Midterm Update: Traffic Pattern Analysis and Comparison of Distributed Deep Learning Models

Li He V00972954

Project Overview

This project aims to analyze and compare traffic patterns in distributed deep learning training across various models and topologies. The goal is to explore the computational and communication costs associated with distributed training, with a focus on reducing overhead, minimizing costs, and optimizing traffic patterns to enhance overall system performance.

Progress So Far

Implementation and Testbed Setup

- Successfully set up an account on Alliance Canada and gained access to clusters Beluga, Cedar, Graham, Narval, and Niagara.
- Configured the JupyterHub environment for interactive development.
- Experiment with basic commands of the SLURM cluster management. SLURM (Simple Linux Utility for Resource Management) is the job scheduler that manages compute resources. It is responsible for efficiently allocating CPU/GPU nodes, handling job queues, and ensuring fair usage across users.

```
# Check available resources in the cluster
!sinfo

# run nvidia-smi slurm job with 1 node allocation
!srun -N 1 nvidia-smi

# run nvidia-smi slurm job with 2 node allocation.
!srun -N 2 nvidia-smi

#!/bin/bash
        #SBATCH -N 1                                # Node count to be allocated for the job
        #SBATCH --job-name=dli_firstSlurmJob        # Job name
        #SBATCH -o /dli/nemo/logs/%j.out        # Outputs log file
        #SBATCH -e /dli/nemo/logs/%j.err        # Errors log file

        srun -l my_script.sh                        # my SLURM script
```

- Nvidia Deep Learning institute: Nvidia offer a deep learning workshop, it allocates a gpu for doing some experiments. Through those materials, I explored and utilized the available hardware configurations by gaining hands-on experience with job scheduling and resource allocation. Understanding the hardware resources accessible for computation: Figure 1 check the CPU information of the system using the lscpu command. Figure 2 check the information of GPUs. I will also begin running experiments on Alliance Canada, focusing on building and deploying large neural networks for the project.

```
!lscpu

Architecture:                    x86_64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
Address sizes:                   48 bits physical, 48 bits virtual
CPU(s):                          96
On-line CPU(s) list:             0-95
```

Figure 1: The CPU information of the system

```
!nvidia-smi

Fri Mar  7 21:27:05 2025
+---------------------------------------------------------------------------------------+
| NVIDIA-SMI 535.104.12             Driver Version: 535.104.12   CUDA Version: 12.2     |
|-----------------------------------------+----------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M. |
|                                         |                      |               MIG M. |
|=========================================+======================+======================|
|   0  NVIDIA A100 80GB PCIe          On  | 00000001:00:00.0 Off |                    0 |
| N/A   38C    P0              54W / 300W |      4MiB / 81920MiB |      0%      Default |
|                                         |                      |             Disabled |
+-----------------------------------------+----------------------+----------------------+
|   1  NVIDIA A100 80GB PCIe          On  | 00000002:00:00.0 Off |                    0 |
| N/A   37C    P0              55W / 300W |      4MiB / 81920MiB |      0%      Default |
|                                         |                      |             Disabled |
+-----------------------------------------+----------------------+----------------------+
|   2  NVIDIA A100 80GB PCIe          On  | 00000003:00:00.0 Off |                    0 |
| N/A   40C    P0              57W / 300W |      4MiB / 81920MiB |      0%      Default |
|                                         |                      |             Disabled |
+-----------------------------------------+----------------------+----------------------+
|   3  NVIDIA A100 80GB PCIe          On  | 00000004:00:00.0 Off |                    0 |
| N/A   39C    P0              56W / 300W |      4MiB / 81920MiB |      0%      Default |
|                                         |                      |             Disabled |
+-----------------------------------------+----------------------+----------------------+

+---------------------------------------------------------------------------------------+
| Processes:                                                                            |
|  GPU   GI   CI        PID   Type   Process name                            GPU Memory |
|        ID   ID                                                             Usage      |
|=======================================================================================|
|  No running processes found                                                           |
+---------------------------------------------------------------------------------------+
```

Figure 2: The information of GPUs

Scalable Training Strategies

Since this project focuses on Traffic Pattern Analysis and Comparison of Distributed Deep Learning Models, I also spent some time understanding scalable training strategies, aggregation algorithms, and communication synchronization. The efficiency of distributed training depends on how models, data, and computation are partitioned across multiple GPUs and nodes, directly impacting traffic patterns and system performance.

- Data Parallelism: 1, Replicate the entire model on each device. 2, Train all replicas simultaneously using different mini-batches. 3, Communication involves aggregating gradients (e.g., using all-reduce).
- Model Parallelism: 1, Split the model across multiple devices. 2, Suitable for large models that don't fit in memory.
- Aggregation Algorithms: 1, Centralized (Parameter Server): Workers report parameter updates to a central server. 2, Decentralized (All-Reduce): Workers exchange parameter updates directly. Decentralized (Gossip): Workers communicate updates with neighbors, achieving consistency at the end.
- Communication Synchronization: 1, Synchronous: Workers synchronize parameter updates after each iteration. Stragglers can impact system throughput. 2, Asynchronous: Workers transmit gradients to the parameter server without waiting for others. Eliminates synchronization but may introduce inconsistency.

Challenges Encountered

Module loading issues on the Alliance clusters (e.g., Python version conflicts). During the setup of TensorFlow under the myenv virtual environment on the Alliance Canada clusters, I faced the installation issue that impacted the ability to run distributed training experiments. I still need to find out what may caused that issue.

Next Steps

Implement further optimizations for distributed training and model parallelism. Conduct experiments on different clusters to compare network latencies. Develop a visualization dashboard for traffic pattern insights.