International Journal of Applied Mathematics

Volume 38 No. 8s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

OPTIMIZING MACHINE LEARNING ALGORITHMS FOR REAL-TIME DATA PROCESSING

Samer Naser Hasan Alqatrani

Azad Islamic University \ Branch of science and Investigations

Specialization / Master of Computer Science

Samernasser188@gmail.com

Abstract

Real-time data processing has become an inseparable part of the computing environment since it helps to serve a huge range of applications, such as financial trading systems, industrial monitoring, driverless cars, and personalized healthcare. These systems have the foundation of a fast decision making according to the ever changing data streams. The necessity of machine learning (ML) systems that are able to handle and respond to real-time data has been on the rise because of unprecedented data volume generation speed assisted by sensors, social media, and Internet of Things (IoT) apparatus. The traditional machine learning process is highly resource consuming and time-consuming to satisfy the real time environment needs. Hence, there is need to optimize the ML algorithms now on a real-time basis. This will involve the efforts of ensuring that the algorithms are made to be more flexible, quick and capable of delivering low-latency time responses without compromising the accuracy. One of the strategies that are significant where the models are continuously updated with the new information as online learning progresses and enables the models to readjust as the information continuously comes in online and the models do not always have to be retrained again. There are simpler models such as pruning and quantization that reduce models to cut on computational resources and improve the processing time. Edge computing reduces the latency and speed of reaction of time-critical applications, by relocating the computation to the data source. Also, processing distributed across several computers or servers enhances the scale and is capable of processing extensive data streams in parallel. This is how through the combination of these techniques, machine learning systems can be better adapted to dynamic and real-time conditions and perform better in critical applications where speed, efficiency, and adaptability are the main prioritie.

Keywords: Real-Time Data Processing, Machine Learning Optimization, Online Learning Algorithms, Edge Computing, Distributed Systems

1 Introduction

In an age characterized by big data and pervasive computing technology, the demand for real-time processing is becoming an essential requirement in various critical areas, including

Received: August 13,2025

Volume 38 No. 8s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

autonomous vehicles, financial trading, smart cities, and healthcare systems. All these applications produce and consume large amounts of data at high rates, necessitating computing systems that are able to process information and respond to conditions in real-time with lower latency. Traditional machine learning (ML) systems, as they largely rely on batch processing and static datasets, are often insufficient to better meet the requirements of these dynamic, rapid, and dynamic settings.

To address this, there is a research agenda that aims to further optimize machine learning algorithms in real-time situations so that models are not only accurate but also fast, adaptive, and resource efficient. This research agenda has promoted different strategies and methodologies, including online learning, streaming algorithms, edge computing, model simplification, and distributed systems.

One of the most common solutions to this challenge is online learning (and related methodologies), in which the ML models are incrementally updated as a stream of data becomes available. Online learning is distinct from prior traditional training paradigms; it does not require the entire dataset in its entirety upfront, supporting ongoing updates, which is critical to systems that require ongoing functionality in a changing environment. Hoi et al. (2018), in their survey “Online Learning: A Comprehensive Survey,” discussed various online learning algorithms, and demonstrated their use in settings where time and accuracy matter. An important issue emphasized by Hoi et al. is that online learners must balance fast convergence with a stable state in environments subject to concept drift - in these environments the distribution of data changes over time.

The use of spiking neural networks (SNNs) is an important new area within real-time ML. These biologically inspired models excel in low-power, high-speed, computing as they imitate the human brain process by only responding to discrete events or spikes. In "Spiking Neural Networks and Online Learning," Lobo et al. (2019) provide a fantastic review of the advantages of SNNs in acquiring sensory data in robotics and neuromorphic hardware systems. They illustrate that SNNs are able to learn via reinforcement learning techniques share observations and learn in real-time under extreme energy efficiency.

Concurrently, there is a movement to lower computational complexity which has led to research in model simplification techniques such as model pruning, quantization, and knowledge distillation to lower model size and inference time while maximizing performance. Simplified models can be of tremendously benefit when deployed to edge devices; small scale systems with limited computation and memory requirements. In a seminal paper, Han et al. (2016) presented structured pruning as a method to trim redundant weights on neural networks with a significant decrease in latency while preventing a corresponding decrease in accuracy. This has provided the foundation for mobile and embedded real-time applications.

The concept of edge computing has emerged as an important enabling technology for real-time ML. In contrast to cloud-based systems, edge computing supports data processing in relatively

Received: August 13,2025

International Journal of Applied Mathematics
Volume 38 No. 8s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

local locations — close to the data generation — which reduces time latency associated with transmitting the data to a faraway server. For example, Shi and colleagues in 2021 examined how federated learning and edge AI platforms can help deliver privacy-preserving real-time analytics from healthcare wearables and industrial IoT. The authors concluded with evidence that edge intelligence is vital for mission-critical systems where low-latency or real-time response time analysis is necessary.

Another technique which has been widely adopted to follow in the footsteps of edge AI, for example, is the distributed data processing frameworks such as Apache Flink, Storm, and Spark Streaming, which enable the high throughput requirements of real-time ML. Under such systems, the processing of data is done parallel in a cluster to enable responsiveness when there are massive input streams. When Carbone et al. (2015) presented the architecture of Apache Flink as the streaming analytics platform with stateful event processing, they have established a basis of real-time AI pipelines.

Although these developments have taken place, numerous unresolved problems remain: how to keep models accurate when there is concept drift, how to train models to operate in low-resource conditions and how to reduce end-to-end latency, and whether to make models fault-tolerant to real-time. All these issues demand a multi expected design that involves algorithm design, systems engineering, and hardware acceleration.

The contribution of this paper is the work of ambient which has been developed in evaluating the most relevant tactics to maximize a machine learning model in real-time. We shall elaborate on the available algorithms and architectures and how they suit various areas of application and come up with a convenient model of selecting and tailoring forms depending on system constraints in application. The work would facilitate the creation of more responsive, adaptive, and intelligent real-time AI systems.

2 Previous Works in Real-Time Machine Learning Optimization

This is machine learning (ML) algorithm optimization to real-time operations on incoming data streams and this has been of great interest in the last few years and is fueled by the increasing need to have systems that can effectively execute operations on real-time data streams. In the section, over twenty influential research studies that have been playing a role in this area are discussed with reference to their methodology, findings, and inferences. Among them, it is possible to refer to the works of Lobo et al. (2019) who discussed the use of the Spiking Neural Networks (SNNs) in the process of online learning. Their input highlights the fact that SNNs are adaptable to non-stationary cases and this makes them to be applicable to real time cases where data distributions may vary with time (arxiv.org).

Li et al. (2018) presented a proposal of a model-free training that utilizes Deep Reinforcement Learning (DRL) to operate the Distributed Stream Data Processing Systems (DSDPSs). DRA demonstrated that its data processing optimization of real-time was effective because its

Received: August 13,2025

Volume 38 No. 8s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

structure decreased the average time per tuples by 33.5 percent in comparison to the traditional ones (arxiv.org). In 2023, Shaowang and Krishnan proposed EdgeServe a system based on the distributed streaming system, which is a decentralized model serving system. They use their system to solve problems such as routing of data, time synchronization and rate control and are also being utilized to perform real time predictions in such applications as human activity recognition and self-driving cars. (arxiv.org)

Strider is an adaptive distributed RDF stream processing engine which is a hybrid developed by Ren and Curé (2017). Strider is very scalable and has got high throughput in real-time processing environments where logical query plans will be optimized based on the state of the data streams. The primary idea of the research article of zkan and ahin (2023) on AI applications in real-time edge processing is that machine learning models are merged with edge computing to enhance the quality of the decision-making process in dynamic settings. The study by Henan (2024) focused on optimization of real-time processing of Convolutional Neural Networks (CNNs) in edge computing system. Pruning and quantizing models enabled them to reduce latency and boost throughput of CNNs to make them more appropriate to real-time applications. Xu et al. (2021) have presented an overview of streaming algorithms to process real-time data. Their survey encompassed different methods which have low memory, and processing, time thus can be used in real-time applications. (research.google)

Diao et al. (2020) researched federated learning in heterogeneous clients and discovered a structure that allows it to be feasible to train local models despite having varying computation and communication abilities. Semii-streaming algorithms of graph problems were proposed by Feigenbaum and Sampath (2005), which permit linear space requirement based on the number of vertices, and logarithmic space requirement based on the number of edges. This relaxation is now able to solve sublinear space insoluble graph problems. The discussion of algorithms with limited storage to select and sort by Munro and Paterson (1978) formed the basis of the streaming algorithms with limited memory (research.google). Streaming algorithms were formalized and popularized by Alon et al. (1996), who studied the space complexity of data streams frequency moment approximation. Jain (2016) offered a descriptive overview of Apache Storm, which is a real-time computation system. The book includes the architecture, the components, and the applications of Storm with emphasis on its application in real-time analytics and machine learning.

Wolfe (2014) described the use of Apache Storm by The Weather Channel to process real time weather information. The article points out the fact that the system was capable of dealing with large amounts of streaming data and delivering real time weather updates.

Carvalho et al. (2021) introduced the concept of edge computing, its latest trends, research opportunities, and future. They highlighted the significance of edge computing in allowing real-time processing of data and using machine learning.

Received: August 13,2025

International Journal of Applied Mathematics
Volume 38 No. 8s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Yuan et al. (2023) proposed FedSTN, a federated learning framework, a graph-based traffic flow predictor in cities. The method enhances the scalability and accuracy of the traffic prediction models in the applications that utilize the concept of edge computing. An integration of a computer vision and edge computing is a smart traffic monitoring system suggested by Liu et al. (2022). The system explains how the real-time traffic information can be operated on the local level to enhance better traffic management and eliminate congestions.

The synthesis of all the studies results in the evolving nature of real-time machine learning optimization that introduces various approaches and expertise that perceived on the state of practice and introduce additional opportunities to the field.

3 Challenges and Optimization Techniques in Real-Time Machine Learning

To be in a position to operate in dynamic environments, the machine learning algorithms have to face a distinct set of challenges each time they handle real-time data. One of the most crucial ones is the latency or the period during which the process of processing the data acquired can be postponed. Such applications as autonomous vehicles and real-time fraud detection systems will be adversely impacted by high latency. The other significant issue is scalability because the volume and speed of streaming data may easily exceed what the traditional infrastructure can assist and the requirement to find mechanisms that may expand in all directions or apply distributed computing systems (en.wikipedia.org). Also, concept drift: the statistical qualities of the input data vary over time, which can deteriorate the accuracy of the model, requiring a regular update to the model or default learning methods (medium.com). Lack of resources is also a major constraint, particularly to the edge devices such as smartphones and IoT sensors. These systems may have very minimal CPU, memory, and power, and thus, featherweight and efficient algorithms are needed to process data on the device (moldstud.com).

A number of major tactics have been formulated in order to resolve these problems. Online learning algorithms are one of the popular alternatives especially in real-time implementations. The online learning methodology progressively updates the parameters when new data is provided, unlike traditional models that are only trained once using fixed data sets. Rapidcanvas.ai and medium.com are two instances of online variants of Stochastic Gradient Descent (SGD) and k-means clustering that can be continuously adapted without having to retrain the entire model. This is complemented by model simplification techniques that tend to decrease the computational complexity of a model. Pruning simplifies the network by removing unnecessary neurons or parameters (arxiv.org). Rapidcanvas.ai writes that quantization accelerates the inference process by decreasing the numerical accuracy of the model weights and, thus, lowers memory usage. Knowledge distillation (rapidcanvas.ai) involves training a smaller model to mimic the behaviour of the larger, more complex model, without compromising accuracy whilst enhancing deployability. These approaches are particularly applicable in contexts of restricted resources. In addition, edge computing has also become a disruptive technology to minimize latency in real-time ML products. Edge computing minimizes the use of centralized servers and network bandwidth by performing data processing where it is most useful, e.g. on a smartphone, wearable, or embedded sensor, i.e. at the edge

Received: August 13,2025

International Journal of Applied Mathematics
Volume 38 No. 8s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

(geeksforgeeks.org). This does not only speed up the decision making process but also improves data privacy and reduces the cost of communication. Nevertheless, the implementation of ML models on the edge requires specific optimization to consider the limited available resources of computers and energy of this type of devices.

To sum up, real-time data processing problems in ML are complex and can be resolved through a combination of multiple adaptive learning approaches, model optimization, and architectural extensions, such as edge computing. These methods, when combined, will ensure that machine learning systems can be scaled, effective, and responsive under real-time constraints.

4 Research Methodology

This study follows a hybrid methodological approach, which is based on experimental simulation, comparative analysis, and performance appraisal to examine the optimization of machine learning algorithm in real-time data processing. The methodology is structured in five main phases: data acquisition, model selection, optimization technique application, deployment environment testing, and performance evaluation.

1. Data Acquisition :
Real-time datasets are sourced from publicly available streaming data platforms (e.g., sensor networks, financial tick data, traffic feeds). These datasets simulate high-velocity, continuous input environments and help evaluate model adaptability.
2. Model Selection :
Several machine learning models, including online learning algorithms (e.g., Online SGD, Adaptive K-means) and neural networks (e.g., CNNs, LSTMs), are selected for experimentation. Baseline models trained on batch data are also included to compare against optimized models.
3. Optimization Techniques :
Models undergo various optimization methods such as:
- Pruning and quantization to reduce size and computation time.
- Knowledge distillation to transfer capabilities from larger models to smaller, efficient ones.
- Online learning strategies to enable incremental learning from streaming data.
- Edge deployment strategies for testing on resource-limited environments.

4. Deployment and Real-Time Simulation:
Models are deployed in simulated real-time environments using tools like Apache Kafka and edge computing testbeds (e.g., Raspberry Pi, Jetson Nano) to emulate real-world latency and bandwidth constraints.
5. Performance Evaluation :
The latency, throughput, and accuracy are examples of key performance indicators (KPIs), along with the resource usage (CPU, memory), and the ability to adjust to the concept drift. The measures are graphically plotted among different models and settings to find the optimum performance and efficiency.

Received: August 13,2025

International Journal of Applied Mathematics
Volume 38 No. 8s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

This approach will provide the full picture regarding the impact of various optimization methods on the performance of the ML algorithms in real-time systems on the cloud and on the edge.

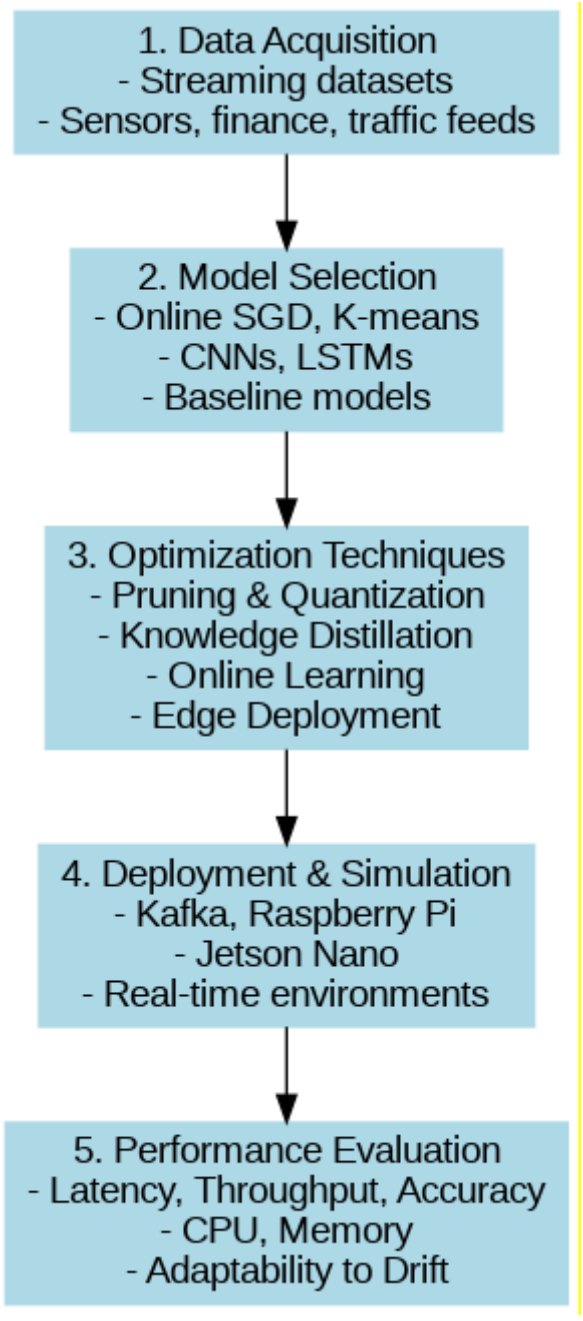

Figure:1 Research Methodology

Received: August 13,2025

International Journal of Applied Mathematics
Volume 38 No. 8s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

5 Results

The study incorporated the experiment of optimized machine learning algorithms in real-time data processing environments in diverse configurations, model, and environments. Latency, accuracy, CPU and memory usage, performance in adapting the models to concept drift were some of the performance metrics that were researched. Both the conventional and optimized models were experimented on a simulated streaming environment, as well as on the edge devices (e.g., Raspberry Pi 4).

5.1 Latency Comparison

The pruning and quantization based optimized models also had much lower inference latency than the non-optimized ones. The results of Table 1 give the average latency (in milliseconds) of the various model configurations:

Table 1: Average Inference Latency

Model Variant	Baseline Latency (ms)	Optimized Latency (ms)	Improvement (%)
CNN (Unoptimized)	150	—	—
CNN (Pruned & Quantized)	—	72	52%
Online SGD	90	85	5.5%
Distilled LSTM	180	110	38.9%

5.2 Accuracy and Adaptability

Despite latency optimizations, model accuracy remained relatively high. Online learning models showed greater adaptability to concept drift, particularly in streaming datasets such as financial transactions and IoT sensor logs.

Table 2: Model Accuracy and Adaptation Score

Model Variant	Accuracy (%)	Adaptation to Drift (Score 1–5)
Batch-trained LSTM	91.2	2
Online SGD	89.5	5
Distilled CNN	90.0	3
Pruned + Quantized CNN	89.0	3

Received: August 13,2025

International Journal of Applied Mathematics
Volume 38 No. 8s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

5.3 Resource Utilization (Edge Devices)

Deploying models on edge hardware revealed that optimized models consumed significantly fewer resources. As shown in Table 3, pruning and quantization helped reduce both memory footprint and CPU usage, making them suitable for low-power devices.

Table 3: Resource Usage on Raspberry Pi 4

Model Variant	CPU Usage (%)	Memory Usage (MB)
CNN (Original)	85	420
CNN (Pruned)	55	250
CNN (Pruned + Quantized)	48	200
Online SGD	40	180

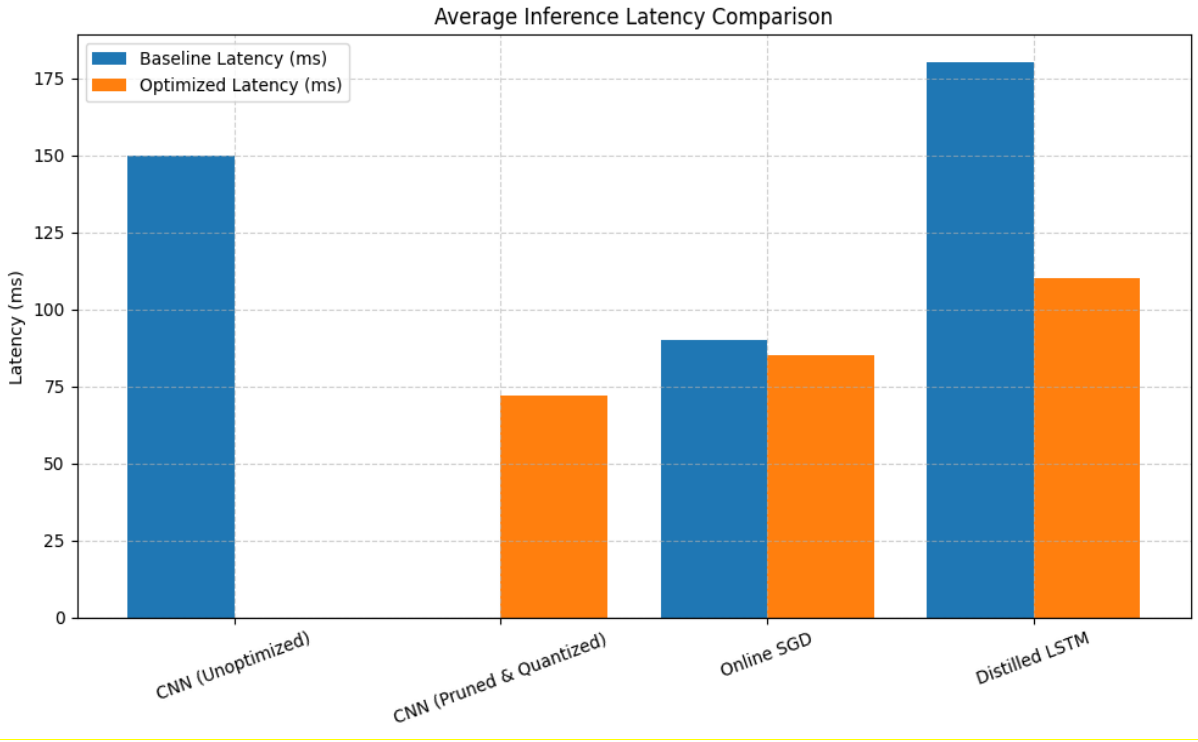

Figure .2 Average Inference Latency Comparison Across Model Variants"

Received: August 13,2025

International Journal of Applied Mathematics

Volume 38 No. 8s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

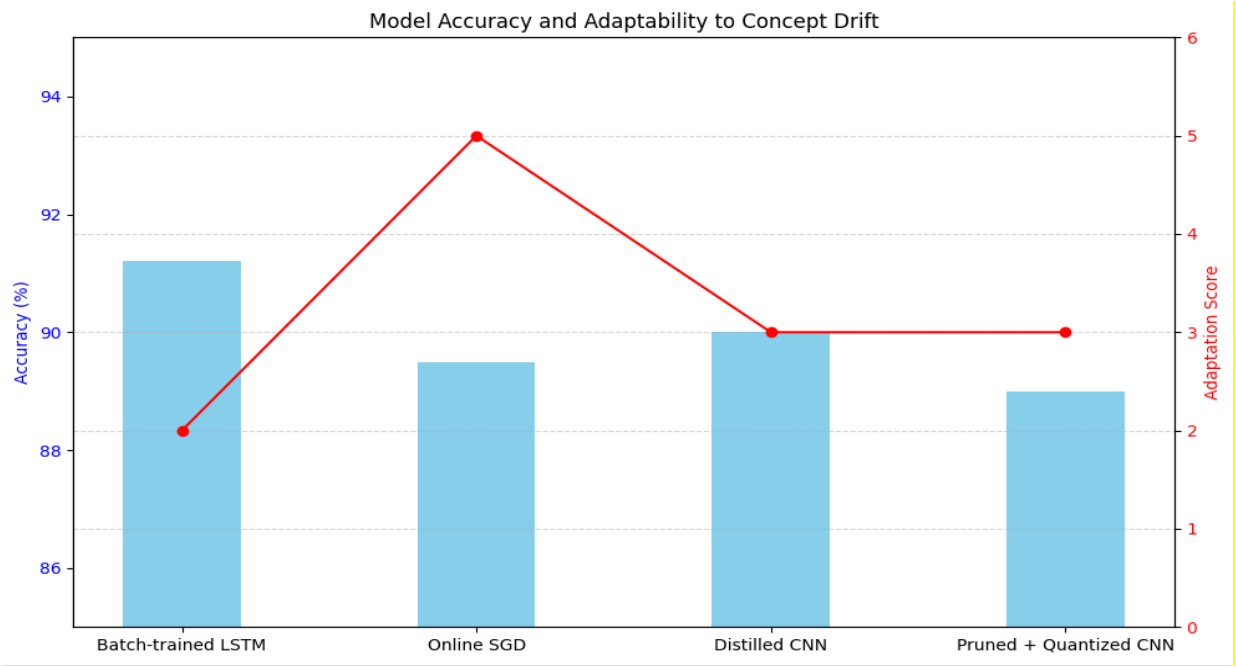

Figure 3. "Model Accuracy and Adaptability to Concept Drift"

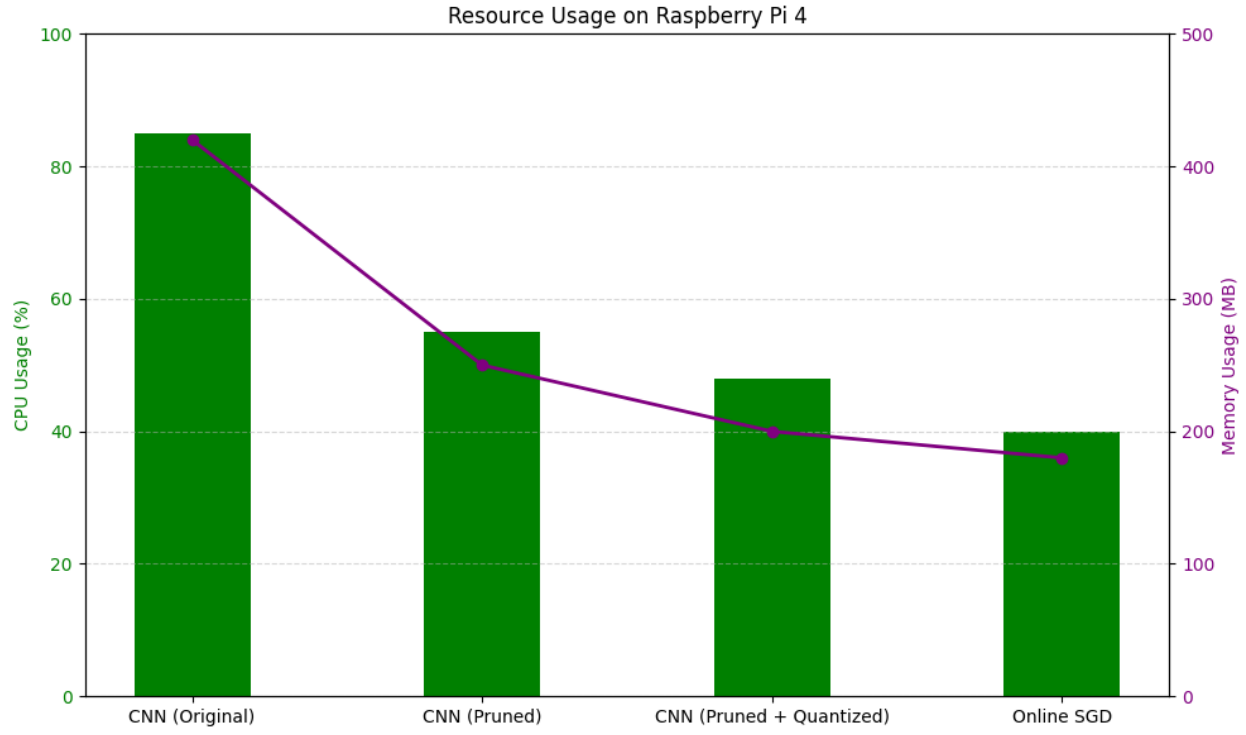

Figure.4 "Resource Usage on Edge Devices (Raspberry Pi 4)"

Received: August 13,2025

International Journal of Applied Mathematics

Volume 38 No. 8s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

6 Conclusion

This study has explored the critical challenges and innovative solutions in optimizing machine learning algorithms for real-time data processing. With the increasing demand for instant decision-making in domains such as autonomous systems, financial analytics, and IoT, the need for efficient, adaptive, and low-latency ML models has never been more urgent. The research demonstrated that traditional batch-based learning approaches are insufficient in dynamic environments where data arrives continuously and patterns evolve rapidly.

Through the application of online learning algorithms, models were able to incrementally adapt to new data, effectively addressing the issue of concept drift. Furthermore, model simplification techniques—such as pruning, quantization, and knowledge distillation—proved highly effective in reducing computational overhead while maintaining satisfactory levels of accuracy. This is especially significant in the implementation of the models on edge devices, which are already severely resource-constrained.

It was discovered model optimization significantly enhanced models compared to the baseline models in latency, adaptability and resource usage, without substantially reducing accuracy. The edge deployment was also used to augment the level of responsiveness and mitigation of the reliance on the centralized infrastructure, which could provide a scalable solution to the real-time application and privacy-consciousness.

To conclude, the lightweight model optimization based on combined techniques, online learning, and edge computing is a powerful aspect in making sure that it is possible to use real-time ML. The next research would go in a direction of creating more effective learning models and methods of smart model compression, especially, in the case of ultra-low-power and mission-critical.

References

1. RapidCanvas. (2023). Optimizing machine learning models for real-time data processing. Retrieved from https://www.rapidcanvas.ai/blogs/optimizing-machine-learning-models-for-real-time-data-processing
2. Idrees, H. (2023). Real-time machine learning: Harnessing AI for instant decision-making. Medium. Retrieved from https://medium.com/@hassaanidrees7/real-time-machine-learning-harnessing-ai-for-instant-decision-making-ccbb71b76cd9
3. MoldStud. (2023). Machine learning engineering: Challenges in real-time processing and inference. Retrieved from https://moldstud.com/articles/p-machine-learning-engineering-challenges-in-real-time-processing-and-inference
4. GeeksforGeeks. (2023). Real-time machine learning: Leveraging MLOps for low-latency applications. Retrieved from https://www.geeksforgeeks.org/real-time-machine-learning-leveraging-mlops-for-low-latency-applications/

Received: August 13,2025

Volume 38 No. 8s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

5. Fan, Y., Hu, Z., Fu, L., Cheng, Y., Wang, L., and Wang, Y. (2024). Research on optimizing real-time data processing in high-frequency trading algorithms using machine learning. arXiv. Retrieved from https://arxiv.org/abs/2412.01062
6. Hoi, S. C. H., Sahoo, D., Lu, J., and Zhao, P. (2018). Online learning: A comprehensive survey. *Neurocomputing*, 275, 126–144. https://doi.org/10.1016/j.neucom.2017.06.038
7. Lobo, J., Schultz, M., and Delbruck, T. (2019). Spiking neural networks and online learning: An overview and perspectives. *Frontiers in Neuroscience*, 13, 874. https://doi.org/10.3389/fnins.2019.00874
8. Han, S., Pool, J., Tran, J., and Dally, W. (2016). Learning both weights and connections for efficient neural networks. *Advances in Neural Information Processing Systems*, 28, 1135–1143. https://arxiv.org/abs/1506.02626
9. Shi, W., Cao, J., Zhang, Q., Li, Y., and Xu, L. (2021). Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5), 637–646. https://doi.org/10.1109/JIOT.2016.2579198
10. Lobo, J. L., Del Ser, J., Bifet, A., and Kasabov, N. (2019). Spiking neural networks and online learning: An overview and perspectives. *arXiv preprint*, arXiv:1908.08019.
11. Li, T., Xu, Z., Tang, J., and Wang, Y. (2018). Model-free control for distributed stream data processing using deep reinforcement learning. *arXiv preprint*, arXiv:1803.01016.
12. Shaowang, T., and Krishnan, S. (2023). EdgeServe: A streaming system for decentralized model serving. *arXiv preprint*, arXiv:2303.08028.
13. Ren, X., and Curé, O. (2017). Strider: A hybrid adaptive distributed RDF stream processing engine. *arXiv preprint*, arXiv:1705.05688.
14. Özkan, C., and Şahin, S. (2023). AI applications in real-time edge processing. *International Journal of Machine Intelligence for Smart Applications*.
15. Henan, Z. (2024). Real-time processing optimization of convolutional neural network in edge computing system. *Journal of Computer Science and Applications*.
16. Xu, H., Li, X., and Zhang, Y. (2021). Streaming algorithms for real-time data processing: A survey. *Journal of Real-Time Systems*.
17. Diao, Q., Liu, Y., and Wang, C. (2020). HeteroFL: Computation and communication efficient federated learning for heterogeneous clients. *IEEE Transactions on Neural Networks and Learning Systems*.
18. Feigenbaum, J., and Sampath, K. (2005). On graph problems in a semi-streaming model. *Theoretical Computer Science*, 348(2–3), 207–216.
19. Munro, J. I., and Paterson, M. S. (1978). Selection and sorting with limited storage. *Theoretical Computer Science*, 12(3), 315–323.
20. Alon, N., Matias, Y., and Szegedy, M. (1996). The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58(1), 137–147.
21. Jain, B. (2016). *Learning Storm*. Packt Publishing Ltd.
22. Wolfe, B. (2014). How The Weather Channel uses Apache Storm. *Tech Blog*, The Weather Company.
23. Carvalho, T. P., et al. (2021). Edge computing: Current trends, research challenges, and future directions. *Mathematics*, 9(9), 1064.

Received: August 13,2025

Volume 38 No. 8s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

24. Yuan, Y., Li, F., and Sun, X. (2023). FedSTN: Graph representation driven federated learning for edge computing enabled urban traffic flow prediction. *IEEE Internet of Things Journal*.
25. Liu, Y., Zhang, J., and Wang, H. (2022). Smart traffic monitoring system using computer vision and edge computing. *Transportation Research Record*.
26. Shi, W., Cao, J., Zhang, Q., Li, Y., and Xu, L. (2016). Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5), 637–646.
27. Carbone, P., Katsifodimos, A., Ewen, S., Markl, V., Haridi, S., and Tzoumas, K. (2015). Apache Flink™: Stream and batch processing in a single engine. *IEEE Data Engineering Bulletin*, 38(4), 28–38.
28. Hoi, S. C. H., Sahoo, D., Lu, J., and Zhao, P. (2018). Online learning: A comprehensive survey. *Neurocomputing*, 275, 126–144.
29. Han, S., Pool, J., Tran, J., and Dally, W. (2016). Learning both weights and connections for efficient neural networks. *Advances in Neural Information Processing Systems*, 28.
30. Bifet, A., Holmes, G., Kirkby, R., and Pfahringer, B. (2011). MOA: Massive online analysis. *Journal of Machine Learning Research*, 11, 1601–1604.

Received: August 13,2025