International Journal of Data Informatics and Intelligent Computing (IJDIIC)
Vol. 5, No. 2, June 2026, pp. 50~61
DOI: 10.59461/ijdiic.v5i2.273

Optimized Large-Scale Data Analytics Using Advanced Computational Methods in Cloud Computing Frameworks

Aditya Gupta[1], Sai Kiran Oruganti[2]
[1] Department of Computer Science and Engineering, Lincoln University College, Petaling Jaya, Selangor Darul Ehsan, 47301, Malaysia.
[2] Faculty of Engineering and Built Science, Lincoln University College, Petaling Jaya, Selangor Darul Ehsan, 47301, Malaysia.

Article Info

Article history:

Received March 30, 2026
Revised May 28, 2026
Accepted June 09, 2026

Keywords:

Large-Scale Data Analytics
Computational Methods
Cloud Computing
Workload Distribution
Capsule Networks for Cloud Analytics

ABSTRACT

In today's data-driven world, the exponential growth of information from diverse sources necessitates robust, scalable, and efficient computational solutions. Large-scale data analytics is pivotal in extracting valuable insights from massive datasets, facilitating informed decision-making across industries. This research proposes an advanced computational framework that integrates Deep Learning (DL) techniques to optimize large-scale data analytics in cloud computing environments. The proposed system leverages the Adaptive Satin Bowerbird Optimizer-driven Dynamic Capsule Network (ASB-Dynamic-CapsNet) to automate workload distribution while minimizing execution time and resource utilization. The data collection process involves gathering large-scale datasets from multiple sources, including cloud platforms, Internet of Things (IoT) devices, transactional systems, social media, and enterprise databases. Preprocessing steps like handling missing values and standardizing data ensure data quality and consistency. The framework is evaluated using key performance metrics like Central Processing Unit (CPU) utilization (86%), Random Access Memory (RAM) utilization (68%), and training accuracy (95.9%). Experimental results demonstrate that the ASB-Dynamic-CapsNet model outperforms alternative scheduling approaches and significantly improves the performance of CPU, RAM, and training accuracy compared to traditional algorithms. Overall, the findings highlight the efficacy of DL-driven scheduling in optimizing cloud-based data analytics, providing a scalable and efficient solution for high-volume workloads in modern computing environments.

Corresponding Author: Aditya Gupta (e-mail: gupta.aditya56@gmail.com)

1. INTRODUCTION

Most monitoring systems were produced for a specific kind of equipment in the early stages of condition monitoring technology development, and each system was dispersed and isolated. The information lacked connectivity and data exchange, making it difficult to manage and effectively evaluate monitoring data [1]. Organizations are filled with enormous volumes of data produced by a variety of sources, including social media, IoT devices, and transactional systems, in the age of big data. For strategic planning and well-informed decision-making, it is essential to be able to gather valuable insights from the data. Accordingly, effective query processing has become essential to large-scale database administration [2]. Technology has always been seen positively, from the introduction of personal computers to the use of radio and television. The potential to transform education was present in every breakthrough. However, the real potential of technology resides in its capacity to enhance and supplement current teaching approaches rather than essentially replacing them [3]. For cloud-based infrastructures to perform reliably and efficiently, cloud monitoring is essential. The many components of cloud services, including resource use, network latency, and security incidents, require methodical data collection, evaluation, and visualization. From a third-party perspective, cloud monitoring solutions are essential for businesses that depend on key information on the

efficacy and condition of cloud-based services and apps [4]. The IoT relies heavily on cloud and edge infrastructures to deliver particular services and practical applications across many application domains. IoT makes all physical items sentient so they can connect, interact, and make intelligent decisions from their perspective. It also automates all areas of life that are considered to be in the public domain [5]. The IoT, edge computing, and smart cities are some of the recent innovations made possible by cloud computing, one of the biggest changes in contemporary Information and Communication Technology (ICT) services. Cloud computing models transform computing from a product that can be purchased to a service that can be delivered via the Internet [6].

1.1. Research Objective

The research aims to optimize large-scale data analytics in cloud computing by incorporating DL techniques for proficient workload distribution. The implemented model minimizes execution time and resource utilization, improving scalability and performance in high-volume cloud applications.

2. LITERATURE REVIEW

A Hybrid Harris Hawks Optimization (HHHO) with K-means and MapReduce was proposed for large-scale data clustering. K-means initialized solutions, while HHO refined them for better optimization. MapReduce ensured parallelization, fault tolerance, and load balancing. The method, implemented in Python, outperformed existing clustering techniques. Numerical evaluations confirm its efficiency in accuracy and error reduction. Future research included DL integration and real-time processing [7]. The objective was to optimize big data processing pipelines in cloud environments by reducing latency, cost, and resource consumption. The research explored AI-driven adaptability, parallelism, and self-scheduling to enhance throughput and operational efficiency. The proposed strategies reduce processing delays, improve scalability, and optimize resource utilization. Comparative evaluation and year-wise performance evaluations confirm their effectiveness in real-world cloud environments. Further research should focus on enhancing resilience, scalability, and real-time adaptability in cloud-based data processing [8]. The role of cloud computing in covering its evolution, service models, deployment strategies, and key platforms was explored. It systematically examined cloud service and deployment models. It helped stakeholders make informed decisions and effectively leverage cloud computing for optimized data processing.

Future research should focus on advancing multi-cloud strategies and optimizing cloud platforms for scalable, real-time data analytics [9]. IoT applications generate vast sensor data, requiring high-speed computation beyond device capabilities. Computation offloading to nearby devices, fog, or cloud helped meet performance needs like time sensitivity and energy efficiency. The research explored context-aware offloading for optimizing IoT services. Various frameworks were evaluated and compared. The research highlighted their strengths and limitations in gathering computation demands. Future directions focus on enhancing offloading strategies for efficient IoT performance [10]. Protein interactions were crucial for understanding protein functions but posed challenges in large-scale analysis. The approach re-employed the Co-Fex algorithm using MapReduce for efficient prediction. A tree-based data structure reduced memory consumption, and distributed processing enhanced scalability. Experiments show a two-order magnitude improvement in efficiency while maintaining accuracy. The framework proved effective for large-scale analysis [11]. Cloud deployment architectures were a preferred model for large data operations. However, data was rarely controlled by users, and security issues surfaced.

The research presented a security-by-design framework for secure large data deployment in cloud environments. It integrated systematic security analysis and automated security assessment to enhance protection. The framework was validated through an Apache Hadoop use case, demonstrating improved security awareness and reduced design time. Open challenges in Big Cloud security and optimization can be addressed in further research [12]. While cloud computing supports the reproducibility of computational experiments, challenges related to automation and interoperability across different cloud platforms were encountered. The research leveraged serverless computing, containerization, and an adapter design pattern to address these issues. An open-source toolkit enabled automated execution, data storage, and seamless reproduction across cloud environments. Experiments on Amazon Web Service (AWS) and Azure confirm its efficiency, scalability, and reproducibility. Results show improved execution performance for Big Data analytics. The adaptability and optimized performance can be enhanced in further research [13].

2.1. Research Gap

The exponential growth of big data, cloud computing, and IoT demands competent, scalable, and protected processing frameworks. Traditional clustering and workload distribution methods resist optimization and real-time compliance. Security vulnerabilities in cloud-based data environments delay widespread implementation. Cross-cloud reproducibility remains a challenge, resisting seamless

interoperability. The research improves scalability, effectiveness, and security through AI-driven optimization and security-by-design frameworks for real-time data analytics. The implemented approach overcomes the limitations of the research and provides better performance than traditional methods.

2.2. Key Contributions

- The research proposed the ASB-Dynamic-CapsNet Framework: Introduced a DL-driven approach to optimize workload distribution in cloud computing.
- Enhanced Computational Efficiency: Improved CPU and memory utilization while ensuring higher model training accuracy compared to traditional scheduling methods.
- Improved Scalability and Performance: Reduced execution delays and optimized resource allocation for efficient processing of large-scale cloud workloads.

3. METHODOLOGY

The research integrates ASB-Dynamic-CapsNet to optimize workload distributions in cloud computing. Large-scale data is collected from an open source and preprocessed using Z-score normalization to ensure quality. The proposed model efficiently allocates resources, reducing execution time and system bottlenecks. Performance is evaluated using CPU utilization, RAM utilization, and training accuracy over various epochs. Figure 1 represents the entire flow of the proposed approach.

- Data collection: The dataset from Kaggle replicates real-world cloud environments, enabling workload scheduling analysis and optimization.
- Data Preprocessing: Z-score standardization enhances data reliability and computational efficiency, while handling missing values improves data quality and ensures consistent performance.
- ASB-Dynamic-CapsNet: A hybrid model comprising ASB for optimization and CapsNet for spatial feature learning in workload scheduling. ASB ensures well-organized resource allocation, while CapsNet increases feature extraction and decision-making in cloud-based data analytics.

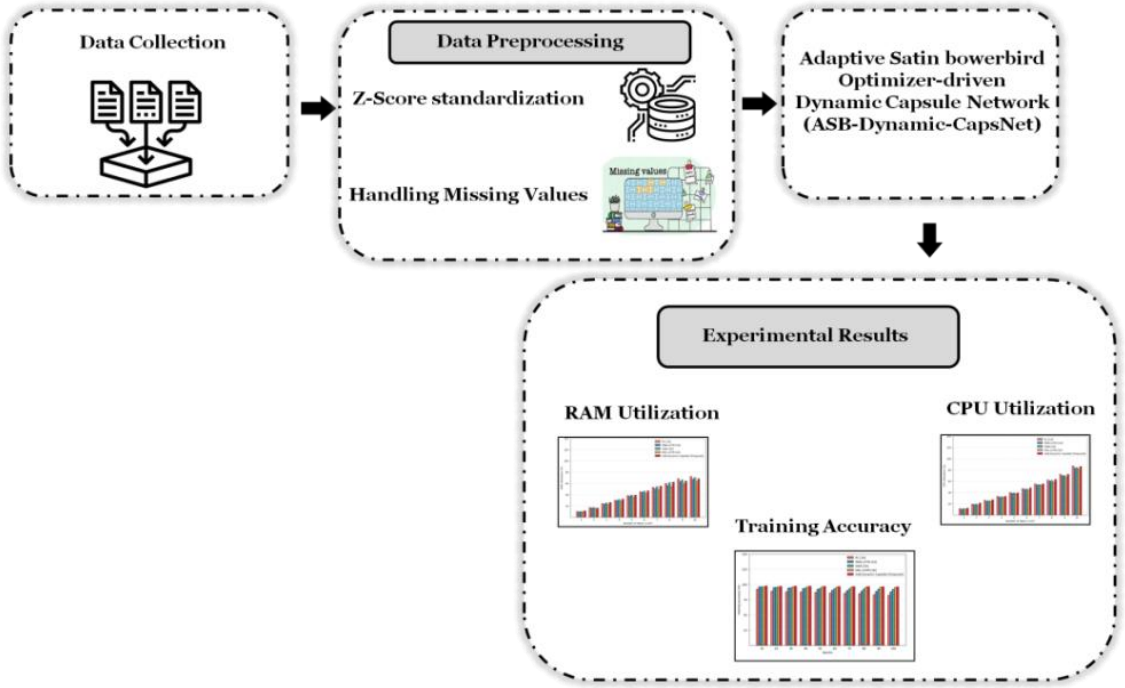

Figure 1. Flow of the proposed method

3.1. Data Collection

The cloud workload dataset for scheduling analysis was collected from the open-source Kaggle website: https://www.kaggle.com/datasets/zoya77/cloud-workload-dataset-for-scheduling-analysis. The dataset is intended for use in DL-based computational techniques for workload scheduling optimization and

large-scale cloud workload analytics. By recording job execution data, resource use, and scheduling tactics, it replicates real-world cloud settings. In cloud computing frameworks, the dataset might be used to assess scheduling efficiency, optimize task allocation, and reduce execution delays.

3.2. Data preprocessing

The procedure of cleaning, transforming, and organizing raw data to improve its value and usability is data preprocessing. In cloud computing, intelligent preprocessing techniques handle missing values and apply Z-score standardization to ensure reliability. The steps improve workload distribution and avoid resource allocation. Feature selection and dimensionality reduction promote optimized computational effectiveness. By incorporating the techniques, the implemented model guarantees increased system performance and resource utilization.

3.3. Z-Score Standardization

The extensively utilized data standardization technique was Z-score standardization [14]. In the context of the research, Z-score standardization is applied to preprocess large-scale datasets collected from cloud platforms, IoT devices, and enterprise databases. Standardizing features before feeding into the proposed model ensures consistent data scaling, preventing dominant features from skewing workload distribution. This contributes to optimizing computational resources, reducing CPU and memory consumption, and enhancing overall system throughput in high-volume cloud computing environments. Equation (1) presents the z-score standardization for a feature x'.

$$x' = \frac{x - mean(x)}{std(x)} \tag{1}$$

The original value is denoted by x, the normalized value by x', the mean value of the feature x by $mean(x)$, and the standard deviation of x by $std(x)$. Compared to other standardization techniques, Z-score standardization has the notable benefit of being resilient against outliers. Extreme values that deviate greatly from the majority of data points are known as outliers, and the values have the potential to skew statistical characteristics and impact how workload is distributed in cloud computing. Applying Z-score normalization ensures balanced scaling, preventing such distortions and enhancing the efficiency of the proposed technique.

3.4. Handling Missing Values

Ensuring data completeness is vital for accuracy and reliability in large-scale data analytics. Missing data, caused by factors like manual errors, sensor failures, or inconsistent collection from cloud and IoT sources, can reduce computational efficiency. The proposed model addresses the limits through robust preprocessing, categorizing missing data into three types to enhance DL-driven scheduling in a cloud environment.

- Missing Completely at Random (MCAR): The absence of data occurs independently of other variables, meaning the probability of missingness is uniform across the dataset. While MCAR data does not introduce systematic bias, its unpredictability can impact model training in large-scale analytics.
- Missing At Random (MAR): Missing values in a variable depend on observed data from other variables but are not influenced by the missing values themselves. The characteristic permits imputation techniques to approximate and improve lost data based on the relationships between observed features in the dataset.
- Not Missing at Random (NMAR): Missing values are scientifically related to unobserved variables, making the values challenging to estimate using available data. NMAR scenarios need advanced techniques, such as DL-based imputation or adaptive models, to reduce data loss effects.

By incorporating strong preprocessing techniques in the implemented technique, the framework ensures high-quality data input, thus enhancing the evaluated metrics. The capability to handle missing values efficiently improves model performance, supporting scalable and adaptive workload scheduling in cloud computing environments.

3.5. Adaptive Satin Bowerbird Optimizer-driven Dynamic Capsule Network (ASB-Dynamic-CapsNet)

The proposed ASB-Dynamic-CapsNet integrates the ASB with a Dynamic CapsNet to improve workload distribution in cloud computing. The ASB efficiently searches for optimal scheduling solutions, reducing resource consumption, while the Dynamic CapsNet conserves spatial relationships in compound

data patterns for precise task execution. The hybrid approach guarantees adaptive learning, dynamically adjusting to workload deviations. By leveraging the included strengths of DL, the framework achieves better scalability, efficiency, and task prioritization in cloud-based data analytics.

3.6. Capsule Network: Evaluation and Optimization

CapsNet was intended to encode spatial hierarchies by utilizing vector-based neurons called capsules. The concept started with Transforming Auto Encoders (TAEs), which need external transformation matrices. Later, Dynamic Routing CapsNet was introduced for internal pose estimation, followed by matrix capsules that improved accuracy while reducing computational complexity. CapsNet [15] optimization focuses on enhancing efficiency and scalability. Dynamic routing mechanisms develop how capsules interrelate, while metaheuristic algorithms enhance routing decisions and workload circulation. The optimizations help in running large-scale data analytics while reducing execution time and resource consumption. Figure 2 shows the general architecture of CapsNet.

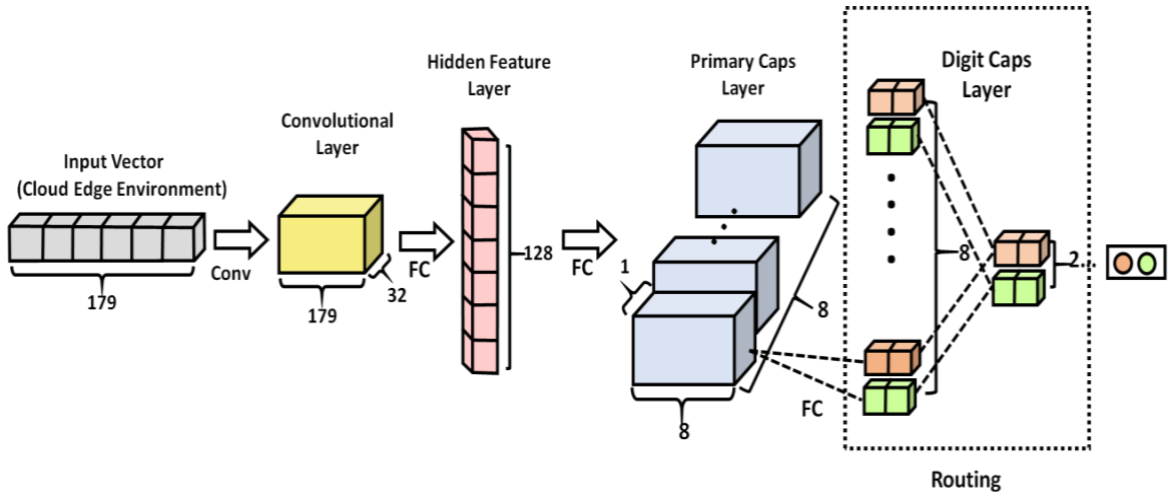

Figure 2. Capsule Network Architecture

3.7. Dynamic Capsule Network (CapsNet)

CapsNet with dynamic routing provides a viable option to perform large-scale data analytics while preserving spatial relations in data. Unlike traditional approaches, CapsNet uses vector-based representations to encode information hierarchically for better feature extraction and classification accuracy. The activation function applied to capsules is a non-linear squashing function, ensuring that the output vector maintains an interpretable magnitude normalized in the range of [0, 1], as represented in equation (2).

$$u_i = \frac{||T_i||^2}{1+||T_i||^2} \cdot \frac{T_i}{||T_i||} \tag{2}$$

Where T_i represents the total input to capsule i; it guarantees that greater vector magnitudes relate to an increased probability that specific features in the data were present, which enhances the prediction accuracy made by computation models in the cloud. The dynamic routing mechanism connects the lower-layer capsules with those at higher layers and adjusts weights utilizing an iteratively derived consensus process, as stated in equation (3).

$$T_i = \sum_j d_{ji} \cdot \hat{v}_{ij} \ , \hat{v}_{ij} = X_{ji} \cdot v_j \tag{3}$$

Where d_{ji} presents coupling coefficients, and X_{ji} is a transformation matrix that ensures optimal feature alignment. The adaptability of any cloud-based analytics system is hence greatly enhanced by this mechanism, especially in those considered real-time data-processing and hierarchical feature-learning setups. Dynamic routing mechanisms based on CapsNet provide an influx of feature retention, minimal redundancy in computation, and realistic prediction accuracy development to the large-scale data analytics workflows running on cloud computing frameworks. It lends itself to computationally intensive optimization schemes in a cloud paradigm to enable feasible data-driven decision-making across distributed systems.

3.8. Adaptive Satin Bowerbird Optimizer (ASB)

The ASB optimizer is an evolutionary computation technique inspired by the nesting activities of satin bowerbirds. In the context of cloud-based large-scale data analytics, ASB is engaged in task scheduling and resource optimization. The effectiveness of a cloud computing system relies on effective workload distribution. The attractiveness of each solution is assessed using a fitness function, which quantifies its efficacy in optimizing cloud performance. The probability of selecting a solution was represented in equation (4).

$$probi = \frac{fiti}{\sum_{j=1}^{NP} fiti} \tag{4}$$

Where $fiti$ represents the efficiency scores of the jth scheduling strategy, and the total number of persons was denoted by NP. The fitness function is computed using equation (5).

$$fiti = \begin{cases} \frac{1}{1+f(w_j)} \; if \; f(w_j) \geq 0 \\ 1 + \left|f\left(w_j\right)\right| \; if \; f(w_j) < 0 \end{cases} \tag{5}$$

Where $f(w_j)$ represents the execution time and resource utilization associated with the jth scheduling strategy. Integrating ASB with CapsNet enhances the load-balancing and task-execution efficiency of the framework. ASB employs dynamic resource allocation for scheduling, while CapsNet maintains spatial data relationships for better feature learning. The fusion of metaheuristic optimization and DL yields a scalable, efficient cloud computing model, which outperforms traditional scheduling methods and provides superior advantages to large-scale data analytics.

4. PERFORMANCE EVALUATION

Evaluating the performance of cloud-based computational models is essential to ensure efficiency, scalability, and resource optimization. The proposed approach adopted several performance parameters. The experimental setup used primarily Python-based DL frameworks for its model training and optimization. The metrics compared were training accuracies, CPU utilization, and RAM retention against conventional methods like Reinforcement Learning (RL) [16], Recurrent Neural Network – Long Short-Term Memory (RNN-LSTM) [16], Deep Q-Network (DQN) [16], and Deep Reinforcement Learning (DRL-LSTM) [16]. The performance evaluation results validate the model's capability to optimize cloud resources while simultaneously performing at a high standard and being scalable.

4.1. Experimental setup

Python 3.8 on a Windows 11 system was used to implement the proposed method, which is operational with an Intel Core i7 processor. DL frameworks, such as TensorFlow and PyTorch, were used for training and optimization of the ASB-Dynamic-CapsNet model. Data preprocessing, which includes handling missing values, feature extraction, and transformation, was accomplished using NumPy and Pandas. Performance evaluation and visualization of the experimental results were done with Matplotlib and Seaborn.

4.2. Correlation Matrix

The correlation matrix is used to quantify the relationships between the metrics and provide insight into the mutual influence. The research eventually enables better decision-making for dynamic scheduling and resource allocation strategies to maximize system performance and efficiency. Figure 3 represents the correlation matrix, where the strength and direction of relationships between different metrics such as CPU_Utilization (%), Memory Consumption (MB), Schedular_Type, and Job_Priority are visually represented using a heat map.

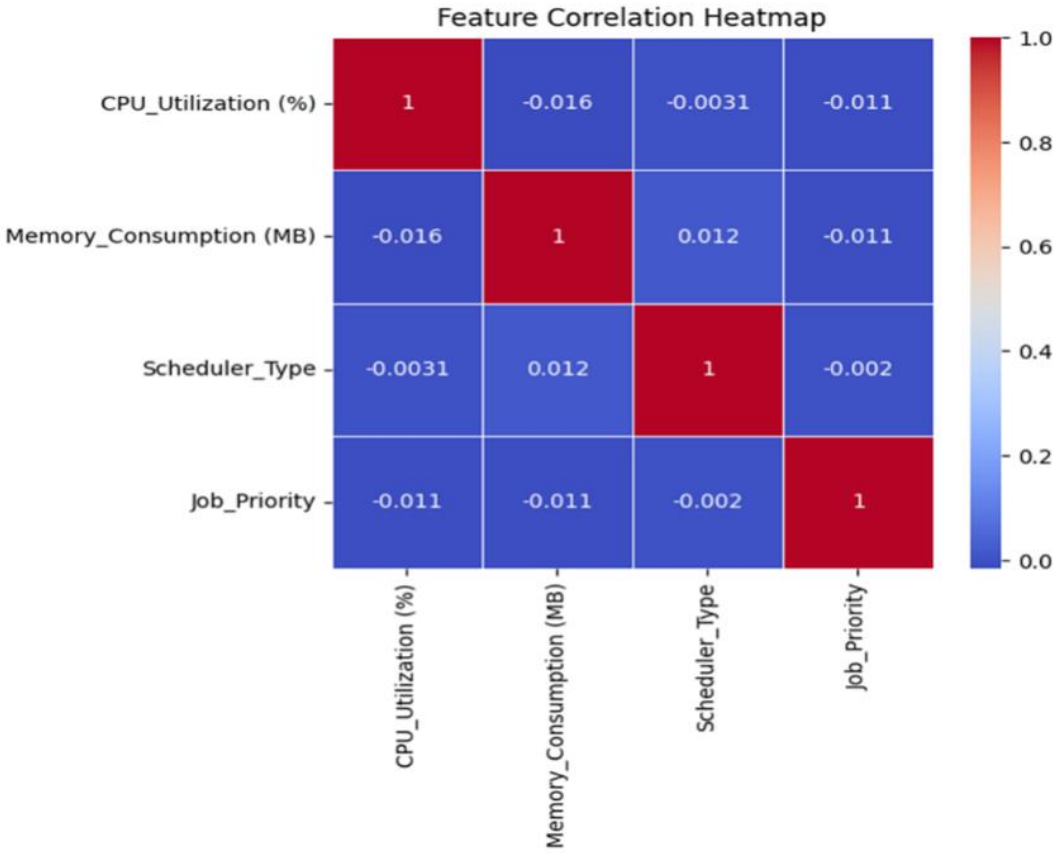

Figure 3. Correlation matrix in cloud-based data analytics.

4.3. CPU usage fluctuations

Efficient CPU utilization is an essential factor in optimizing workload execution and resource management in cloud environments. Monitoring CPU usage patterns over time provides insights into task execution efficiency and system performance. Figure 4 presents a time-series analysis of CPU utilization, illustrating fluctuations in processing demand across various tasks.

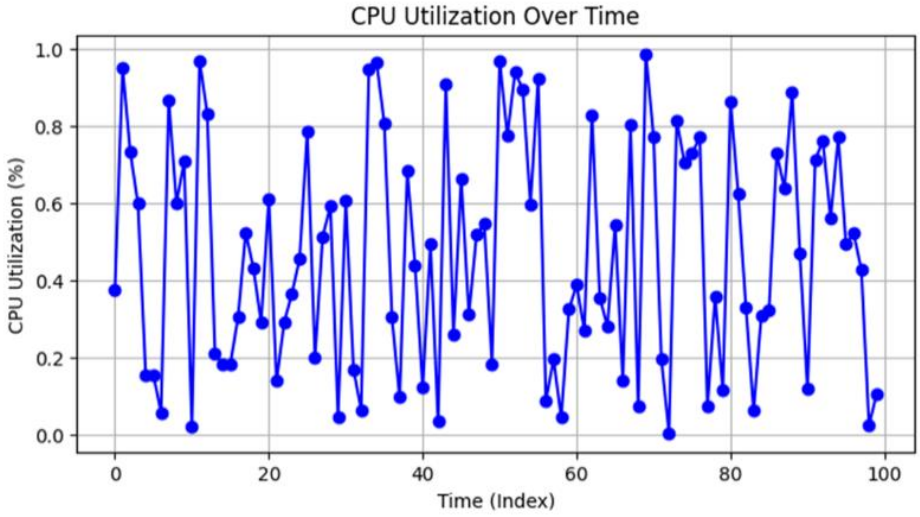

Figure 4. CPU Utilization over time in cloud environments.

4.4. Evaluation Metrics

The performance of ASB-Dynamic-CapsNet was evaluated using key evaluation metrics. Training accuracy measures learning efficiency across epochs. CPU utilization evaluates workload distribution, ensuring optimal processing. RAM utilization analyzes memory efficiency for large-scale task management. These metrics collectively exhibit the model's scalability and resource optimization in cloud computing.

4.4.1. Training Accuracy

To assess the efficiency of the proposed model in optimizing workload distribution in cloud computing environments, the training accuracy was compared with existing methods over multiple epochs. Table 1 presents the performance trends of ASB-Dynamic-CapsNet over traditional models, and Figure 5 depicts the graphical illustration of the performance.

Table 1. Training Accuracy comparison over various epochs.

Epochs	RL [16]	RNN-LSTM [16]	DQN [16]	DRL-LSTM [16]	ASB-Dynamic-CapsNet [Proposed]
10	92.0	96.0	96.0	96.0	97.3
20	89.0	94.8	95.0	96.0	96.8
30	88.0	94.0	94.5	96.0	96.9
40	87.5	93.0	94.0	96.0	97.0
50	87.0	92.0	93.5	96.0	96.7
60	86.0	90.5	93.2	95.8	96.2
70	85.2	89.5	92.8	95.7	96.5
80	84.5	89.0	92.5	95.5	96.3
90	83.0	88.5	92.2	95.3	96.0
100	82.0	88.0	92.0	95.2	95.9

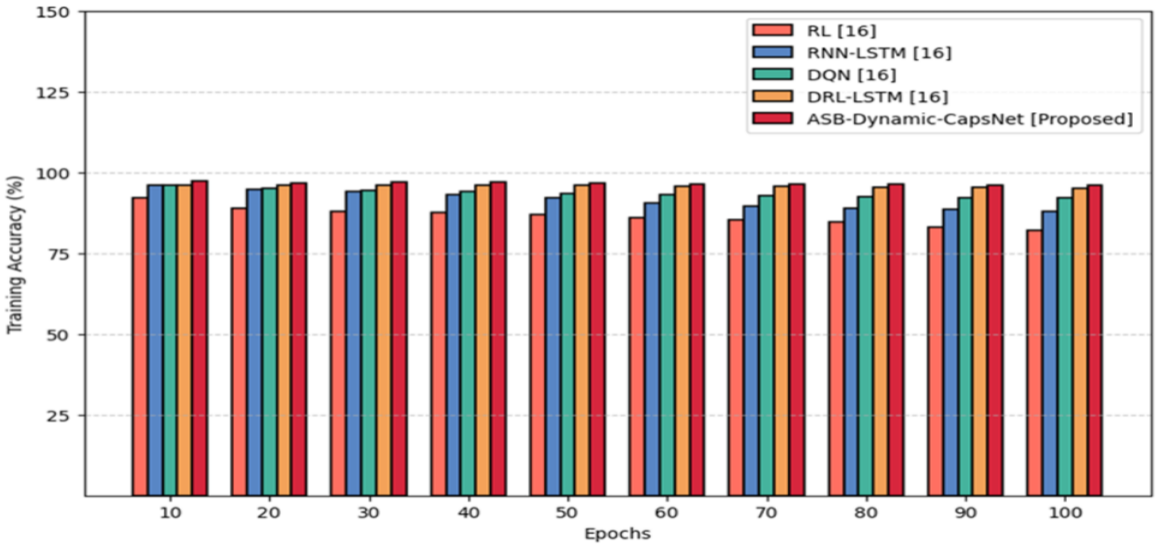

Figure 5. Graphical representation of training accuracy.

Figure 5 illustrates the training accuracy over epochs for different models. The ASB-Dynamic-CapsNet consistently achieves greater accuracy, reaching around 95.9% at 100 epochs.

4.4.2. CPU Utilization Analysis

Efficient CPU utilization is a critical factor in optimizing workload distribution in cloud computing environments. To evaluate the performance of the proposed approach, its CPU utilization was compared to traditional models across various numbers of tasks. Table 2 represents the comparison of CPU utilization, demonstrating the efficiency in handling increasing workloads.

Table 2. Comparative Analysis of CPU utilization across different numbers of tasks.

Number of Tasks (×10⁴)	RL [16]	RNN-LSTM [16]	DQN [16]	DRL-LSTM [16]	ASB-Dynamic-CapsNet [Proposed]
1	11	10	11	10	12
2	19	18	19	17	21
3	26	24	25	24	27
4	33	31	32	31	33
5	40	38	39	37	39
6	47	45	46	44	48
7	55	53	54	52	55
8	62	60	61	58	63
9	72	69	70	67	72
10	87	83	84	81	86

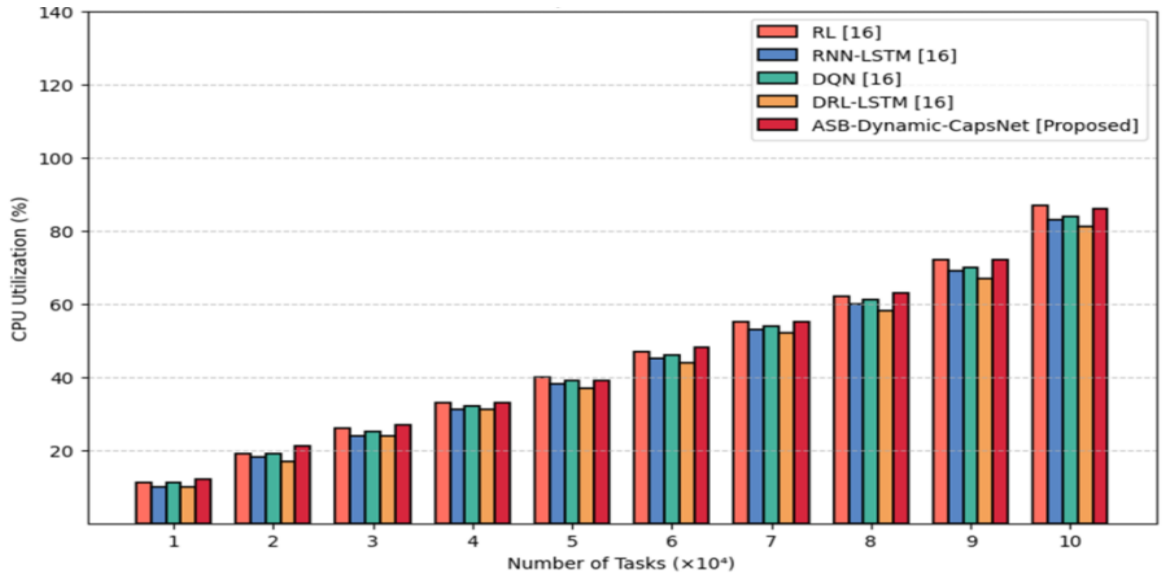

Figure 6. Graphical illustration of CPU utilization

Figure 6 compares CPU utilization across various scheduling models as the number of tasks increases. The proposed model demonstrates enhanced efficiency in workload distribution compared to traditional models. As the workload scales up, CPU utilization rises diagonally in all models, with the present approach achieving the most efficient utilization (86%) compared to DRL-LSTM and DQN. RL and RNN-LSTM demonstrate lower CPU efficiency, respectively.

4.4.3. RAM utilization

Memory efficiency plays an essential role in optimizing workload distribution in cloud computing environments. To evaluate the performance of ASB-Dynamic CapsNet, its RAM usage was compared against existing models across different task volumes. Table 3 presents the RAM usage trends, highlighting the efficiency of the proposed model in managing memory resources.

Table 3. RAM Usage Comparison: ASB-Dynamic-CapsNet vs. Existing Methods

Number of Tasks (×10⁴)	RL [16]	RNN-LSTM [16]	DQN [16]	DRL-LSTM [16]	ASB-Dynamic-CapsNet [Proposed]
1	10	9	10	9	11
2	17	16	17	15	16
3	24	23	25	22	26
4	30	29	31	28	32
5	38	37	39	35	39
6	45	44	46	42	47
7	53	50	54	48	55
8	60	56	61	54	62
9	68	63	66	59	64
10	72	67	70	63	68

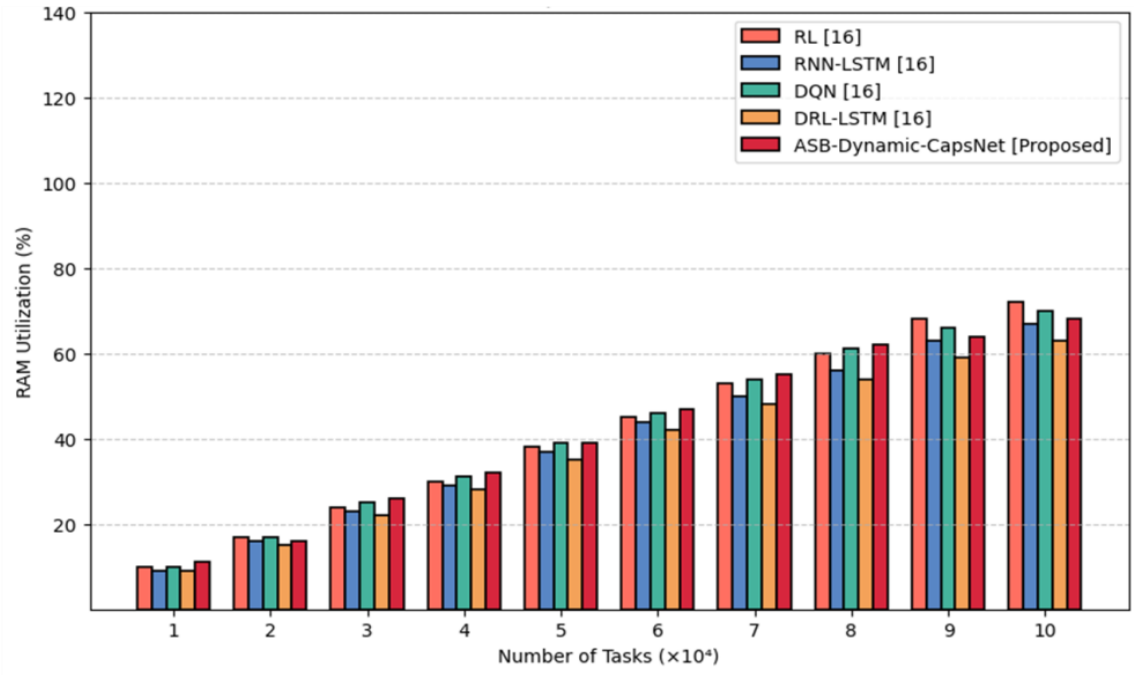

Figure 7. Graphical illustration of RAM utilization.

Figure 7 illustrates the utilization of RAM for various numbers of tasks. The proposed ASB-Dynamic-CapsNet attains the most proficient RAM usage (68%), outperforming DRL-LSTM RL, RNN-LSTM, and DQN in terms of elevated memory.

4.5. Discussion

Optimizing large-scale data analytics in cloud computing requires efficient workload distribution and adaptive scheduling strategies. The implemented ASB-Dynamic-CapsNet model improves scheduling performance by combining DL with metaheuristic optimization, outperforming existing models [16], such as RL, RNN-LSTM, DQN, and DRL-LSTM, in adaptability and resource efficiency. Unlike these approaches, which can struggle with dynamic workload variations and higher computational overhead, ASB-Dynamic-CapsNet ensures optimal resource utilization while maintaining task execution efficiency. The findings highlight its potential for improving cloud-based analytics and minimizing system bottlenecks. However, challenges such as computational complexity and scalability remain. Future research should focus on reducing processing overhead, integrating federated learning for decentralized cloud environments, and strengthening security to enhance real-time applicability in large-scale cloud computing.

5. CONCLUSIONS

The research aims to develop an efficient workload distribution and scheduling strategy to optimize large-scale data analytics in cloud environments. By leveraging DL techniques, the proposed ASB-Dynamic-CapsNet model capably automates workload distribution, reducing execution time and resource utilization. Experimental results exhibit enhanced CPU utilization (86%), RAM consumption (68%), and training accuracy (95.9%), outperforming traditional scheduling methods. The findings highlight the efficiency of DL-driven scheduling in increasing scalability, optimizing resource management, and enhancing the overall performance of cloud-based data analytics. However, the framework has certain limitations, including increased computational complexity and potential scalability challenges in real-time environments. Additionally, workload variations might impact performance, involving further adaptive mechanisms. Future research can focus on optimizing the model for real-time analytics, reducing computational overhead, incorporating associated learning for decentralized cloud environments, and increasing security and privacy in workload scheduling.

DATA AVAILABILITY STATEMENT

The original data presented in the study are openly available in Kaggle at https://www.kaggle.com/datasets/zoya77/cloud-workload-dataset-for-scheduling-analysis

CONFLICTS OF INTEREST

The authors declare that they have no conflicts of interest to this work.

REFERENCES

[1] A. H. A. AL-Jumaili, R. C. Muniyandi, M. K. Hasan, J. K. S. Paw, and M. J. Singh, "Big Data Analytics Using Cloud Computing Based Frameworks for Power Management Systems: Status, Constraints, and Future Recommendations," *Sensors*, vol. 23, no. 6, p. 2952, Mar. 2023, doi: 10.3390/s23062952.

[2] Y. Kumar, J. Marchena, A. H. Awlla, J. J. Li, and H. B. Abdalla, "The AI-Powered Evolution of Big Data," *Appl Sci*, vol. 14, no. 22, p. 10176, Nov. 2024, doi: 10.3390/app142210176.

[3] J. Govea, E. Ocampo Edye, S. Revelo-Tapia, and W. Villegas-Ch, "Optimization and Scalability of Educational Platforms: Integration of Artificial Intelligence and Cloud Computing," *Computers*, vol. 12, no. 11, p. 223, Nov. 2023, doi: 10.3390/computers12110223.

[4] V. K. Prasad, D. Dansana, M. D. Bhavsar, B. Acharya, V. C. Gerogiannis, and A. Kanavos, "Efficient Resource Utilization in IoT and Cloud Computing," *Information*, vol. 14, no. 11, p. 619, Nov. 2023, doi: 10.3390/info14110619.

[5] K. Bajaj, B. Sharma, and R. Singh, "Implementation analysis of IoT-based offloading frameworks on cloud/edge computing for sensor generated big data," *Complex Intell Syst*, vol. 8, no. 5, pp. 3641–3658, Oct. 2022, doi: 10.1007/s40747-021-00434-6.

[6] S. M. Seyyedsalehi and M. Khansari, "Virtual Machine Placement Optimization for Big Data Applications in Cloud Computing," *IEEE Access*, vol. 10, pp. 96112–96127, 2022, doi: 10.1109/ACCESS.2022.3203057.

[7] M. Q. Bashabsheh, L. Abualigah, and M. Alshinwan, "Big Data Analysis Using Hybrid Meta-Heuristic Optimization Algorithm and MapReduce Framework," 2022, pp. 181–223. doi: 10.1007/978-3-030-99079-4_8.

[8] S. T. V. Sresth, S. P. Nagavalli, "Optimizing data pipelines in advanced cloud computing: Innovative approaches to large-scale data processing, analytics, and real-time optimization," *Int J Res Anal Rev*, vol. 10, no. 2023, pp. 478–496.

[9] Ü. Demirbaga, G. S. Aujla, A. Jindal, and O. Kalyon, "Cloud Computing for Big Data Analytics," in *Big Data Analytics*, Cham: Springer Nature Switzerland, 2024, pp. 43–77. doi: 10.1007/978-3-031-55639-5_4.

[10] L. Hu, S. Yang, X. Luo, H. Yuan, K. Sedraoui, and M. Zhou, "A Distributed Framework for Large-scale Protein-protein Interaction Data Analysis and Prediction Using MapReduce," *IEEE/CAA J Autom Sin*, vol. 9, no. 1, pp. 160–172, Jan. 2022, doi: 10.1109/JAS.2021.1004198.

[11] M. Adhikari and H. Gianey, "Energy efficient offloading strategy in fog-cloud environment for IoT applications," *Internet of Things*, vol. 6, p. 100053, Jun. 2019, doi: 10.1016/j.iot.2019.100053.

[12] F. M. Awaysheh, M. N. Aladwan, M. Alazab, S. Alawadi, J. C. Cabaleiro, and T. F. Pena, "Security by Design for Big Data Frameworks Over Cloud Computing," *IEEE Trans Eng Manag*, vol. 69, no. 6, pp. 3676–3693, Dec. 2022, doi: 10.1109/TEM.2020.3045661.

[13] X. Wang *et al.*, "Reproducible and Portable Big Data Analytics in the Cloud," *IEEE Trans Cloud Comput*, vol. 11, no. 3, pp. 2966–2982, Jul. 2023, doi: 10.1109/TCC.2023.3245081.

[14] A. K. Alkhalifa *et al.*, "Hybrid dung beetle optimization based dimensionality reduction with deep learning based cybersecurity solution on IoT environment," *Alexandria Eng J*, vol. 111, pp. 148–159, Jan. 2025, doi: 10.1016/j.aej.2024.10.053.

[15] M. Costa, D. Costa, T. Gomes, and S. Pinto, "Shifting Capsule Networks from the Cloud to the Deep Edge," *ACM Trans Intell Syst Technol*, vol. 13, no. 6, pp. 1–25, Dec. 2022, doi: 10.1145/3544562.

[16] G. Rjoub, J. Bentahar, O. Abdel Wahab, and A. Saleh Bataineh, "Deep and reinforcement learning for automated task scheduling in large-scale cloud computing systems," *Concurr Comput Pract Exp*, vol. 33, no. 23, Dec. 2021, doi: 10.1002/cpe.5919.

BIOGRAPHIES OF AUTHORS

Aditya Gupta is working as a Research associate in computer science and engineering at Lincoln University College, Malaysia, and as a Senior Software Developer in a reputed organization in Lucknow, UP, India. He can be contacted at email: gupta.aditya56@gmail.com

Sai Kiran Oruganti is a professor specializing in wireless power transfer and IoT security, currently affiliated with Jiangxi University of Science and Technology in China and Lincoln University College in Malaysia. His career includes research at the Ulsan National Institute of Science and Technology, developing wireless systems for Hyundai and Samsung, a faculty position at the Indian Institute of Technology, and a recognized history of innovation, including pioneering work on Zenneck Wave WPT and over 16 granted patents. He can be contacted at email: saisharma@lincoln.edu.my