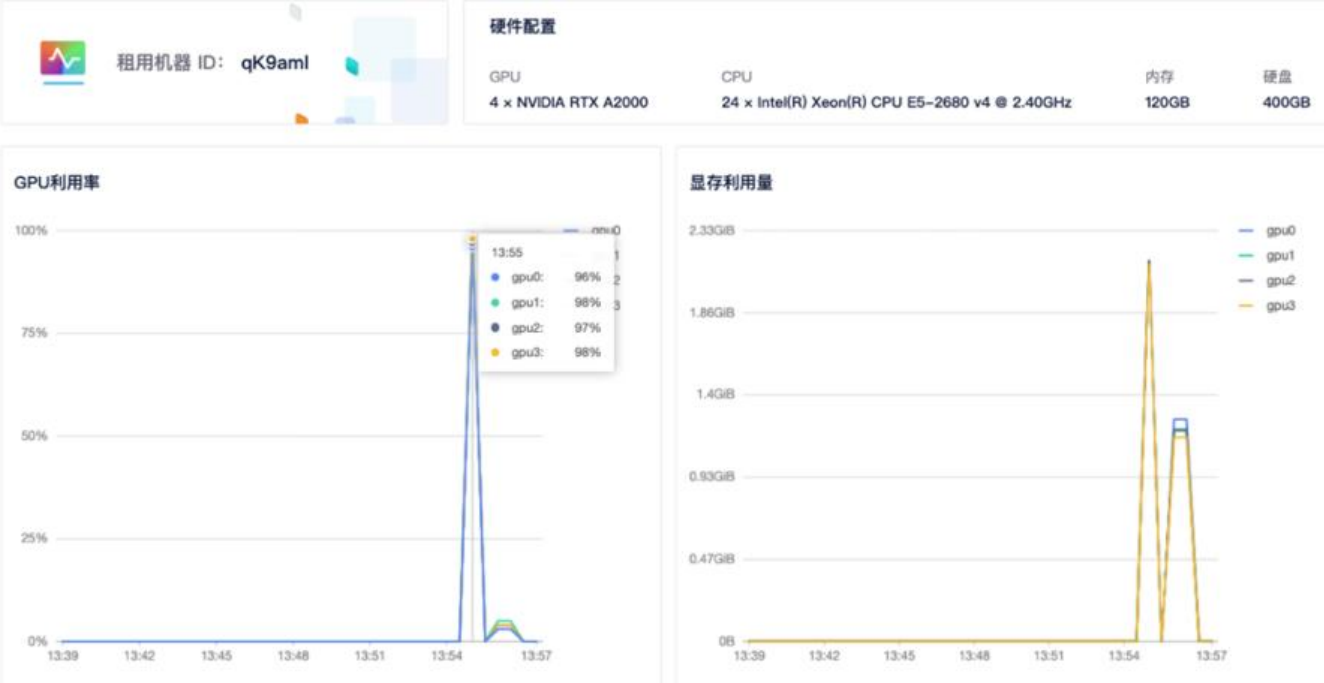
--nproc_per_node 指定每个节点的GPU数量，每个节点GPU数量应该一样，不然无法运行成功，--nodes 指定节点数（总共2个节点），--node_rank 指定节点顺序（主节点故为0号），--master_add和master_port 设定主节点ip和端口号。demo.py 为执行脚本（如需下载cifar10数据集，修改download=True)。

登录剩余节点，运行：

```
cd /mnt/test/multi-card/torch
python -m torch.distributed.launch --nproc_per_node=2 --nnodes=2 --node_rank=1 --master addr="192.168.1.2" --master_port=12345 mnmc_ddp_launch.py
```

其中，--node_rank 指定节点顺序（第二个节点故为1号），如有更多节点，需做相应修改，其他参不用修改。运行后，系统会自动连接并运行训练任务。

7）查看GPU使用情况： 租用界面点击 详情按钮即可查看 GPU、CPU使用情况。


租用机器 ID： qK9aml
硬件配置
GPU
4 × NVIDIA RTX A2000
CPU
24 × Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz
内存
120GB
硬盘
400GB
GPU利用率
100%
75%
50%
25%
0%
13:39
13:42
13:45
13:48
13:51
13:54
13:57
13:55
gpu0: 96%
gpu1: 98%
gpu2: 97%
gpu3: 98%
显存利用量
2.33GiB
1.86GiB
1.4GiB
0.93GiB
0.47GiB
0B
gpu0
gpu1
gpu2
gpu3