Question 1 Computational Efficiency w.r.t Batch Size (5pt)

One Resnet-18 model

- it takes 19.27 seconds to train one epoch (w/o data-loading) using mini-batch size 32
- 12.05 seconds using mini-batch size 128
- 10.468 seconds using mini-batch size 512
- 10.41 seconds using mini-batch size 1024
- 10.3 seconds using mini-batch size 2048

Overall, when batch size is larger, it takes shorter time to train, because by using bigger batch sizes, we speed up training, as it is equivalent to taking a few big steps, instead of taking many little steps.

Question 2 Speedup Measurement (5pt)

	Batch-size 32 per GPU		Batch-size 128 per GPU		Batch-size 512 per GPU	
	Time (sec)	Speedup	Time(sec)	Speedup	Time (sec)	Speedup
1-GPU	19.27	1	12.05	1	10.468	1
2-GPU	30.38	0.634	12.82	0.94	8.91	1.17
4-GPU	36.92	0.522	11.73	1.03	5.51	1.90

- It's strong scaling because of Constant problem size while increasing number of GPUs
- Weak scaling would be worse.
    - When using one GPU and increasing mini-batch size from 512 to 2048, the speedup is 1.07% (from 12.05s to 10.468s)
    - Comparing 1 GPU to 4 GPUs with mini-batch size of 512, the speedup is 89.98% from 10.468s to 5.51s

Question 3 Computation vs Communication (15pt)

3.1 How much time spent in computation and communication (5pt)

	Batch-size 32 per GPU		Batch-size 128 per GPU		Batch-size 512 per GPU	
	Compute(sec)	Comm(sec)	Compute(sec)	Comm(sec)	Compute(sec)	Comm(sec)
2-GPU	10.813	20.448	6.02	6.8	5.234	3.676
4-GPU	15.27	21.65	3.01	8.72	2.617	2.893

- Let G_number be total number of GPU used
- Total time of using 1 GPU is from question 1
- Total time of using multiple GPUs is from question 2
- Communication time = (Total time of using 1 GPU/G_number) – (Total time of using multiple GPU)
- Compute time = Total time – communication time

3.2 Communication bandwidth utilization (10pt)

the formula to calculate how long does it take to finish an allreduce.

- the minimum time for the all-reduce operation is at least

$$T(\lceil 2 \times \frac{N-1}{N} \times X \rceil \times itsize)$$

the formula to calculate the bandwidth utilization.

- Assume that (1) data are not compressed during the all-reduce operation; and initial items are independent of one another. To perform an all-reduce operation on X
- items with itsize bytes item size on N processes, there exists at least one process that must communicate a total of at least

$$\lceil 2 \times \frac{N-1}{N} \times X \rceil \times itsize \text{ bytes data}$$

list the calculated results in Table 3.

	Batch-size-per-GPU 32	Batch-size-per-GPU 128	Batch-size-per-GPU 512
	Bandwidth Utilization(GB/s)	Bandwidth Utilization(GB/s)	Bandwidth Utilization(GB/s)
2-GPU	32	128	512
4-GPU	48	192	768

Question 4 Large Batch Training (10 pt)

4.1 Accuracy when using large batch

- For batch size of 8192 per GPU on 4 GPUs
  - The average training loss is 1.996
  - Training accuracy is 26%

4.2 How to improve training accuracy when batch size is large

Two remedies that can improve training accuracy when batch size is large.

- Remedy 1: apply the linear scaling rule with a minibatch of 8k images ($\eta = 0.1 \cdot 32$)
- Remedy 2: with customized initialization. To be specific, $\gamma = 0$ for the final BN layer of each residual block

Question 5 Distributed Data Parallel (5pt)

- One needs to set up epoch ID because it is necessary to make shuffling work properly across multiple epochs. Otherwise, the same ordering will be always used.
- The difference between DP and DDP is how they handle gradients. DP accumulates gradients to the same .grad field, while DDP first use all_reduce to calculate the gradient sum across all processes and divide that by world_size to compute the mean.

Question 6 What are passed on network ? (5pt)

- No, each GPU must communicate both gradients and parameter values on every update step due to batch norm layer.

Question 7 What if we only communicate gradients ? (5pt)

- Yes, because there is no data dependency between gradients across layers. as soon as the gra-dient for a layer is computed, it is aggregated across workers, while gradient computation for the next layer continues.