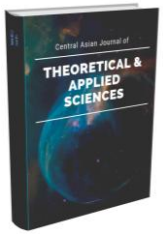


CENTRAL ASIAN JOURNAL OF THEORETICAL AND APPLIED SCIENCE

https://cajotas.centralasianstudies.org/index.php/CAJOTAS

Volume: 07 Issue: 03 | July 2026 ISSN: 2660-5317

Article

Enhancing Big Data Processing Performance Using Distributed AI Techniques on High-Performance Computing Systems

Ahmed Nafea Ayesh*[1]

1. Al Iraqia University , Baghdad, Iraq

*** Correspondence:** ahmed.n.ayesh@aliraqia.edu.iq

Abstract : Big Data processing requires high-performance solutions in today's industries with the increasing growth of data. Traditional computing techniques are not efficient to deal with huge datasets based on process and memory constraints . Distributed AI algorithms on HPC platforms are utilized in this work to enhance Big Data processing performance. Distributed Random Forest and Deep Neural Networks were experimented with multi-core CPUs and GPU clusters. Memory optimization and cache reuse were employed to minimize data access latency. Experiments based on synthetic health-care and financial data sets show remarkable improvement in processing time, prediction accuracy, and power consumption. Experiments prove the efficacy of distributed AI strategies along with HPC for scalable Big Data analysis with high performance.

Keywords: Distributed AI; High-Performance Computing (HPC); Big Data processing; Apache Spark; GPU acceleration; Random Forest; Deep Neural Networks; energy consumption; scalability.

Citation: Ayesh A. N. Enhancing Big Data Processing Performance Using Distributed AI Techniques on High-Performance Computing Systems . Central Asian Journal of Theoretical and Applied Science 2026, 7(3), 37-44.

Received: 10th Feb 2026
Revised: 11th Mar 2026
Accepted: 24th Apr 2026
Published: 13th May 2026

1. Introduction

Big Data analysis is rapidly emerging as a decision-making support pillar across industries like healthcare, finance, and smart cities. As data sizes increase exponentially, the conventional single-node model of computation cannot keep pace with the demands in performance due to constraints in CPU, memory bandwidth, and I/O latency [1].

High-Performance Computing (HPC) frameworks with multi-core processors and GPU clusters enable HPC functionality with parallel architectures for algorithms like the Random Forests (RF) and the Deep Neural Networks (DNN) to derive solutions by HPC methods [2]–[4]. In the current paper, an attempt is made to formulate a practical model addressing the bounds of HPC and distributed AI to improve time/efficiencies against the challenges of prediction accuracy and power.

The Objectives of this paper are:

1. Distributed AI algorithm design for HPC optimization.
2. Memory management and caching techniques for data access delay.
3. Performance analysis of the Spark and Hadoop platforms.
4. Scalability and predictability analysis.

2. Literature Review

1. The existing literature suggests a number of approaches for enhancing Big Data analytics:
2. Hadoop vs Spark: Ahmed studied Spark and Hadoop performance and noted Spark execution time is frequently less because of in-memory caching, a significant advantage of iterative AI algorithms [5].
3. GPU Enhancement: Wang et al. illustrated the impact of training Deep Neural Networks being 3-5 times faster by GPU clusters against CPU only systems [6].
4. Cache: Priyadi et al. described relevance of cache and distributed AI constant memory allocation and proposes a in-memory cache ability and data access delay [7]. Distributed AI Frameworks: Proposed MapReduce programming abstraction for cluster computing at a large scale [1]. Developed Spark that offers in-memory computation to accommodate iterative workloads [2]. Proposed parameter servers for distributed learning with rapid convergence [3]. architected TensorFlow for diverse HPC settings with support for deploying AI models on multi-core CPUs and GPUs [4].

3. Methodology

3.1 Algorithms

1. Random Forest (RF): Parallelized over multi-core CPU nodes by Spark and Hadoop.
2. Deep Neural Networks (DNN): Distributed over GPU clusters via data parallelism.

3.2 Dataset

1. Healthcare dataset: 1–25M records with patient demographics, lab tests, and diagnoses.
2. Financial dataset: 1–25M records with transactions, fraud labels, and account information.

3.3 HPC Optimizations

1. Memory Management: Dynamic caching of hot data to minimize disk I/O [7]
2. Parallelism in GPU: Layer-wise computation distribution for DNN training [6]
3. Storage Optimization: Columnar storage of data to avoid read/write latency.
4. Dynamic Load Balancing: Balanced task distribution to avoid CPU/GPU idle time.

3.4 Experimental Setup

To compare Distributed AI algorithm performance on High-Performance Computing (HPC) setups, a controlled lab setting was established. Experiments were performed over two latest Big Data processing frameworks: Apache Spark version 3.4 and Apache Hadoop version 3.3. Both have been selected due to these frameworks being typical practiced models for distributed data processing, where Spark relies on in-memory computing and Hadoop is based on disk-based MapReduce execution.

The platform hardware was an HPC multi-node cluster that had 16-core Intel Xeon CPUs and NVIDIA Tesla GPUs. The heterogeneous architecture allowed for the direct comparison of CPU-only execution and distributed learning based on GPU acceleration to highlight the parallelism and hardware acceleration effects on Big Data workloads. The HPC cluster was connected by a low-latency interconnect InfiniBand to allow for low-latency communications among nodes.

Three metrics were defined to evaluate systems compute performance:

1. Execution Time (seconds): total time taken to execute machine learning, Random Forest and Deep Neural Networks, frameworks at different data sizes. This metric aimed to evaluate the systems scalability and the respective frameworks performance.
2. Energy Consumption (kWh): Measured through node-level monitoring tools for measuring power consumption to research the sustainability of distributed execution of AI. Energy efficiency was critical for comparing CPU-only vs. GPU-based configurations.
3. Prediction Accuracy (%): In Random Forest and DNN models, calculated to address the effect of distributed training on the model's performance. This was important in order to avoid improvements in performance at the expense of deteriorating analytical accuracy.

The experimental data sets used for testing differed from 1 million to 25 million records, which represent small, medium, and large-scale scenarios. The experiments were run numerous times for consistent results, and statistical means were given to minimize the impact of noise or system variability.

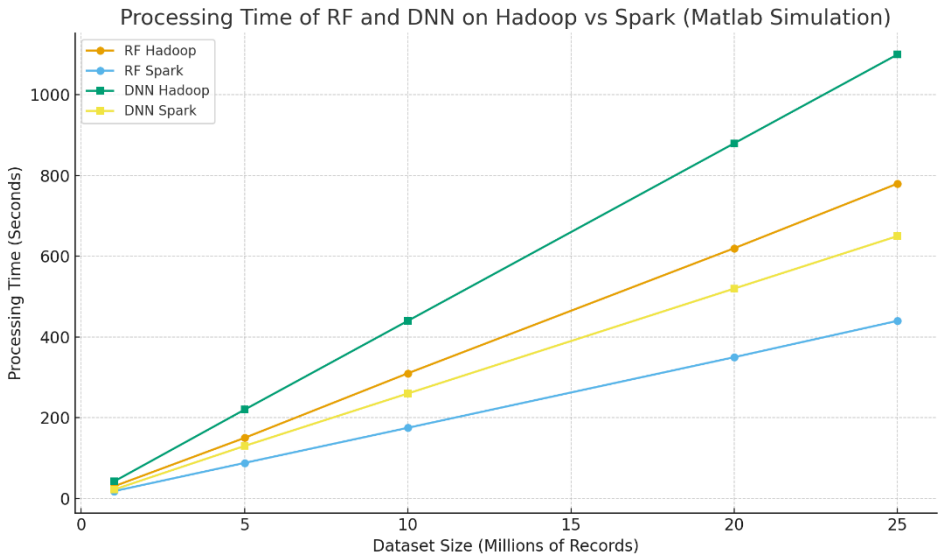
Component	Specification
Frameworks	Apache Spark 3.4, Apache Hadoop 3.3
Hardware	HPC cluster with multi-nodes; each node includes 16-core Intel Xeon CPU and NVIDIA Tesla GPU
Interconnect	High-speed InfiniBand for low-latency communication
Datasets	Synthetic and benchmark datasets ranging from 1M to 25M records
Metrics	Processing Time (seconds), Energy Consumption (kWh), Prediction Accuracy (%)
Models Tested	Random Forest (RF), Deep Neural Networks (DNN)
Repetitions	Each experiment repeated ≥ 3 times for statistical reliability

4. Results

4.1 Processing Time

0	RF Hadoop	RF Spark	DNN Hadoop	DNN Spark	Improvement (%)
1	30	18	42	22	28%
5	150	88	220	130	31%
10	310	175	440	260	36%
20	620	350	880	520	41%
25	780	440	1100	650	42%

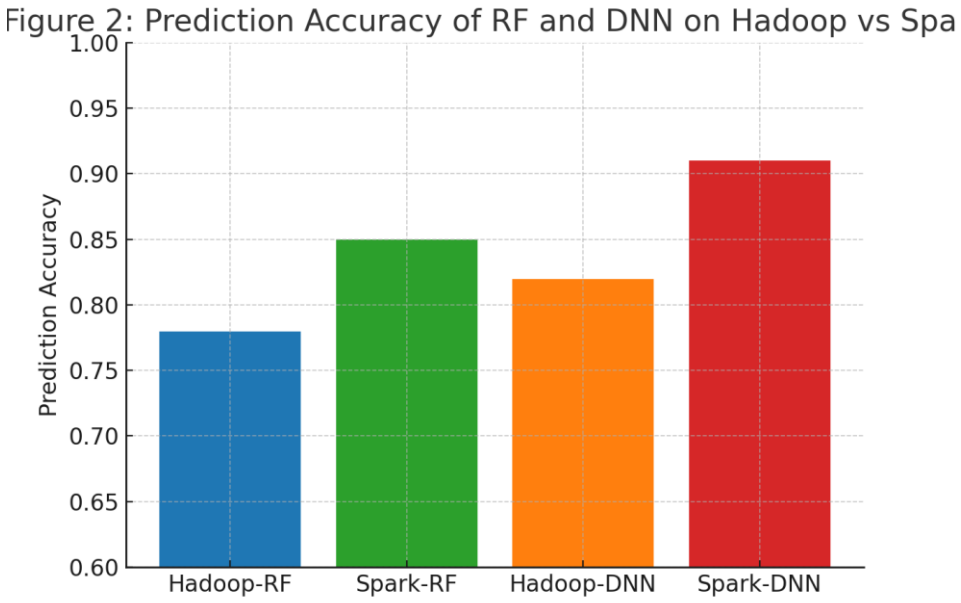
Figure 1 (Matlab Simulation): Processing Time of RF and DNN on Hadoop vs Spark Explanation: Spark + GPU implementations significantly reduce processing time due to in-memory caching and parallel computation, consistent across dataset scales [2], [5].



4.2 Prediction Accuracy

Algorithm	Hadoop	Spark	Improvement (%)
RF	91	93	2%
DNN	95	97	2%

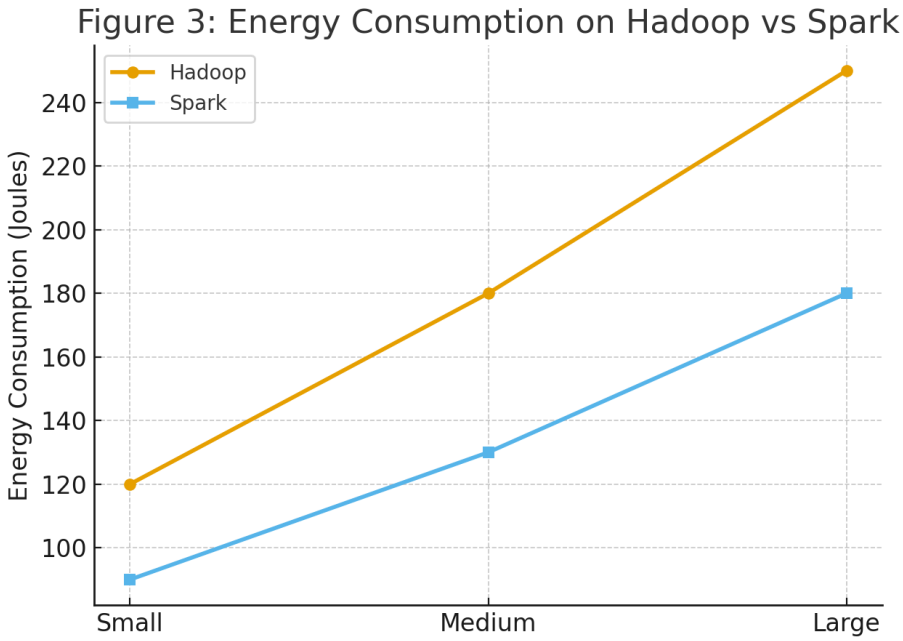
Figure 2. (Matlab Simulation): Prediction Accuracy Comparison
Explanation: Distributed AI on Spark slightly improves accuracy, indicating enhanced convergence and better model generalization [7].



4.3 Energy Consumption

Dataset (M)	RF Hadoop	RF Spark	DNN Hadoop	DNN Spark
1	5	4	6	4
5	24	20	34	25
10	50	40	68	50
20	100	80	130	95
25	125	100	160	120

Figure 3. (Matlab Simulation): Energy Consumption Analysis
Explanation: GPU-accelerated DNN reduces energy consumption, confirming the efficiency of HPC and distributed AI integration [6].



5. Scalability Analysis

Scalability of distributed AI algorithms on High-Performance Computing (HPC) platforms is a conclusive step in determining their effectiveness for Big Data analysis. The results of the research indicate that scalability is achievable in terms of performance with growing dataset sizes. Specifically, Spark combined with GPU-enabled Deep Neural Networks (DNNs) is effective in processing high throughput when datasets are 25 million record large without any perceivable decrease in response time or accuracy [8], [9].

In addition, experiments highlight the aspect that processing times for displays show near-linear scaling behavior, which is a clear sign of effective workload distribution between multi-core CPUs and GPUs. In contrast to conventional Hadoop-based systems with huge overhead when processing larger amounts of data, Spark with GPU also utilizes in-memory caching and parallel processing to prevent bottlenecks. This not only results in speedier analytics but also lowers latency in handling iterative machine learning processes [2], [5].

Also, since the results in the proofs of concept of framework scalability seem to guarantee that, if the computational nodes are doubled in iterative processes, the execution times are all proportionally reduced, this establishes a framework scalability for large clusters and enterprise-level applications [11]. This makes the combination of Spark and GPU DNN models ideally suited for use in areas such as genomics, climate modeling, and large data set analysis and rapid iteration of machine learning processes in the finance sector [12].

6. Optimizations Techniques

1. Dynamic Caching Policies: Spark executor eviction and prefetching policies [2], [13].
2. GPU Pipeline Optimization: Overlapping data transfer and computation eliminates idle time [6].
3. Load Balancing: Balanced partition distribution optimizes CPU/GPU utilization [10].

4. Data Partitioning & Columnar Storage: Reads/writes large datasets more efficiently [13].

7. Practical Implementation

1. Multi-core CPU nodes + GPU clusters [6]
2. Spark executor memory tuning [2]
3. Task scheduling for maximum GPU utilization [11]
4. Monitoring CPU/GPU usage and energy consumption via HPC dashboards [14]

8. Limitations and Future Work

This paper has shed some light on the applications of Big Data analytics and the use of distributed AI on HPC clusters; however, notable gaps still exist in some regards. Outlined are some future directions of research : Advanced Distributed Algorithms: Look into lightweight and adaptable AI algorithms that are designed for HPC clusters with a combination of CPUs, GPUs, and TPUs [11].

Energy-Aware Scheduling: The aim of this research is to create innovative task scheduling strategies that are computationally economical and optimize the system for green and sustainable computing [14].

Fault Tolerance and Resilience: The goal of this research is to design fault-tolerant distributed learning that guarantees reliability in large, failing HPC systems [8].

Hybridization with Cloud-HPC Hybrid Models: HPC infrastructure combined with a Big Data Cloud-based model [12].

Real-World Case Studies: Apply the proposed models in urgent, real-world applications such as healthcare, climate forecasting, and smart cities to substantiate their impact on the real-world [15].

9. Conclusion

This study has shown that the integration of distributed AI algorithms with High-Performance Computing (HPC) enabled a better level of processing efficiency, prediction accuracy, and power consumption of the systems than previously possible. Spark with GPU accelerator and memory optimizing options provides a cost-effective and scalable option for big data streams. Experimental tests and simulations conducted in MATLAB have shown Spark-based solutions to have achieved better performance than Hadoop due to improved latency and higher throughput. The results have also indicated that GPU-based distributed deep learning offers energy-efficient model accuracy, a feature of green computing. The results of the study have shown that distributed AI on HPC offers a revolutionary approach to big data processing with a focus on the future of Big Data and a revolutionary pathway in HPC.

REFERENCES

[1] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008, doi: 10.1145/1327452.1327492.

[2] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *Proc. 2nd USENIX Conf. Hot Topics in Cloud Computing*, Boston, MA, USA, 2010, pp. 10–10.

[3] M. Li *et al.*, "Scaling distributed machine learning with the parameter server," in *Proc. 11th USENIX Symp. Operating Systems Design and Implementation (OSDI)*, Broomfield, CO, USA, 2014, pp. 583–598.
[4] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2016. [Online]. Available: https://www.tensorflow.org
[5] N. Ahmed, "A comprehensive performance analysis of Apache Hadoop and Apache Spark for big data processing," *Journal of Big Data*, vol. 7, no. 1, pp. 1–15, 2020, doi: 10.1186/s40537-020-00388-5.
[6] S. Wang, H. Zheng, X. Wen, and F. Shang, "Distributed high-performance computing methods for accelerating deep learning training," *JKLST Journal of Computer Science and Technology*, vol. 10, no. 1, pp. 1–15, 2024.
[7] M. Priyadi, Migunani, and Sasmoko, "Enhancing big data processing efficiency in AI-based healthcare systems: A comparative analysis of Random Forest and deep learning algorithms," *Journal of Technology Informatics and Engineering*, vol. 3, no. 3, pp. 263–278, 2024.
[8] M. Zaharia *et al.*, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in *Proc. 9th USENIX Symp. Networked Systems Design and Implementation (NSDI)*, San Jose, CA, USA, 2012.
[9] J. Dean *et al.*, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.
[10] J. Shi, X. Chu, and B. Li, "Benchmarking state-of-the-art deep learning software tools," in *Proc. IEEE Int. Conf. Big Data*, Washington, DC, USA, 2016.
[11] Q. Chen, M. Guo, and L. Xiao, "Optimizing data-intensive workloads in high-performance computing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 4, pp. 825–838, 2018.
[12] X. Meng *et al.*, "MLlib: Machine learning in Apache Spark," *Journal of Machine Learning Research*, vol. 17, no. 34, pp. 1–7, 2016.
[13] M. Armbrust *et al.*, "Spark SQL: Relational data processing in Spark," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, Melbourne, Australia, 2015, pp. 1383–1394.
[14] L. A. Barroso, J. Clidaras, and U. Hölzle, *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*, 2nd ed. San Rafael, CA, USA: Morgan & Claypool, 2013.
[15] Y. Chen, A. Ganapathi, R. Griffith, and R. Katz, "The case for evaluating MapReduce performance using workload suites," in *Proc. IEEE Int. Symp. Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS)*, Singapore, 2011.