The Usage of LLMs

Computational Challenges

CUDA (Compute Unified Device Architecture): a collection of libs and tools for Nvidia GPUs

Solution: Quantization

1 parameter = 4 bytes (32-bit float)
1B parameters = 4 x 10^9 bytes = 4GB

4GB @ 32-bit full precision

	Bits	Exponent	Fraction	Memory needed to store one value
FP32	32	8	23	4 bytes
FP16	16	5	10	2 bytes
BFLOAT16	16	8	7	2 bytes
INT8	8	–/–	7	1 byte

	Bytes per parameter
Model Parameters (Weights)	4 bytes per parameter
Adam optimizer (2 states)	+8 bytes per parameter
Gradients	+4 bytes per parameter
Activations and temp memory (variable size)	+8 bytes per parameter (high-end estimate)
TOTAL	**=4 bytes per parameter +20 extra bytes per parameter**

80GB @ 32-bit full precision

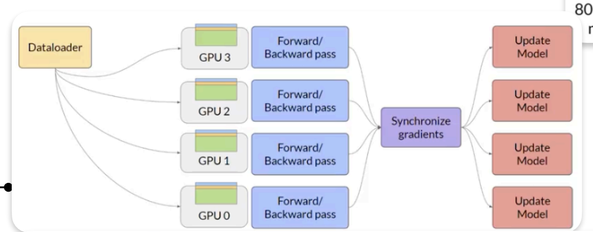
40GB @ 16-bit half precision

20GB @ 8-bit precision

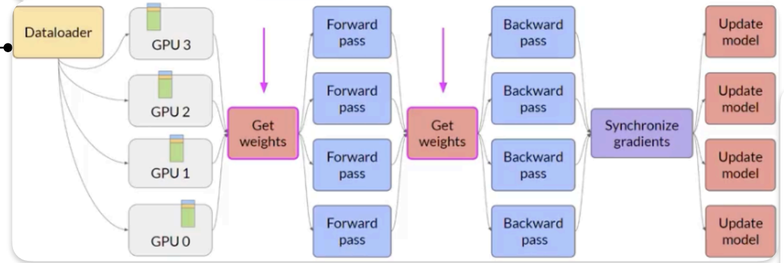
80GB is the maximum memory for the Nvidia A100 GPU, so to keep the model on a single GPU, you need to use 16-bit or 8-bit quantization.

Efficient Multi-GPU Compute Strategies

Distributed Data Parallel (DDP)



Fully Sharded Data Parallel (FSDP)



Scaling Choices for Pre-training

Goal: maximize model performance

Constraint: Compute budget (GPUs, time, cost), which can be measured by "PetaFlops per second-Day"

A petaflop/s-day (pfs-day) consists of performing 10^15 neural net operations per second for one day, or a total of about 10^20 operations. The compute-time product serves as a mental convenience, similar to kW-hr for energy. We don't measure peak theoretical FLOPS of the hardware but instead try to estimate the number of actual operations performed. We count adds and multiplies as separate operations, we count any add or multiply as a single operation regardless of numerical precision (making "FLOP" a slight misnomer), and we ignore ensemble models.

OpenAI定义神经网络中的一次乘法或者一次加法为一个操作，这一点和一般意义上的浮点运算操作FLOP略有不同。如果每秒钟可以进行10的15次方运算，也就是1 peta flops，那么一天就可以进行约10的20次方运算，这个算力消耗被称为1个petaflop/s-day。

Scaling Choice: Dataset size (no. of tokens)/ Model size (no. of param.)

Chinchilla scaling laws: Compute optimal training data size is ~20x number of params

Pre-training for Domain Adaptation, e.g. BloombergGPU