大模型（LLMs）显存问题面

1. 大模型大概有多大，模型文件有多大?

一般放出来的模型文件都是 fp16 的，假设是一个 nB 的模型，那么模型文件占 2nG，fp16 加载到显存里做推理也是占 2nG，对外的 pr 都是 10n 亿参数的模型。

2. 能否用 4 * v100 32G 训练 vicuna 65b?

不能。

首先，llama 65b 的权重需要 5* v100 32G 才能完整加载到 GPU。

其次，vicuna 使用 flash-attention 加速训练，暂不支持 v100，需要 turing 架构之后的显卡。（fastchat 上可以通过调用 train 脚本训练 vicuna 而非 train_mem，其实也是可以训练的）

3. 如果就是想要试试 65b 模型，但是显存不多怎么办?

最少大概 50g 显存，可以在 llama-65b-int4（gptq）模型基础上 LoRA[6]，当然各种库要安装定制版本的。

4. nB 模型推理需要多少显存?

考虑模型参数都是 fp16，2nG 的显存能把模型加载。

5. nB 模型训练需要多少显存?

基础显存：模型参数+梯度+优化器，总共 16nG。

activation 占用显存，和 max len、batch size 有关。

解释：优化器部分必须用 fp32（似乎 fp16 会导致训练不稳定），所以应该是 2+2+12=16，参考 ZeRO 论文。

注：以上算数不够直观，举个例子？

7B 的 vicuna 在 fsdp 下总共 160G 显存勉强可以训练。（按照上面计算 7*16=112G 是基础显存）所以全量训练准备显存 20nG 大概是最低要求，除非内存充足，显存不够 offload 内存补。

6. 如何估算模型所需的 RAM?

首先，我们需要了解如何根据参数量估计模型大致所需的 RAM，这在实践中有很重要的参考意义。我们需要通过估算设置 batch_size，设置模型精度，选择微调方法和参数分布方法等。接下来，我们用 LLaMA-6B 模型为例估算其大致需要的内存。

首先考虑精度对所需内存的影响：

fp32 精度，一个参数需要 32 bits, 4 bytes.

fp16 精度，一个参数需要 16 bits, 2 bytes.

int8 精度，一个参数需要 8 bits, 1 byte.

其次，考虑模型需要的 RAM 大致分三个部分：

模型参数

梯度

优化器参数

模型参数：等于参数量*每个参数所需内存。

对于 fp32，LLaMA-6B 需要 6B*4 bytes = 24GB 内存

对于 int8，LLaMA-6B 需要 6B*1 byte = 6GB

梯度：同上，等于参数量*每个梯度参数所需内存。

优化器参数：不同的优化器所储存的参数量不同。

对于常用的 AdamW 来说，需要储存两倍的模型参数（用来储存一阶和二阶 momentum）。

fp32 的 LLaMA-6B，AdamW 需要 6B*8 bytes = 48 GB

int8 的 LLaMA-6B，AdamW 需要 6B*2 bytes = 12 GB

除此之外，CUDA kernel 也会占据一些 RAM，大概 1.3GB 左右，查看方式如下。

```
> torch.ones((1，1)).to("cuda")
> print_gpu_utilization()
>>>
GPU memory occupied: 1343 MB
```

综上，int8 精度的 LLaMA-6B 模型部分大致需要 6GB+6GB+12GB+1.3GB = 25.3GB 左右。再根据 LLaMA 的架构（hidden_size = 4096, intermediate_size =11008, num_hidden_layers = 32, context_length = 2048）计算中间变量内存。

每个 instance 需要：

(4096 +11008)* 2048 *32 * 1byte = 990MB

所以一张 A100（80GB RAM）大概可以在 int8 精度，batch_size = 50 的设定下进行全参数训练。查看消费级显卡的内存和算力：

2023 GPU Benchmark and Graphics Card Comparison Chart

https://www.gpucheck.com/gpu-benchmark-graphics-card-comparison-chart

7. 如何评估你的显卡利用率

zero3 如果没有 nvlink，多卡训练下会变慢。但是一直不知道究竟会变得多慢，下面给出几种方法来评估自己在训练时发挥了多少 gpu 性能，以及具体测试方法。

7.1 flops 比值法

测试工具：deepspeed

参考数据：nvidia 公布的显卡 fp16 峰值计算速度（tensor core）

gpu 利用率 = 实测的 flops/显卡理论上的峰值 flops

举例：deepspeed 实测 flops 100tflops，而用的是 A100 卡理论峰值 312tflops，可以得到 GPU 利用率只有 32.05%

7.2 throughout 估计法

测试工具：手动估算 或者 deepspeed

参考数据：论文中的训练速度或者吞吐量

吞吐量 = example 数量/秒/GPU * max_length

gpu 利用率 = 实际吞吐量 / 论文中的吞吐量（假设利用率 100%）

举例：

实测训练时处理样本速度为 3 example/s，一共有 4 卡，max length 2048，则吞吐量为 1536 token/s/gpu

根据 llama 论文知道，他们训练 7B 模型的吞吐量约为 3300 token/s/gpu，那么 GPU 利用率只有 46.54%

7.3 torch profiler 分析法

测试工具：torch profiler 及 tensorboard

参考数据：无

利用 torch profiler 记录各个函数的时间，将结果在 tensorboard 上展示，在 gpu kenel 视图下，可以看到 tensor core 的利用率，比如 30%

总结：以上三种方法，在笔者的实验中能得到差不多的利用率指标。

从准确性上看，方案三 > 方案一 > 方案二

从易用性上看，方案二 > 方案一 > 方案三

如果不想改代码就用方案二估算自己的训练速度是不是合理的，如果想精确分析训练速度的瓶颈还是建议使用方案三。

8. 测试你的显卡利用率实现细节篇

8.1 如何查看多机训练时的网速？

iftop 命令，看网速很方便。

8.2 如何查看服务器上的多卡之间的 NVLINK topo？

```
$ nvidia-smi topo -m
```

8.3 如何查看服务器上显卡的具体型号？

```
cd /usr/local/cuda/samples/1_Utilities/deviceQuery
make
./deviceQuery
```

8.4 如何查看训练时的 flops？（也就是每秒的计算量）

理论上，如果 flops 比较低，说明没有发挥出显卡的性能。

如果基于 deepspeed 训练，可以通过配置文件很方便的测试。

```
{
  "flops_profiler": {
    "enabled": true,
    "profile_step": 1,
    "module_depth": -1,
    "top_modules": 1,
    "detailed": true,
    "output_file": null
    }
}
```

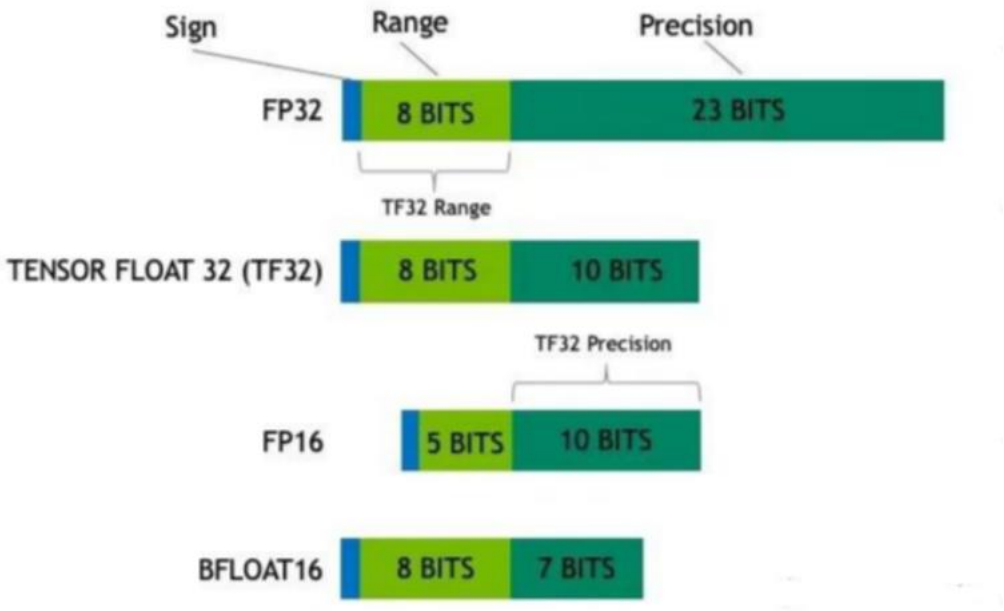
参考：https://www.deepspeed.ai/tutorials/flops-profiler/

8.5 如何查看对 deepspeed 的环境配置是否正确？

```
$ ds_report
```

8.6 tf32 格式有多长？

19 位



8.7 哪里看各类显卡算力比较？

https://lambdalabs.com/gpu-benchmarks

8.8 （torch profiler）如何查看自己的训练中通信开销？

用 pytorch profiler 查看，下面给出基于 transformers 的一种快捷的修改方式。

https://github.com/yqhu/profiler-workshop/blob/c8d4a7c30a61cc7b909d89f88f5fd36b70c55769/hf_training_trainer_prof.py

用记录的 pt.trace.json 文件放到 tensorboard 上，可以看出 tensor core 的利用率。

根据实践经验，使用 deepspeed zero3 时，pcie 版本的卡很大部分时间都在通信上，AllGather 和 ReduceScatter 的时间超过 tensor core 计算的时间，所以 flops 上不去。