Go, change the world®

RV College of Engineering®

OPERATING SYSTEM

Distributed Deep Learning

SHASHANK SHENOY B(1RV22CS183)
YASHAS DONTHI(1RV22CS238)
DEPARTMENT OF COMPUTER SCIENCE ENGINEERING
R V COLLEGE OF ENGINEERING, BENGALURU – 560059, INDIA

INTRODUCTION

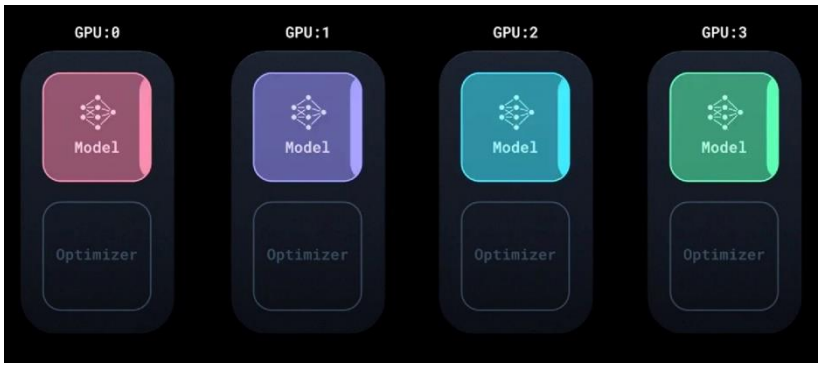
Deep learning has emerged as a powerful technique in the field of artificial intelligence, enabling machines to learn complex patterns and representations directly from data. Training such models on massive datasets often poses significant challenges in terms of computational resources, memory requirements, and training time. Distributed deep learning offers a promising solution to these challenges by distributing the computational workload across multiple nodes or devices

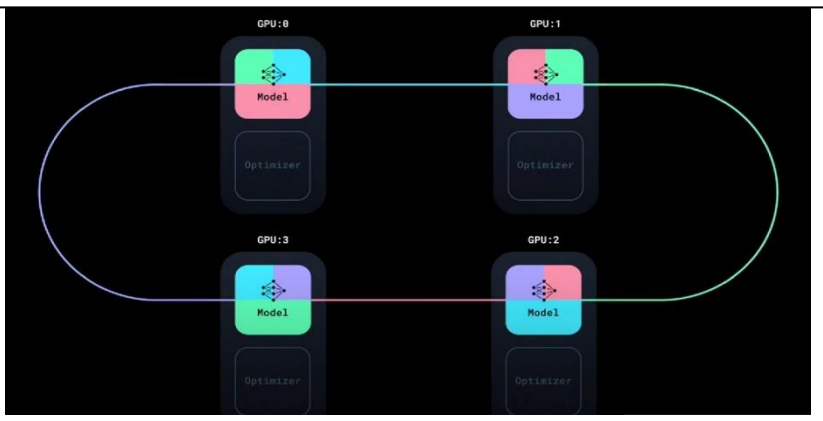
SYSTEM ARCHITECTURE

To implement distributed deep learning effectively, a comprehensive system architecture is required to coordinate the parallelization of training across multiple nodes or devices: Master Node and Worker Nodes. The dataset is partitioned and distributed across worker nodes to enable data parallelism. In addition to data parallelism, the model itself may be partitioned and distributed across worker nodes to enable model parallelism. In some distributed deep learning architectures, a parameter server may be used to manage the storage and synchronization of model parameters. The distributed training algorithm orchestrates the flow of data and computation across worker nodes to optimize the training process.

METHODOLOGY

Each node represents a computer or processing unit like a CPU or GPU. Data is divided into smaller chunks and distributed across the nodes. Worker nodes communicate with the parameter server to read and update global model parameters. Computing resources are dynamically allocated and managed to optimize performance and scalability. Load balancing algorithms adjust resource allocation dynamically based on node performance and workload demands.





SYSTEM CALLS USED

In distributed deep learning, OS system calls facilitate inter-process communication and synchronization between nodes. Functions like socket(), send(), recv(), pipe(), and mutex_lock() are commonly utilized to establish network connections, transfer data, and manage shared resources. These system calls enable efficient coordination and parallel execution of distributed deep learning tasks across multiple nodes within a networked environment.

SIMILARITIES WITH OPERATING SYSTEM CONCEPTS :

OS and distributed deep learning share concepts of resource management, synchronization, and fault tolerance across multiple entities (processes/nodes), highlighting the importance of scalability and reliability. Similarities include handling concurrency, resource allocation, and fault tolerance mechanisms, essential for efficient coordination and robustness in both domains.

OUTPUT/RESULTS

```
[GPU0] Epoch 20 | Batchsize: 32 | Steps: 16
[GPU3] Epoch 20 | Batchsize: 32 | Steps: 16[GPU2] Epoch 20 | Batchsize: 32 | Steps: 16

[GPU1] Epoch 20 | Batchsize: 32 | Steps: 16
[GPU3] Epoch 21 | Batchsize: 32 | Steps: 16
[GPU2] Epoch 21 | Batchsize: 32 | Steps: 16
Epoch 20 | Training snapshot saved at snapshot.pt
[GPU1] Epoch 21 | Batchsize: 32 | Steps: 16
[GPU0] Epoch 21 | Batchsize: 32 | Steps: 16
[GPU0] Epoch 22 | Batchsize: 32 | Steps: 16
[GPU3] Epoch 22 | Batchsize: 32 | Steps: 16
[GPU2] Epoch 22 | Batchsize: 32 | Steps: 16
[GPU1] Epoch 22 | Batchsize: 32 | Steps: 16
```

CONCLUSION:

- In conclusion, our project on distributed deep learning has provided us with valuable insights into the principles, challenges, and applications of leveraging distributed computing techniques for training deep learning models.
- Our investigation into system architecture revealed the intricate design of distributed deep learning frameworks, which leverage parallel processing, communication protocols, and resource management techniques to efficiently distribute computations across multiple nodes.

ACKNOWLEDGEMENTS

Dr. Jyoti Shetty
Dept of Computer Science

TEAM INFORMATION

Shashank Shenoy B(1RV22CS183), Yashas Donthi(1RV22CS238)