UNIVERSITÉ CÔTE D'AZUR

MASTER INFORMATIQUE

Project report

Evaluating the Efficiency of Data Parallelism in Machine Learning within NEF

Prepared by
Enzo Ghizzardi

Supervised by
No supervisor given

Master Informatique
at Université Côte d'Azur
Fall semester 2024-25

Contents

Part I
Introduction

1 About NEF Cluster

The NEF infrastructure serves as the multidisciplinary scientific experimentation platform of the Inria Sophia Antipolis Méditerranée research center, designed for advanced computing, storage, and visualization needs. This platform is built on a cluster-based architecture, leveraging multiple generations of high-performance servers to deliver a heterogeneous parallel environment. It integrates fast networking capabilities, ample storage capacity, and a seamless connection to Inria Sophia's interactive visualization tools.

This platform is accessible to all Inria research teams, with accounts also available for academic collaborators and industrial partners associated with Inria Sophia. NEF supports a variety of computational tasks, including parallel processing, GPU-based calculations, and sequential workloads, making it a versatile tool for diverse research applications.

NEF is a key component of the UCA OPAL distributed computation mesocenter. Initially funded by Inria, it now benefits from additional contributions through UCA, the UCAjedi IDEX initiative, and CPER 2015-2020 funding from Région Sud Provence-Alpes-Côte d'Azur, the French Ministry of Higher Education, Research, and Innovation (MESRI), and the CASA initiative.

We were fortunate, as part of this project, to obtain access to this platform to carry out our tests.

2 Nodes tested

For our experiments, we utilized two specific types of nodes available in the NEF cluster. The first was a Dell C6145 CPU node equipped with 64 cores, from which we reserved 4 cores for a duration of approximately one hour. The second was a Dell R7525 GPU node featuring two NVIDIA A100 GPUs, which we used for about half an hour. These nodes were selected based on a combination of their hardware specifications—providing sufficient resources for testing parallelization on 4 CPU cores and 2 GPUs—and their availability, as the NEF cluster often experiences significant wait times for more heavily utilized nodes.

Considering the computational requirements of our workload, we intentionally opted not to reserve the cluster's most advanced hardware. This decision allowed us to focus on efficiency and accessibility, ensuring our tests could be conducted without prolonged delays. Further details about the specific hardware configurations of these nodes are provided in Section 4.

3 Task

This report focuses on assessing the effectiveness of data parallelism in machine learning using the NEF cluster. The evaluation is conducted with a single machine learning model, utilizing both specific CPUs and GPUs available in the cluster. By varying the number of cores allocated to the computations, we aim to examine how performance metrics are influenced.

We must build a framework of the code that can generalize to a zoo of the models of torchvision and batch sizes.

We will measure the loading time and the computation + communication time for each execution.

We will store these results in CSV files to analyze them and generate plots.

Part II
Evaluation

4 Details of the node tested

In this evaluation, we used two specific nodes within the NEF cluster: nef008 for CPU-based computations. Below is a detailed overview of each node's hardware specifications.

We will use up to 8 CPU cores and 2 GPU cores (we should have used up to 3 GPU cores, but I never managed to reserve 3)

4.1 CPU node

- **Model**: Dell C6145
- **Processor**: Quad AMD Opteron 6376
  - **Clock Speed**: 2.3 GHz
  - **Total Cores**: 64 cores
- **Memory**: 256 GB RAM
- **Storage**: Single 500 GB SATA Hard Disk Drive
- **Network**:
  - Two Gigabit Ethernet ports (one active)
  - One InfiniBand QDR card
- **Hyper-Threading**: Not supported

Dell C6145 :

The Dell C6145 series, to which *nef008* belongs, is designed for high-performance computing environments, offering substantial processing power suitable for parallel computing tasks. The node's ample memory and storage capacity support data-intensive applications, while the InfiniBand connectivity facilitates high-speed data transfer, essential for distributed computing scenarios.

This node was selected to assess the performance of data parallelism in machine learning tasks within the NEF cluster, providing insights into the capabilities of CPU resources available in this high-performance computing environment.

4.2 GPU node

- **Model**: Dell R7525
- **Processor**: Dual AMD EPYC 7413
  - **Clock Speed**: 2.65 GHz
  - **Total Cores**: 48 cores
- **Memory**: 512 GB RAM
- **Storage**:
  - 4 × 960 GB NVMe drives
  - 1 × 1.6 TB NVMe drive for system
- **GPU Configuration**:
  - 4 × NVIDIA A100 GPUs
  - **CUDA Cores per GPU**: 6,912
  - **Memory per GPU**: 40 GB HBM2
  - **Single-Precision Performance**: 19.5 TFLOPS
  - **Double-Precision Performance**: 9.7 TFLOPS
  - **Memory Bandwidth**: 1,555 GB/s
- **Network**:
  - Two 100 Gbps InfiniBand HDR cards
  - Two 10 Gbps Ethernet ports
- **Hyper-Threading**: Supported

Dell R7525 :

The Dell R7525 nodes are designed for high-performance GPU-accelerated workloads. Equipped with four NVIDIA A100 GPUs, this node is optimized for intensive machine learning and deep learning tasks, delivering exceptional throughput and scalability. The high memory bandwidth and massive computational capacity of the A100 GPUs make them ideal for large-scale parallel processing.

5 Details of the training model tested - swin_b

The Swin Transformer Base (Swin-B) model is part of the Swin Transformer family, a groundbreaking architecture designed to revolutionize vision tasks by leveraging hierarchical attention mechanisms. Unlike traditional convolutional neural networks (ConvNets), Swin Transformers divide images into non-overlapping patches, enabling efficient modeling of both local and global dependencies. This design bridges the gap between ConvNets and Vision Transformers (ViTs), offering strong performance in image classification, object detection, and semantic segmentation tasks.

The Swin-B model features a hierarchical architecture with shifted window-based attention, which significantly reduces computational overhead while maintaining scalability. Its adaptability allows it to handle high-resolution images efficiently, making it suitable for tasks requiring detailed spatial understanding. The model has demonstrated state-of-the-art performance, achieving a top-1 accuracy of 85.2% on ImageNet-1K and excelling in downstream tasks such as COCO object detection and ADE20K semantic segmentation.

In our evaluation, we will use the Swin-B model with the following specifications:

- Model: Swin Transformer Base
- Pretrained Weights: ImageNet1K
- Number of Parameters: 87,768,321
- GFLOPS: 15.3 (for a 224 × 224 input image)
- File Size: 325 MB

The Swin-B model will be employed for image classification tasks on the NEF cluster. Its performance will be analyzed by varying the number of CPU and GPU cores used during computations, providing insights into the model's scalability and efficiency in parallelized environments. This evaluation will highlight its ability to handle data-intensive workloads effectively, making it an excellent candidate for distributed training experiments.

6 Impact of the number of computing devices

CPU tests were performed with 1, 2, 4, and 8 CPU cores.

GPU tests were performed with 1 and 2 GPU cores.

Test programs, plots generation programs, as well as raw results in CSV files are available in the rendering archive.

6.1 Plots results

Here are the graphs obtained after our tests.

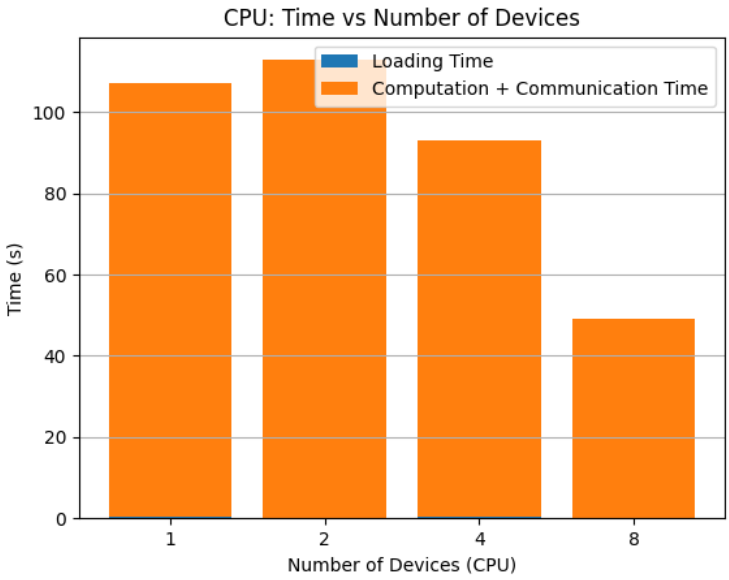

Figure 1: Time VS Number of devices (CPU)

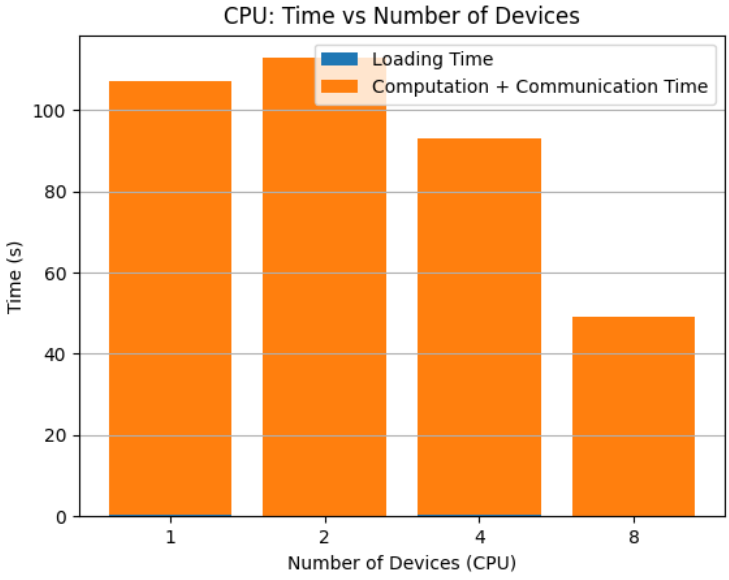

Figure 2: Throughput VS Number of devices (CPU)

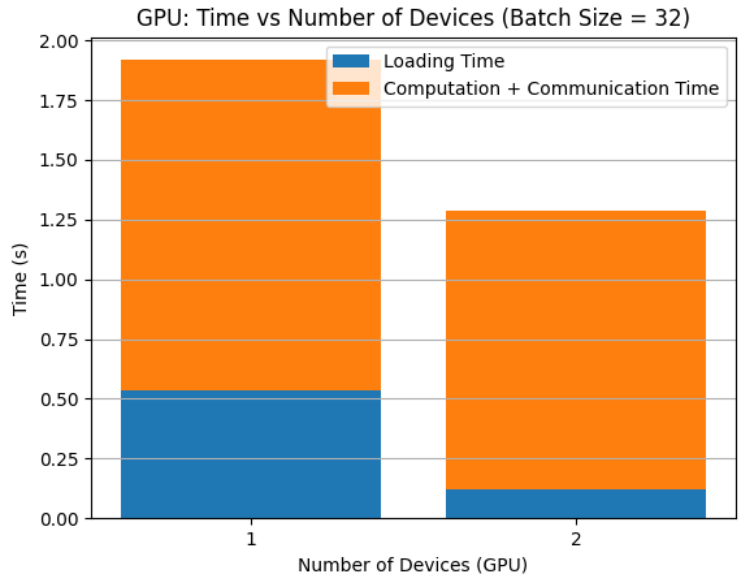

Figure 3: Time VS Number of devices (CPU)

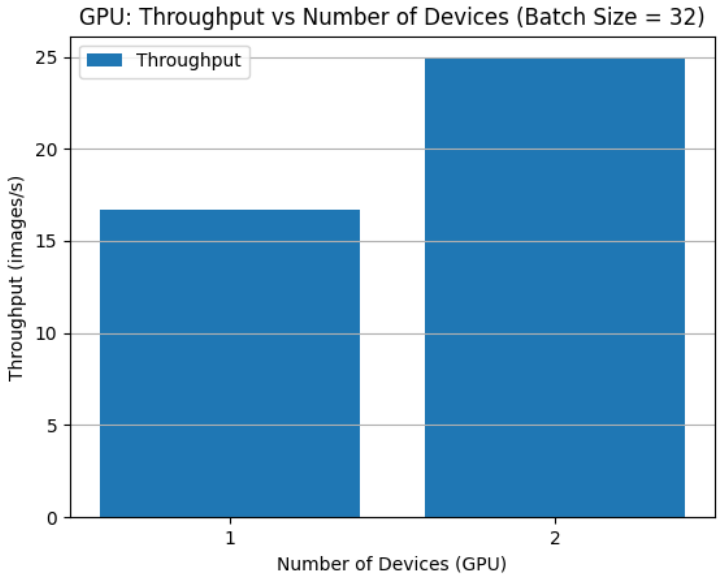

Figure 4: Throughput VS Number of devices (GPU)

Let us now move on to the analysis of these results.

6.2 Analyses

CPU results :

- **General Observation:**
  - The loading time decreases exponentially with the increasing number of CPU cores.
  - The computation time also decreases but remains significantly higher compared to the loading time.
  - The throughput increases.
- **Detailed Analysis:**
  - At 1 core, the computation time is high (106.91 s), and the loading time is 0.30 s.
  - At 8 cores, the computation time drops significantly to around 49 s (a reduction of nearly 54
  - Interestingly, for 2 cores, the total time is slightly higher than with 1 core. This anomaly can likely be attributed to the overhead of inter-core communication. With only 2 cores, the communication cost may outweigh the benefits of parallel computation, resulting in reduced performance compared to a single-core setup.
- **Conclusion:**
  - Increasing the number of CPU cores improves performance exponentially. The observed anomaly at 2 cores highlights the importance of balancing communication overhead and computation efficiency in parallel systems.

GPU results :

- **General Observation:**
  - The loading time significantly decreases with the addition of a second GPU.
  - The computation time also decreases, though the reduction is less pronounced than for loading time.
  - The The throughput significantly increases.
- **Detailed Analysis:**
  - For 1 GPU, the computation time is 3.12 s for a batch size of 64, while the loading time is 1.84 s.
  - With 2 GPUs, the computation time drops to approximately 1.33 s, and the loading time reduces to 0.27 s.
- **Conclusion:**
  - Adding a second GPU greatly reduces execution times, though the improvements are not strictly linear.

7 Impact of the batch size

GPU tests were performed with 1 and 2 GPU cores, as well as with different batch sizes (32 but also 16, 64 and 128).

For some reason, the tests did not work with a batch size of 128 with one GPU core.

7.1 Plots results

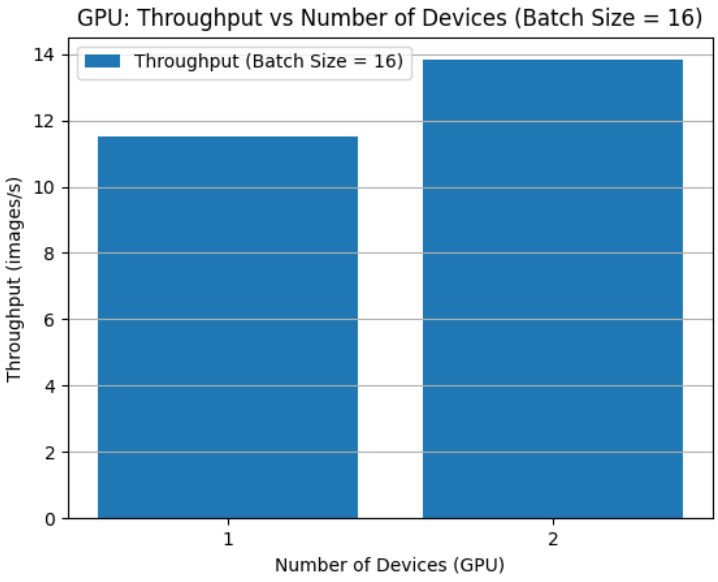

Figure 5: Throughput VS Number of devices (GPU with batch size 16)

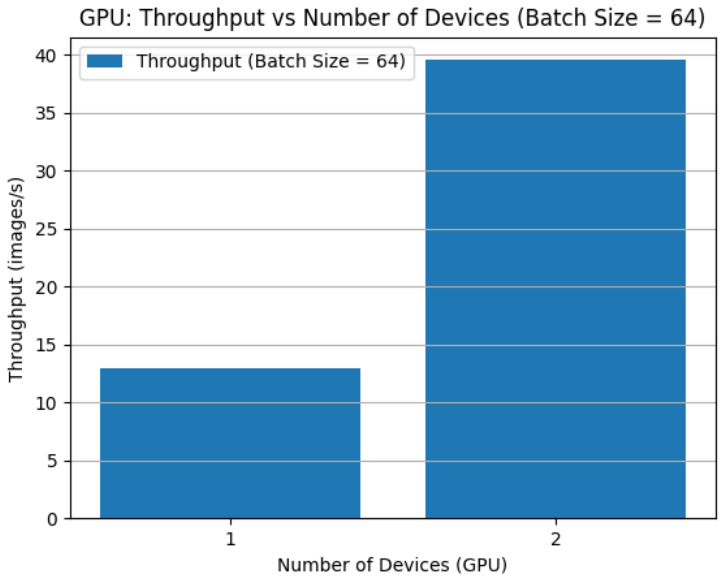

Figure 6: Throughput VS Number of devices (GPU with batch size 64)

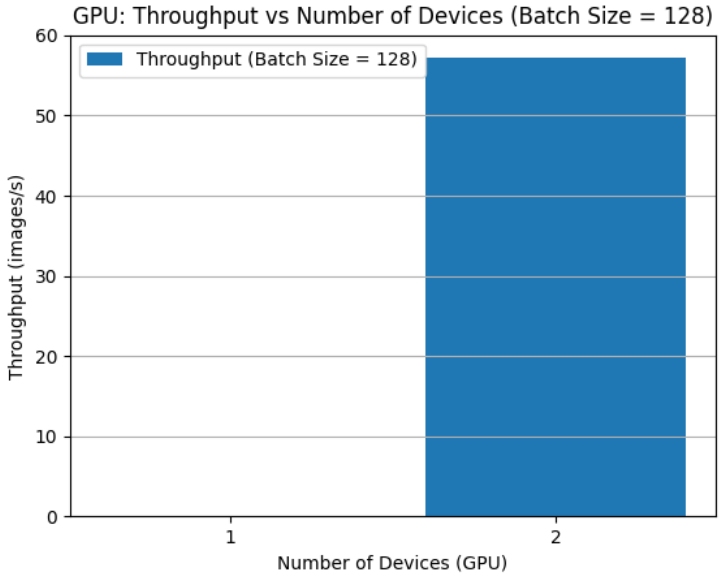

Figure 7: Throughput VS Number of devices (GPU with batch size 128)

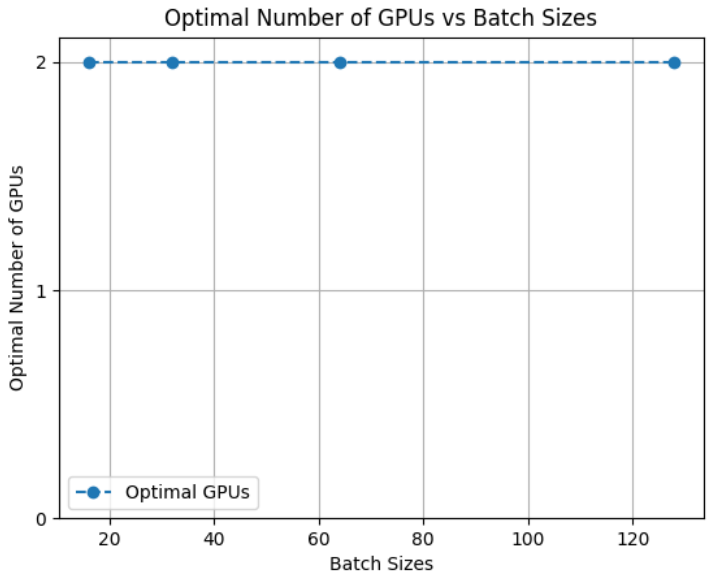

Figure 8: Optimal number of gpus VS batch sizes)

7.2 Analyses

Throughput vs Number of Devices for Batch Sizes 16, 64, 128 :

- **General Observation:**
  - Throughput increases with batch size, but efficiency diminishes for larger sizes.
- **Detailed Analysis:**
  - **Batch Size 16:** Throughput increases from 12.1 images/s (1 GPU) to 26.7 images/s (2 GPUs).
  - **Batch Size 64:** Throughput sees a more significant rise, reaching around 36.8 images/s with 2 GPUs.
  - **Batch Size 128:** Performance plateaus around 58 images/s with 2 GPUs with 2 GPUs. I was unable to do the tests with 1 GPU.
- **Conclusion:**
  - The larger the batch size, the larger the throughput.

Optimal Number of GPUs vs Batch Sizes :

- **General Observation:**
  - The optimal number of GPUs remains at 2 for all tested batch sizes.

- **Detailed Analysis:**
  - Performance gains are substantial between 1 and 2 GPUs, but beyond that, the available data does not provide insights into further scalability.
- **Conclusion:**
  - The results suggest that 2 GPUs are sufficient to efficiently leverage the Swin-B model under the tested conditions.

8 Conclusion

The tests conducted with the Swin-B model highlight the importance of parallel scalability for intensive computations. Performance increases significantly with the number of devices (CPU or GPU). GPUs offer remarkable performance gains, particularly with 2 devices, which appear sufficient for typical environments. The anomaly observed at 2 CPU cores underscores the need to account for communication overhead in parallel computing setups.

9 Appendix

The plots are in the "plots" directory of the archive. The raw results as CSV files are in the "results" directory of the archive. "results_cpu.csv" lists the results on the CPU node, "results_gpu.csv" on the GPU node. The plots generation and test scripts are in the "scripts" directory of the archive.

References

'

Demo code - https://gitlab.inria.fr/chxu/pytorch_exercice/-/tree/main/project?ref_type=heads

List of models - https://pytorch.org/vision/0.9/models.html

Introduction of the cluster NEF - https://wiki.inria.fr/ClustersSophia/Clusters_Home

Information on the Hardware - https://wiki.inria.fr/ClustersSophia/Hardware

NEF OAR Cluster nodes - https://nef-frontal.inria.fr/monika