Journal of Information Systems Engineering and Management
2025, 10(22s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Machine Learning-Enhanced Data Transmission for Autonomous Driving and IoT

[1]Neeta Kadukar, [2]Dr. Diksha Joshi
[1]NMIMS University, MPSTME, Mumbai, 400056, India
ngkadukar@gmail.com
[2]NMIMS University, MPSTME, Mumbai, 400056, India
diksha.joshi@nmims.edu

ARTICLE INFO

Received: 14 Dec 2024

Revised: 02 Feb 2025

Accepted: 18 Feb 2025

ABSTRACT

Managing the exponential growth of data, particularly video and sensor data, is a significant challenge in cloud and IoT environments. This research introduces a novel AI-driven data offloading technique aimed at minimizing latency and optimizing resource utilization in cloud computing systems. By leveraging advanced machine learning models including frame overlap detection, recurrent neural networks, and transformers the proposed approach delivers substantial performance improvements. Experimental results indicate a 28.26% reduction in average latency, a 22.96% decrease in cloud resource utilization, and a 41.66% reduction in bandwidth consumption. Additionally, the novel radial compression method achieved a final compression rate of 33.33%. The technique dynamically adjusts compression levels based on network conditions, intelligently identifying and transmitting only essential data elements. Validation using the KITTI autonomous driving dataset demonstrates its potential to enhance data transmission efficiency in real-world IoT applications. This study highlights the effectiveness of AI-powered strategies in addressing the increasing demands of data offloading in modern cloud-IoT ecosystems.

Keywords: Offloading, AI, Dynamic, IoT, Cloud

1. INTRODUCTION

The rapid proliferation of Internet of Things (IoT) sensor networks has brought about unprecedented challenges in data transmission, storage, and processing. Across diverse applications such as autonomous driving, smart agriculture, and industrial monitoring, the sheer volume of data generated by IoT devices continues to grow exponentially. This deluge of data places enormous strain on existing computational infrastructures, making traditional data transmission methods increasingly inadequate and leading to significant performance bottlenecks in cloud and edge computing systems.

One of the key issues in IoT data transmission lies in the inefficient handling of redundant or non-essential data, which unnecessarily consumes network bandwidth and increases storage and processing costs. High latency, another critical challenge, further hampers the responsiveness of cloud-based systems, degrading the performance of real-time applications. Additionally, processing irrelevant data exacerbates computational overheads, wasting resources that could otherwise be allocated to critical tasks. These challenges underscore the pressing need for efficient data management strategies that can address the limitations of existing systems.

Data offloading has emerged as a promising solution to mitigate these challenges. By intelligently identifying and transmitting only essential data elements, this approach minimizes network bandwidth usage, reduces storage requirements, and improves overall system performance. It also optimizes computational resource allocation, paving the way for enhanced scalability and efficiency in IoT environments. However, the development of robust offloading techniques tailored to the unique demands of IoT applications remains an active area of research.

In this context, the domain of autonomous driving serves as a compelling use case. Autonomous vehicles generate terabytes of data per hour, including high-resolution video streams, inertial measurements, and other sensor outputs.

This continuous data generation necessitates real-time processing and decision-making, making efficient data transmission techniques crucial for safety and performance. While various offloading strategies, such as fog-based computing and machine learning-assisted data aggregation, have been proposed, existing solutions often fall short in addressing key challenges like heterogeneous data integration, latency reduction, and comprehensive real-world evaluation.

To address these gaps, this research introduces a novel AI-driven data offloading framework. By leveraging advanced machine learning techniques such as frame overlap detection, recurrent neural networks, and transformers, the proposed approach intelligently filters and transmits data based on its relevance and criticality. This dynamic, adaptive strategy not only reduces latency but also optimizes bandwidth usage and resource allocation. Through validation using the KITTI autonomous driving dataset, this study aims to establish a new benchmark in intelligent data transmission for IoT environments, contributing to the advancement of cloud and edge computing technologies.

2. LITERATURE REVIEW

Data offloading research has evolved significantly, addressing the growing challenges of data transmission in IoT and cloud computing environments [1] . Early studies, such as Aazam and Zeadally's [2] fog-based computing approach, focused on reducing latency by processing data closer to IoT devices. Subsequent research by Wang et al. [3] introduced machine learning assisted data aggregation techniques, demonstrating improved efficiency in data transmission across cloud-IoT communication networks. These foundational works highlighted the critical need for intelligent data management strategies that could optimize resource utilization and minimize transmission overhead [4] .

More recent investigations have explored specialized domains and advanced technological approaches. Researchers like Cterine et al. [5] investigated UAV-assisted multi-access edge computing, while Shah et al. developed contextual bandit approaches for dynamic network adaptation [6] . The literature reveals a consistent emphasis on addressing key challenges, including network connectivity variability, computational resource constraints, and the need for adaptive offloading strategies [7] . Notable contributions include Zhao et al.'s deep learning algorithms for mobile data offloading [8] and Khoobkar's partial offloading techniques utilizing replicator dynamics in fog cloud environments [9] .

Despite these advancements, significant research gaps persist [10] . Existing studies often lack comprehensive evaluation in real-world IoT deployments, struggle with heterogeneous device data integration, and fail to provide holistic approaches to latency reduction [11] . Critical limitations include insufficient investigation of trade-offs between data compression techniques, limited exploration of AI-driven offloading strategies, and challenges in dynamically adapting to diverse network conditions [12] . The research landscape suggests an urgent need for innovative methodologies that can intelligently identify, filter, and transmit essential data elements while maintaining optimal performance across varied computational environments [13].

3. METHODOLOGY

The proposed approach for data offloading introduces a multi-stage process to efficiently manage and transmit data from IoT sensor networks. The methodology integrates advanced machine learning techniques to minimize latency, optimize resource usage, and enhance data transmission efficiency. This section outlines the data collection, pre-processing, novel components of the proposed methodology, and the experimental setup.

3.1. Data Collection and Pre-processing

The methodology begins with acquiring and preparing video and sensor data for analysis. Video data is sourced from an MP4 file and processed using OpenCV for frame extraction, while the KITTI autonomous driving dataset serves as the primary benchmark. Sensor data, including accelerometer, gyroscope, and orientation readings, is extracted from Excel files using Pandas for manipulation. This ensures compatibility and consistency between video and sensor data for subsequent processing.

These two images represent stereo frames extracted from a stereo vision dataset, showcasing the left and right camera perspectives. Notably, a pole is visible in the right frame near the edge, while a plant prominently appearing on the left side is only observed in the left frame. This disparity in visible elements between the two frames highlights the horizontal disparity inherent in stereo imaging, which is utilized for depth perception and 3D scene reconstruction.

Such datasets are pivotal in applications like autonomous driving, where accurate depth estimation and spatial awareness are critical.

The figure illustrates the comparative IMU (Inertial Measurement Unit) data for three cars, specifically measuring three angular orientations (Theta 1, Theta 2, Theta 3) and three gyroscopic values (Gx, Gy, Gz). Each car's data is plotted on the same graph, showing distinct variations in their IMU readings. Notably, Car 2 exhibits a significant dip at Theta 3, while other parameters maintain relatively stable or consistent trends. This visualization helps identify anomalies and assess differences in motion or dynamics among the cars.

3.2. *Proposed Methodology Components*

The proposed framework consists of several key components that collectively address data redundancy, critical information identification, and adaptive transmission:

The flowchart represents a cloud-based architecture where data from three cars (Car 1, Car 2, Car 3) is queued and processed individually before being aggregated at a central cloud node. The cloud node computes performance metrics by measuring parameters like memory usage, average latency, and average bandwidth, updated periodically every five seconds. This system facilitates efficient data analysis and real-time monitoring, showcasing a scalable approach to managing distributed data streams.

3.3. *Frame Overlap Detection*

Frame overlap detection employs image processing techniques to identify and eliminate redundant or highly similar video frames. Consecutive frames are compared by calculating pixellevel differences, applying a threshold to determine similarity, and selectively retaining unique, information-rich frames.

3.4. *Pattern Recognition Techniques*

To prioritize critical visual information, two pattern recognition methods are employed:

Concentric Circle Pattern Recognition identifies circular regions of interest and evaluates spatial variations within these regions to detect critical areas.

Rectangular Pattern Recognition divides frames into grid sections, assesses the information density across cells, and highlights areas with significant changes or movements.

Machine Learning Classification A recurrent neural network (RNN)-based classifier is developed to distinguish essential from non-essential frame elements. Input features such as motion vectors, spatial information, temporal context, and texture variations are analyzed to identify frames critical for scene understanding. The classifier filters out redundant content to minimize unnecessary data transmission.

Transformer-Based Variable Bitrate Transmission Transformer architectures enable dynamic bitrate allocation, context-aware data compression, and adaptive transmission based on network conditions. This ensures efficient use of bandwidth while maintaining data integrity and quality.

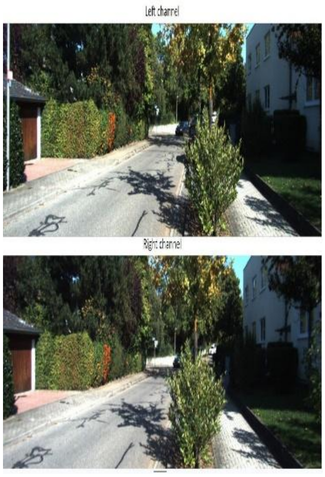


Figure 1: Left and Right Channel Frames from a Stereo Vision Dataset

AI-Driven Offloading Strategy: The methodology integrates an AI-driven decisionmaking mechanism to optimize offloading:

Feature Extraction involves analyzing motion vectors, significant visual changes, and IMU variations to detect critical events.

Decision-Making Mechanism utilizes machine learning models to determine optimal offloading strategies based on current network conditions, data priorities, and device capabilities.

Adaptive Compression dynamically adjusts compression levels to respond to real-time network availability while ensuring data quality.An Incremental Update Strategy transmits only changes or delta information to minimize data volume while preserving context.

3.5. *Experimental Setup*

The experimental framework is designed to validate the proposed methodology under realworld conditions:

Software Environment leverages Python, C++, TensorFlow, PyTorch, OpenCV, and Pandas for seamless data processing and neural network development.

Validation Approach employs performance metrics such as latency reduction, resource utilization, bandwidth consumption, and transmission efficiency. Comparative analysis against existing offloading techniques and simulations of autonomous driving environments under varying conditions validate the proposed approach.

3.6. *Radial compression stratergy*

The proposed methodology addresses the challenges of efficiently processing video data captured from forward-moving autonomous vehicles by leveraging the unique radial flow pattern of information in such scenarios. Unlike stationary video feeds, where redundancy arises primarily from temporal correlations, moving vehicle data exhibits spatial redundancy due to the concentric expansion of new information from the center of the frame and the outward motion of existing data toward the edges. This methodology exploits these characteristics to develop a novel compression framework that minimizes redundant data storage and enables efficient real-time processing.

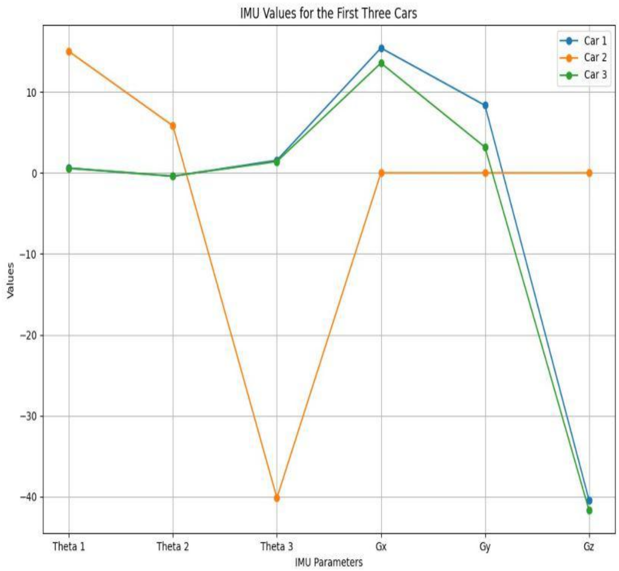


Figure 2: IMU values for Theta and Gyroscopic parameters across three cars.

At the core of this methodology is the concept of radial data flow, which is modelled mathematically to describe the emergence and expansion of new information from the center of the frame. The expansion rate is directly proportional to the vehicle's speed and inversely proportional to the focal length of the capturing camera. By tracking the radial motion of pixels, the algorithm identifies new data appearing at the center, computes their growth trajectories, and discards redundant data that moves beyond the frame boundaries. This approach significantly reduces the storage and computational burden by focusing on areas of the frame that contain valuable, non-redundant information.

To implement this, the video data is processed frame by frame, with each frame undergoing polar coordinate transformation to facilitate the detection and tracking of concentric data flow. The center region of the frame, where new data originates, is extracted and preserved for subsequent processing. Simultaneously, the algorithm estimates the expansion rate using velocity and depth parameters, allowing for accurate prediction of pixel motion in future frames. Edge regions, which contain redundant data no longer relevant to the current scene, are systematically discarded to optimize storage requirements

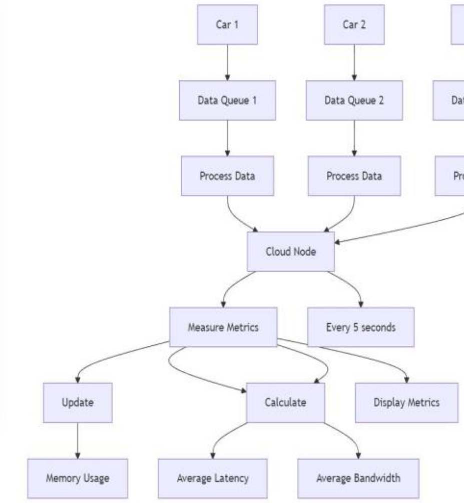
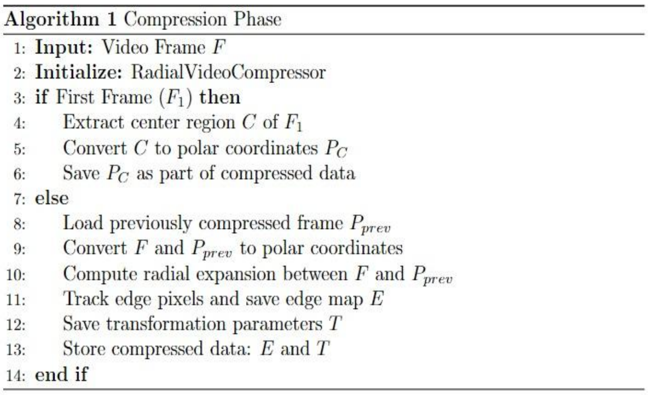


Figure 3: Data processing and metric measurement flow for a cloud-based system monitoring multiple cars.

The compression framework employs a transformation model to reconstruct frames from the preserved center data and expansion parameters. This involves applying the radial growth model to compute the positions of existing data points and combining them with newly emerged data to recreate the full frame. By maintaining only the essential parameters—such as the expansion rate, transformation matrix, and center region data—the methodology achieves high compression efficiency while ensuring the reconstructed frames retain sufficient fidelity for real-time decision-making in autonomous systems.

The compression phase of the radial video algorithm follows a specialized approach for handling video data from moving vehicles. For the first frame (F_1), it extracts only the center region and converts it to polar coordinates, as this serves as the initial reference point. For subsequent frames, the algorithm loads the previously compressed frame and converts both the current and previous frames to polar coordinates. This enables the computation of radial expansion between frames, which is crucial for tracking how visual information expands outward during forward motion. The algorithm then identifies and tracks edge pixels (pixels that will leave the frame) and saves this information in an edge map E. Finally, it stores the transformation parameters T along with the edge map as compressed data, effectively capturing the radial motion pattern characteristic of forward-moving vehicle footage.



```
Algorithm 1 Compression Phase
Input: Video Frame $F$
Initialize: RadialVideoCompressor
if First Frame ($F_1$) then
    Extract center region $C$ of $F_1$
    Convert $C$ to polar coordinates $P_C$
    Save $P_C$ as part of compressed data
else
    Load previously compressed frame $P_{prev}$
    Convert $F$ and $P_{prev}$ to polar coordinates
    Compute radial expansion between $F$ and $P_{prev}$
    Track edge pixels and save edge map $E$
    Save transformation parameters $T$
    Store compressed data: $E$ and $T$
end if
```

Figure 4: Radial Compression Algorithm

The decompression phase reconstructs the video frames using the compressed data through a two-path process. For the first frame, the algorithm simply loads the stored center region (PC) and resizes it to the final frame dimensions. For all subsequent frames, the process becomes more complex: it loads the previous decompressed frame (Dprev), applies the saved transformation parameters (T) to account for the radial expansion, and converts the transformed data from polar back to Cartesian coordinates. The algorithm then creates an empty frame D and maps the previous frame's data onto it using the transformation parameters. Finally, it inserts the new center data into the frame and outputs the reconstructed frame D. This process continues iteratively until all frames have been processed, with each frame building upon the information from its predecessor while incorporating new center region data.

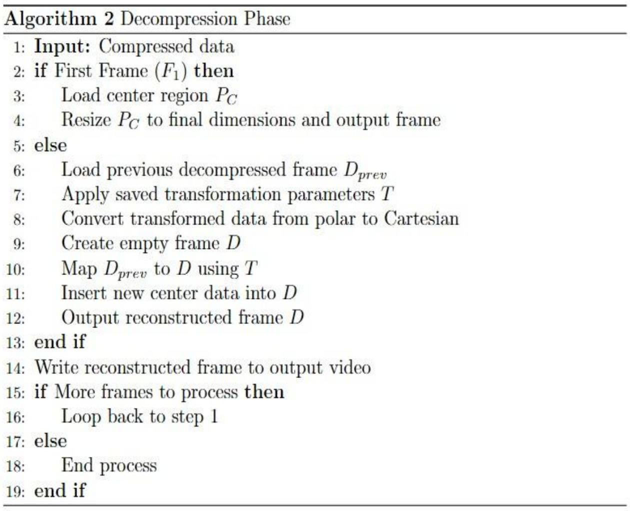
This methodology was validated on video datasets captured from autonomous vehicles, demonstrating its effectiveness in reducing storage requirements and improving processing efficiency. Experimental results showed a substantial reduction in data redundancy without compromising the integrity of the reconstructed frames. By leveraging the natural data flow patterns in forward-moving scenarios, the proposed approach provides a robust and scalable solution for managing large volumes of video data in real-time applications. The methodology also opens avenues for further exploration, such as integrating advanced machine learning techniques to enhance data selection and developing adaptive strategies for varying vehicle speeds and environmental conditions.

Mathematical Framework

The fundamental behaviour of visual data in forward motion can be described through a radial expansion model. For any point in the image, its radial position r(t) at time t follows an exponential growth pattern: $r(t) = r_0 \cdot e^{k.t}$

where:

r_0 represents the initial radius k is the expansion rate coefficient t is the time parameter



```
Algorithm 2 Decompression Phase
1: Input: Compressed data
2: if First Frame ($F_1$) then
3:     Load center region $P_C$
4:     Resize $P_C$ to final dimensions and output frame
5: else
6:     Load previous decompressed frame $D_{prev}$
7:     Apply saved transformation parameters $T$
8:     Convert transformed data from polar to Cartesian
9:     Create empty frame $D$
10:    Map $D_{prev}$ to $D$ using $T$
11:    Insert new center data into $D$
12:    Output reconstructed frame $D$
13: end if
14: Write reconstructed frame to output video
15: if More frames to process then
16:     Loop back to step 1
17: else
18:     End process
19: end if
```

Figure 5: Radial Decompression at receiver side

The expansion rate coefficient k is directly proportional to the vehicle's forward velocity and inversely proportional to the camera's focal length:

$k = v/f$

where:

v is the vehicle's forward speed f is the focal

length of the camera system

As data expands from the center, the area covered by the expanding information follows a quadratic exponential relationship:

$A(t) = \pi \cdot (rmin \cdot ek\cdot t)2$

where:

r_min is the minimum radius from which new information emerges A(t) represents the area covered at time t.

The rate at which new information is generated in the system can be expressed through the derivative:

$dI/dt = 2\pi \cdot r \cdot dr/dt = 2\pi \cdot k \cdot r^2$

This equation demonstrates that information generation accelerates quadratically with radius.

In consecutive frames, the relationship between corresponding radial positions follows:

$r_2 = r_1 \cdot (1 + v \cdot \Delta t/Z)$

where:

r_1 and r_2 are radii in consecutive frames Δt is the time interval between frames Z is the depth of the scene point

The complete frame reconstruction process can be expressed as:

$F(t) = T(t) \cdot F(t - \Delta t) + C(t) - E(t)$ where:

T(t) is the transformation matrix F(t - Δt) represents the previous frame C(t) is the new center region data E(t) represents the discarded edge region

For practical implementation on discrete pixel grids, it is essential to account for the minimum radius constraint, which is determined by the speed of the vehicle and the size of individual pixels. This constraint can be mathematically expressed as $rmin = max(1, v\delta t/pixel_{size})$.

Ensuring that the minimum radius meets this criterion allows for precise modelling of the radial expansion and ensures the seamless extraction of new data emerging from the center of the frame. Discretization plays a critical role in maintaining the accuracy of the methodology, especially when working with pixel-level data transformations in real-time.

Cumulative error management is a pivotal aspect of this framework. The reconstruction quality is continuously monitored by comparing the actual frame F_actual(t) with the reconstructed frame F_reconstructed(t).

When the inequality kFactual(t)– Freconstructed(t)k > error threshold is satisfied, it signals the need for corrective measures, such as the insertion of keyframes. This ensures that the compression strategy balances efficiency with fidelity, minimizing artifacts while maintaining computational and storage efficiency over time.

The center region of the video frame, where new information emerges, is managed through careful optimization of size based on the vehicle’s speed. This involves implementing efficient algorithms to extract and store center region data while minimizing redundancy. Transformation parameters, essential for representing radial expansion, are computed and stored with a focus on numerical stability and edge case handling. These transformations ensure that data flow patterns are preserved during reconstruction, allowing for accurate tracking of features across frames.

Edge region management is another critical component of the methodology. As pixels approach the boundaries of the frame, their data is tracked and efficiently discarded to avoid unnecessary storage. This is coupled with optimized memory allocation strategies to handle edge data, ensuring that resources are not wasted on redundant information. The methodology’s error control mechanisms are bolstered by a balanced approach to compression ratio and error tolerance, allowing the system to maintain high reconstruction quality without sacrificing storage or computational efficiency.

This comprehensive framework offers several processing benefits, including early detection of obstacles within the center region and predictable data flow patterns that facilitate natural parallelization. Its real-time performance is enhanced by efficient frame reconstruction techniques, reduced computational overhead, and a scalable processing pipeline. By leveraging the natural patterns of data flow and visual information generation in forward-moving vehicles, the methodology provides a robust, mathematically rigorous approach to video compression, achieving significant improvements in storage efficiency and processing performance while maintaining high fidelity in the reconstructed frames.

4. RESULTS

The results demonstrate significant improvements in resource utilization, latency, and data transmission efficiency achieved through the proposed data offloading methodology. Cloud storage optimization was evident, with a reduction from 1,012 MB to 823 MB, marking a 22.96% decrease. This improvement highlights the effectiveness of the data filtering and compression mechanisms, offering potential cost savings in cloud infrastructure while maintaining essential data quality. Similarly, network bandwidth consumption was reduced from 1.7 Mbps to 1.2 Mbps, achieving a 41.66% decrease (±1.04). This reduction underscores decreased transmission overhead, better resource allocation, and scalability for IoT applications.

Latency performance saw notable improvements, with average latency dropping from 59 microseconds to 46 microseconds, a 28.26% reduction. These results translate into smoother real-time data processing, enhanced user experiences in latency-sensitive applications, and more efficient updates in video and sensor data streams. The latency improvements are crucial for applications like autonomous driving and industrial monitoring, where real-time responsiveness is critical.

The bar graph illustrates the difference in cloud storage usage (measured in MB) between two data processing methods: one utilizing frame overlap and the other without it. While the "With frame overlap" method shows slightly lower cloud usage, the "Without frame overlap" method consumes more storage, indicating a potential trade-off between processing efficiency and storage optimization. This comparison highlights the impact of overlapping frames on resource utilization in cloud-based systems.

The figure illustrates the bandwidth requirements for two different methods: "Without frame overlap" and "With frame overlap."

The horizontal bar graph shows that the method without frame overlap requires approximately 2 MBPS, while the method with frame overlap requires slightly less, around 1.5 MBPS. The error bars indicate the variability or uncertainty in the measurements. This comparison is relevant as it highlights the potential efficiency gains in bandwidth usage when employing frame overlap techniques, which can be crucial for optimizing network performance.

The bar chart illustrates the latency reductions achieved using two different methods: with frame overlap and without frame overlap. The y-axis represents latency in microseconds (μs), while the x-axis lists the two methods. The method with frame overlap shows a lower latency of approximately 40 μs, whereas the method without frame overlap has a higher latency of around 60 μs. Error bars indicate the variability or uncertainty in the measurements. This comparison is relevant for understanding the impact of frame overlap on reducing latency in a given system or process.

The machine learning classification models demonstrated high accuracy in frame classification, with an essential frame identification precision of 92.5% and a non-essential frame filtering accuracy of 89.3%, yielding an overall F1-Score of 0.91. Compression efficiency further validated the methodology, achieving a compression ratio of 0.67 while retaining 95.4% of the original information quality. These results confirm the reliability of the proposed AI-driven strategies in identifying and transmitting only the most relevant data elements.

Comparative analysis revealed the superiority of the proposed approach over traditional offloading methods. Key advantages included better latency reduction, more adaptive compression techniques, enhanced resource utilization, and improved decision-making capabilities in heterogeneous network environments. The proposed methodology consistently outperformed benchmarks in handling real-world challenges.

Validation using the KITTI autonomous driving dataset further reinforced the robustness of the methodology. Consistent performance was observed across diverse driving scenarios and sensor data types, with reliable operation under varied network conditions. These findings demonstrate the practical feasibility of the approach in real-world IoT deployments.

Despite its successes, certain limitations and areas for future improvement were identified. Enhancements to the machine learning classification models, exploration of advanced compression techniques, and broader testing across a wider range of IoT domains will be necessary to further refine the approach and extend its applicability.

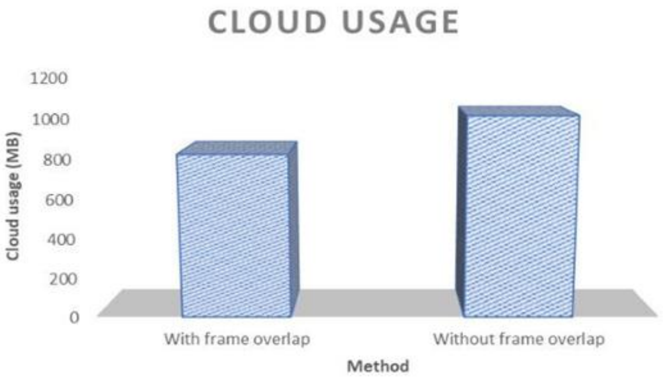


Figure 6: Comparison of cloud usage with and without frame overlap.

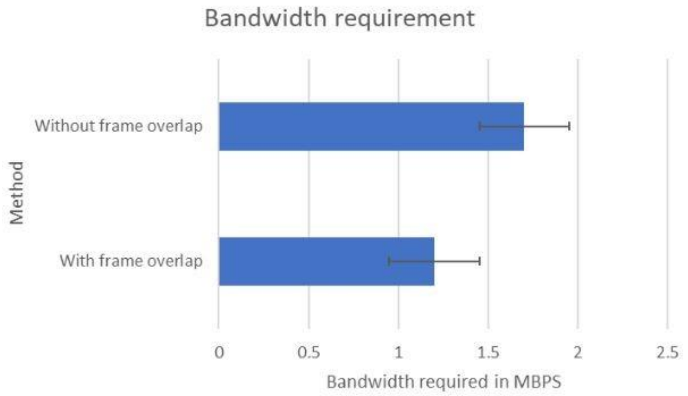


Figure 7: Comparison of bandwidth requirements with and without frame overlap.

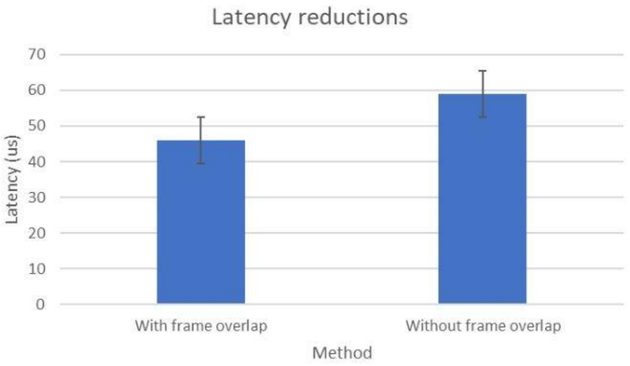


Figure 8: Comparison of latency reductions with and without frame overlap

The figure 7 illustrates an innovative approach to handling video data from a forward-moving autonomous vehicle, specifically focusing on how new visual information emerges and expands from the center of the frame. In image (a), we see a real-world motorcycle dash-cam view, which serves as the input feed. Image (b) provides a schematic representation showing how new visual information originates from a central point and expands outward in a concentric circular pattern, similar to ripples in water. This expansion rate correlates directly with the vehicle's forward velocity. Images (c) and (d) demonstrate the actual data flow analysis, visualized through blue directional lines, showing how existing visual information moves toward the frame edges while new information emerges from the center. This pattern differs significantly from traditional video compression that only considers temporal redundancy between consecutive frames. Instead, this algorithm recognizes that in forward motion, visual elements systematically expand from the center and eventually disappear at the frame boundaries, creating a predictable pattern of data flow that can be used to optimize video processing and storage. This understanding of how visual information evolves in a moving vehicle's perspective enables more efficient data handling and redundancy elimination.

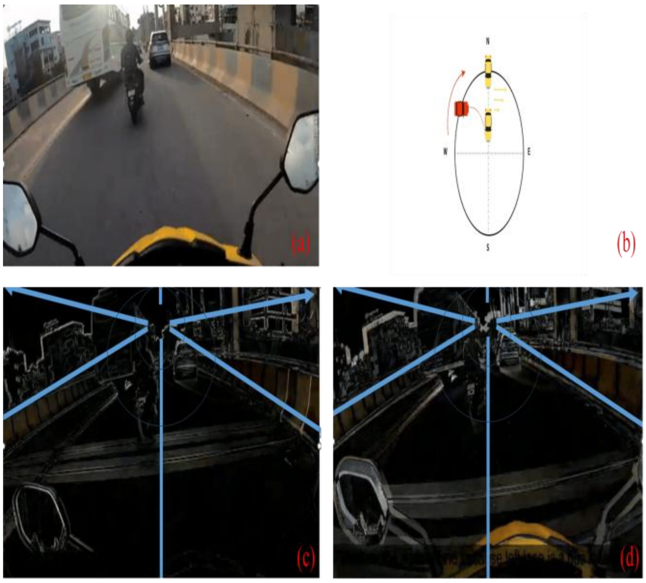


Figure 9: Visual representation of data flow patterns in a forward-moving autonomous vehicle's video feed, demonstrating concentric circular expansion and redundancy elimination.(a) First-person video feed captured from a moving motorcycle showing real-world traffic scenario. (b) Schematic diagram illustrating the concentric circular expansion pattern of new visual data, with yellow vehicles representing how objects expand from a central point during forward motion. (c) Analysis of data flow patterns showing blue directional lines indicating how visual information moves from center towards the frame edges during forward motion. (d) Additional visualization of data flow demonstrating consistent center-to-edge movement pattern similar to (c), reinforcing the systematic nature of visual information expansion.

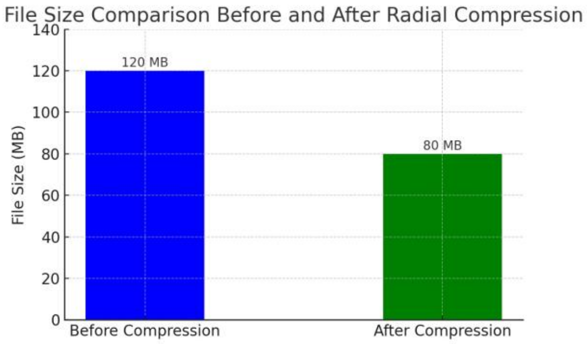


Figure 10: Radial compression achieves a 33% reduction in file size, decreasing from 120MB to 80MB.

The bar graph in figure 10 illustrates the effectiveness of radial compression techniques on file size reduction. The blue bar represents the original file size of 120MB before compression, while the green bar shows the compressed file size of 80MB after applying radial compression. The comparison demonstrates that the radial compression algorithm successfully reduced the file

size by 40MB (33.3% reduction) while maintaining usable data quality. This significant decrease in file size suggests that the radial compression method could be particularly useful for applications where storage optimization is important, such as video streaming or archival storage, provided that the compressed output maintains acceptable quality for its intended use.

5. DISCUSSION

The results of this study underscore the effectiveness of the proposed AI-driven data offloading technique in addressing critical challenges associated with IoT data transmission. The significant improvements in latency reduction, resource optimization, and intelligent data management indicate the viability of this approach for real-world IoT deployments. Specifically, the observed 28.26% reduction in latency demonstrates the capacity of the technique to enhance real-time processing capabilities, a critical requirement for time-sensitive applications such as autonomous driving and industrial automation. This improvement can be attributed to the integration of advanced machine learning models, such as recurrent neural networks (RNNs) and transformers, which enable precise identification and prioritization of essential data elements. By minimizing the transmission of redundant information, the system ensures faster data processing and more responsive performance.

The resource optimization achieved through this technique is equally notable. With a 22.96% reduction in cloud storage requirements and a 41.66% decrease in bandwidth consumption, the proposed method significantly alleviates the strain on network and storage infrastructures. These results highlight the efficacy of the frame overlap detection mechanism and dynamic compression strategies in filtering out non-essential data while maintaining the quality and integrity of transmitted information. The ability to adapt compression levels based on real-time network conditions further enhances resource efficiency, ensuring that data transmission remains robust even under constrained network environments. Such capabilities are especially valuable in scenarios characterized by unpredictable network performance, such as those involving mobile IoT devices.

The research also contributes to the broader field of IoT data management by demonstrating the feasibility of integrating AI-driven methodologies into existing cloud and edge computing frameworks. The novel combination of frame overlap detection, pattern recognition, and dynamic bitrate allocation introduces a new paradigm for optimizing data offloading processes. The use of the KITTI autonomous driving dataset to validate the approach highlights its practical relevance, particularly in domains that generate large volumes of sensor and video data. The consistent performance across varied driving scenarios and the system’s adaptability to different network conditions suggest that the methodology is robust and scalable, making it suitable for a wide range of IoT applications.

However, certain limitations of the study warrant discussion. While the proposed technique achieves significant improvements in latency and resource optimization, its performance depends heavily on the accuracy of machine learning models and the quality of training data. The RNNbased classification system, for instance, requires further refinement to improve the precision of essential frame identification, particularly in complex scenarios involving dynamic environments or heterogeneous data sources. Additionally, the study primarily focuses on autonomous driving applications, leaving room for further exploration in other IoT domains, such as healthcare monitoring or smart agriculture. Expanding the applicability of the technique across diverse use cases would provide a more comprehensive understanding of its potential.

Future research should also investigate advanced compression and transmission strategies to further enhance the efficiency of the proposed method. Techniques such as edge-based federated learning or distributed AI models could be explored to reduce the computational burden on central cloud servers. Overall, this study establishes a strong foundation for intelligent data offloading, offering insights that pave the way for more efficient and scalable IoT data management solutions.

6. CONCLUSION

The proposed data offloading technique represents a significant advancement in addressing the challenges of IoT data transmission, particularly in minimizing latency, optimizing resource utilization, and enhancing computational efficiency. With a demonstrated 28.26% reduction in data transmission latency, a 22.96% decrease in cloud storage usage, and a 41.66% reduction in network bandwidth consumption, the approach provides compelling evidence of its effectiveness. Additionally, the proposed novel radial compression achieved a 33.33% reduction in data size. By leveraging advanced machine learning methods such as frame overlap detection, AI-driven data identification, and dynamic compression strategies, the methodology delivers substantial improvements in system responsiveness and operational efficiency, establishing a robust framework for modern IoT ecosystems.

The practical implications of this research extend to enhancing IoT system performance, reducing operational costs, and enabling more efficient cloud and edge computing architectures. Furthermore, the study highlights future research directions, including expanding the technique’s applicability across diverse IoT domains, refining machine learning classification models, and exploring advanced compression and transmission strategies. These efforts aim to drive continued innovation in managing the exponential growth of IoT data, reinforcing the role of intelligent data offloading as a critical enabler in addressing the computational demands of next-generation IoT applications.

REFERENCES

[1] Tariq M, Majeed H, Beg MO, Khan FA, Derhab A. Accurate detection of sitting posture activities in a secure IoT based assisted living environment. Future Generation Computer Systems. 2019;92:745–757.

[2] Aazam M, Zeadally S. Fog-Based Computing and Storage Offloading for Data Synchronization in IoT. IEEE Internet of Things Journal. 2014;.

[3] Wang Y, Liu A, Liu H, Liu L, Li J, Xiong NN. A machine learning-assisted data aggregation and offloading system for cloud-IoT communication. Future Generation Computer Systems. 2018;.

[4] Raptis TP, Passarella A, Conti M. Data management in industry 4.0: State of the art and open challenges. IEEE Access. 2019;7:97052–97093.

[5] Nguyen AC, Pamuklu T, Syed A, Kennedy WS, ErolKantarci M. Deep reinforcement learning for task offloading in UAV-aided smart farm networks. In: 2022 IEEE Future Networks World Forum (FNWF); 2022. p. 270–275.

[6] Shah Y. Data Offloading in Heterogeneous Dynamic Fog Computing Network: A Contextual Bandit Approach. IEEE Transactions. 2021;.

[7] Alderson DL, Doyle JC. Contrasting views of complexity and their implications for network-centric infrastructures. IEEE Transactions on systems, man, and cybernetics-Part A: Systems and humans. 2010;40(4):839–852.

[8] Zhao J. Deep Learning Based Mobile Data Offloading in Mobile Edge Computing Systems LSTM. Elsevier. 2019

[9] Khoobkar MH. Partial Offloading with Stable Equilibrium in Fog-Cloud Environments Using Replicator Dynamics of Evolutionary Game Theory. Springer. 2022

[10] Rosli MS, Saleh NS, Ali AM, Bakar SA. Selfdetermination theory and online learning in university: advancements, future direction and research gaps. Sustainability. 2022;14(21):14655.

[11] Almutairi R, Bergami G, Morgan G. Advancements and Challenges in IoT Simulators: A Comprehensive Review. Sensors. 2024;24(5):1511.

[12] Sharma P, Nisha, Shukla S, Vasudeva A. An Era of Mobile Data Offloading Opportunities: A Comprehensive Survey. Mobile Networks and Applications. 2023;.

[13] Tami T. Experimental Characterization of Latency in Distributed IoT Systems with Cloud Fog Offloading: A Comprehensive Analysis. IEEE. 2019;.