	P1	P2A	P2B	P3
Accuracy	22%	10%	10%	11%
Average running time for 9 iterations	2.27s	3.381s	2.95s	2.65s
Average loss	2.2	2.33	2.32	2.32

P1 Analysis

For p1, there is hardly any space for any analysis as the model is a typical forward and backward pass for the model VGG-11.

With the batch size 256, we print out the loss every 20 iterations.
It could be seen that the loss is approximately 2.2.
It is noted that the accuracy in the final is pretty low. This is because we only run for 1 epoch. And the gradient is hardly converged after 1 epoch.
However, part 1 serves as a good criterion for measuring the performance of p2 and p3.

In terms of the scalability in P1. The implementation of VGG-11 is not going to scale well even with a simple data parallelism. Since the DataParallel replicates the model in every forward pass, and its single-process multi-thread parallelism naturally suffers from the locking and queuing mechanism.

P2 Analysis

For P2a and P2b, there is barely difference in accuracy since only sync method is different. There might be some float precision differences such as cut-offs which cause the minor differences. The times of two programs have more significant differences since gathering and scattering burdens are heavier on master's internet bandwidth, while all_reduce utilizes peer-to-peer communication so the overhead for synchronization is lower. In addition, all_reduce might have some built-in optimizations while written gather-scatter does not. This further slows down the performance.

It is clear that gather-scatter is not well scalable since it requires very high network bandwidth and computation for master, while all_reduce is weakly scalable since the workload increases linearly.

Note that the code change in p2a from p1 is trivial. This is of the original Data Parallelism as mentioned in the paper:: **Non-intrusive.**

P3 Analysis

As our experiments in p2 points out, the bottleneck of syncing the gradient lies in the master node. Using the built in distributed module in pytorch overall performs similar to the P2, like the training result accuracy is lower than the none-distributed running on single machine and the running for each iteration is also slower because of use of distributed machine. However, it is worth to mention that the built in version has better latency compared to the manual gather and scatter operations on master in Part2, so PyTorch DDP training latency, which mostly consists of AllReduce communications in backward pass, has some optimizations than manual gather and scatter for each layer.

Contribution:

- **Zhikang Hao:**
  - **P2 code and analysis**
- **Ruoyu He:**
  - **P1 code and analysis**
- **Jiujiu Pan**
  - **P3 code and analysis**