Metaversalize

www.meta.reapress.com

Research Expansion Alliance

Metaverse. Vol. 2, No. 3 (2025) 161–174.

Paper Type: Original Article

Efficient Processing of IoT Data Streams: Architectures, Algorithms, and Future Directions

Niloofar Qadiri[1,*], Farzaneh Kaviani[1]

[1] Department of Computer Science, Faculty of Artificial Intelligence and Advanced Social Technologies, Isfahan (Khorasgan) Branch, Islamic Azad University, Isfahan, Iran; niloofarqadiri.priv@gmail.com; f.kaviani@khuisf.ac.ir.

Citation:

Received: 19 January 2025 *Revised: 04 April 2025* *Accepted: 17 June 2025*	Qadiri, N., & Kaviani, F. (2025). Efficient processing of IoT data streams: Architectures, algorithms, and future directions. *Metaversalize*, 2(3), 161-174.

Abstract

The rapid proliferation of Internet of Things (IoT) devices has led to an exponential increase in the volume and velocity of data streams, necessitating real-time processing. Extracting actionable insights from these continuous data flows is essential for enabling intelligent decision-making across applications such as smart homes, industrial automation, and intelligent transportation systems. However, the resource-constrained nature of IoT devices and edge nodes—characterized by limited computational power, memory, and energy—presents significant challenges in achieving efficient and accurate data Stream Processing (SP). This paper presents a comprehensive review of state-of-the-art approaches for effectively managing and processing IoT data streams. We examine various architectural paradigms, including edge computing and distributed SP systems, designed to handle high-throughput, low-latency data streams. Additionally, we explore advanced algorithms, such as machine learning and deep learning techniques optimized for real-time analysis, prediction, and anomaly detection, as well as Approximate Computing (AC) methods and specialized data structures like Bloom Filters (BFs) and sketches that enhance resource utilization and reduce memory overhead. Furthermore, this review highlights critical challenges in the field, including data privacy, security, scalability, and fault tolerance, while identifying promising research directions toward building more scalable, energy-efficient, and intelligent IoT data SP systems. By synthesizing recent advancements and outlining future opportunities, this work serves as a valuable resource for researchers and practitioners seeking to address the complexities of real-time IoT data analytics.

Keywords: Internet of things, Data stream processing, Efficient algorithms, Edge computing, Anomaly detection, Approximate query, Resource optimization.

1 | Introduction

The widespread deployment of Internet of Things (IoT) devices has transformed various domains, leading to the creation of intelligent environments in sectors such as smart homes, industrial automation, intelligent transportation, and healthcare. This unprecedented growth in the number of connected devices – from simple

Corresponding Author: niloofarqadiri.priv@gmail.com
https://doi.org/10.22105/metaverse.v2i3.84

sensors and actuators to complex home appliances and smart vehicles – has led to the generation of unprecedented volumes and high-velocity continuous data streams [1]. Efficient processing of this massive flood of information is crucial for extracting timely insights and making informed decisions in critical applications, such as real-time fraud detection, predictive maintenance in industry, and remote patient monitoringTherefore, efficient algorithms and data structures are needed to enable the processing and Effective management of IoT data streams. Effective management is essential under these challenging conditions. There is a pressing need for new techniques that can perform the necessary computations and analyses close to the data source, intelligently summarize or filter data, and manage or store state or information with a reduced resource footprint. Approaches such as Approximate Computing (AC), which trade absolute accuracy for computational efficiency, and specialized Probabilistic Data Structures (PDS), such as Bloom Filters (BFs) and sketches, which can handle large datasets with minimal memory, have become increasingly important in this area [2], [3].

This review aims to provide a comprehensive and comparative analysis of the state-of-the-art in efficient algorithms and data structures for processing and managing IoT data streams, based on recent research. We aim to examine the various architectural paradigms, algorithms, and data structures that are being developed to address the challenges of limited resources and high-volume streaming data in IoT environments. By highlighting different approaches and their strengths and weaknesses, this review aims to provide a coherent overview of the current landscape and identify key areas for future innovation.

This paper is organized as follows. Section 2 provides background and necessary introductions related to IoT data streams and processing principles. Section 3 reviews different architectural paradigms for processing IoT data streams, including edge computing and distributed Stream Processing (SP) systems. Section 4 reviews efficient algorithms for real-time analysis and focuses on techniques adapted for streaming data in resource-constrained environments, including machine learning and anomaly detection methods. Section 5 examines the role of specialized data structures, such as PDS and approximation techniques, in optimizing memory and computation. Section 6 discusses key challenges inherent in efficient processing of IoT data streams, leveraging insights from anomaly detection and the general challenges of big data. Finally, Section 7 outlines promising future research directions and opportunities in this rapidly evolving field.

2 | Characteristics and Challenges of Internet of Things Data Streams

IoT is rapidly becoming a vital and interconnected technological landscape consisting of a network of sensor devices [4]. This phenomenon represents a significant shift from the "Internet of People" to the "Internet of Things" that enables various objects to be wirelessly connected. Wireless Sensor Networks (WSNs) serve as a key infrastructure to support the IoT and facilitate intelligent data collection from diverse contexts [5]. Many edge devices and real-world objects are integrated with wireless sensors to monitor and collect real-time data. With the rapid adoption and development of IoT devices, the demand for data processing and transmission is continuously increasing, generating a massive volume of data streams at high speed.

2.1 | Nature of Internet of Things Data

The data streams generated by IoT devices have distinct characteristics that pose specific challenges for processing [6]:

I. Continuous: data is continuously generated and received over time. SP pipelines must be able to handle these continuous data streams from different sources.

II. High volume: IoT devices generate a massive volume of data streams. The number of connected IoT devices is growing exponentially, with projections showing billions of devices worldwide. Working with large amounts of streaming time-series data can be a challenging task.

III. High speed: data can be received at high speeds and at different flow rates. IoT data volumes must be transferred to cloud servers at maximum speed to provide real-time information.

IV. Diversity: data can come from different sources in different formats, from IoT sensors to social networks. This diversity encompasses numerical values, text, images, sounds, videos, or a combination of these, which increases the complexity of processing.

V. Temporal component: data often has a temporal component, which means the evolution of one or more variables over time. This characteristic creates the need to preserve the order of the data, handle out-of-order data, and use time windows for more accurate analysis, especially in forecasting problems.

However, real-time processing of this data is not always straightforward. The large volume of data generated by IoT can lead to transmission collisions and energy waste due to data redundancy. Additionally, since devices may be vulnerable to security attacks or malfunctions, or unexpected events may occur, methods for detecting anomalies in data streams are crucial.

2.2 | Challenges from Limited Resources

IoT devices, especially small sensors deployed in various environments, are often challenging to charge after deployment, resulting in significant energy waste. Processing and transmitting data from these devices is expensive and consumes a lot of bandwidth, energy, and time. Resource-constrained IoT devices have limited computational resources. These constraints necessitate new approaches that strike a balance between computational efficiency and acceptable accuracy, while also ensuring system performance and energy efficiency [7].

Traditional computing methods used in these devices are plagued by high energy consumption and low performance. Therefore, novel approaches such as AC techniques are essential for building a new generation of low-power IoT devices. By balancing computational efficiency with acceptable accuracy, AC offers significant improvements in energy efficiency, speed, and area for resource-constrained IoT devices. Specific examples of energy-saving strategies and resource-relief strategies in IoT include the use of BFs, 6T SRAM, Dynamic Voltage/Frequency Scaling (DVFS) management techniques, IoT approximate processors, in-memory computing, and DRAM refresh rate tuning. Despite these advances, sensor devices still face significant challenges in terms of energy consumption, cost, throughput, and accuracy. Optimal energy utilization is critical for time-sensitive applications enabled by WSNs. Traditional static sink-based approaches in WSNs suffer from energy inefficiency, high latency, and uneven load distribution, with nodes close to the sink draining energy faster and creating "Energy holes." Hence, dynamic sink-based strategies are needed to optimize mobility, balance energy, and minimize query response time.

3 | Distributed Processing and Scalability in IoT Systems

Managing and processing the massive amounts of data generated by the growing number of IoT devices poses significant challenges in terms of distributed processing and scalability. Traditional centralized architectures, where all data is transferred to the cloud for processing, can lead to data latency and pose privacy and security risks. This approach may also involve transmission issues, such as congestion, bandwidth limitations, and a single point of failure [8]. Therefore, there is a need for an efficient and secure architecture for local data processing to reduce the cloud server's processing load and provide data redundancy at the cluster level. Decentralized architectures for IoT networks are needed to provide local storage, computation, and protection to overcome the limitations of centralized cloud systems.

Edge computing, as an emerging technology, has emerged as a key component in solving these challenges. It is considered an extension of cloud server services that moves cloud storage and processing power to the edge of the IoT network, closer to the sources of data generation [9]. The importance of processing data close to the source stems from several critical factors:

I. Latency reduction: processing data at edge nodes significantly reduces latency, which is critical for applications that require real-time feedback and fast response (such as autonomous driving).

II. Bandwidth optimization: preprocessing data at the edge and sending only the necessary information to the cloud can significantly reduce data traffic and save bandwidth, which is especially important in IoT where devices generate many data.

III. Increased data privacy and security: processing data locally at the edge reduces the transmission of sensitive data, thereby improving data security and privacy.

IV. Cloud server load reduction: processing data locally at edge nodes helps reduce the computational load, storage requirements, and network traffic of the cloud. The edge server can reduce the load on the cloud server by serving frequently used services in its own cache.

V. Improved energy efficiency: edge computing can help address energy consumption challenges in IoT devices.

VI. Fault tolerance: processing data locally ensures a degree of autonomy for the system, even when cloud services are unavailable.

Fog computing also offers advantages in data SP for IoT by processing data closer to its sources, thereby reducing latency and increasing network capacity [10]. Integrating edge, cloud, and high-performance computing environments is essential for processing significant data streams to ensure low latency and high throughput.

However, this distributed processing paradigm comes with its own challenges. One of the main challenges is communication scalability, which requires efficient strategies to manage large data transfers between distributed nodes [8]. It is crucial to deploy edge nodes securely and efficiently and to ensure reliable communication between edge nodes and the cloud server. Also, implementing data flow pipelines on low-end IoT hardware and managing these pipelines in a decentralized manner pose challenges. In data flow applications that adopt architectures such as serverless computing, it becomes crucial to address the challenges of managing stateful operations, coordinating resources in a volatile distributed environment, and ensuring low latency with inherent uncertainty.

3.1 | The Role of Edge AI and Approximate Query Processing

The rapid adoption of IoT devices and the increasing advancement of Artificial Intelligence (AI) underscore the growing importance of processing IoT data streams at the network edge and integrating AI into them [11]. Edge AI, which is a combination of edge computing and AI, not only changes the way data is processed but also opens up unprecedented opportunities for various application scenarios [12]. AI enables autonomous data analysis and decision-making directly on edge devices, thereby reducing the dependency on cloud servers. This makes the processing of many traditional applications that were previously difficult relatively simple.

AI-based edge computing is crucial for optimizing decision-making in various IoT applications:

I. In everyday life, autonomous driving is a prime example, where AI models on vehicle equipment make driving decisions using sensor data processed through edge computing technology.

II. Smart furniture, such as cameras and home robots, uses AI and edge computing to provide personalized services and security.

III. In the industrial IoT environment, the combination of edge computing and AI plays a vital role in production lines, enabling equipment failure monitoring and prediction and preventive maintenance.

However, the actual deployment of edge AI still faces many challenges. These challenges include limited computing resources on edge devices, which are generally less than those of cloud servers. Implementing large and complex AI models on edge devices requires developing larger and more efficient models, as well as considering hardware accelerators. Energy efficiency of edge computing is also a challenge, as most devices rely on battery power, and highly optimized AI algorithms can still generate significant energy consumption.

In the context of processing data streams generated by IoT devices, the need for fast query response is critical, especially in real-time environments. Approximate Query Processing (AQP) emerges as a crucial strategy for overcoming the challenge of efficiently processing large, high-speed data streams while providing timely query responses [13]. AQP operates on the principle of sacrificing a degree of accuracy in exchange for increased performance and improved energy efficiency. This approach is particularly beneficial for applications where absolute, precise results are not strictly required or where a certain degree of imprecision can be tolerated without significantly impacting the overall outcome. Key techniques used in AQP to achieve this balance include:

I. Sampling: involves analyzing a representative subset of data rather than processing the entire data set, which can significantly reduce computational complexity and speed up processing time.

II. Sketching: use of PDS to summarize large data sets in a quantitative space, providing approximate answers to queries with probabilistic error bounds. Examples include BFs for Approximate Membership Queries (AMQ), Count-Min Sketch (CMS) for frequency estimation, and HyperLogLog (HLL) for cardinality estimation.

III. Aggregation-based approaches: perform aggregation operations on summarized data (such as samples or sketches) rather than on the raw data. Data aggregation in IoT sensor networks is critical to reducing data redundancy and saving energy.

AQP finds significant applications in big data visualization. As the size of data sets increases, traditional visualization models struggle to provide fast, efficient, and interactive representations. Approximate visualization techniques, such as those based on sampling, bridge the gap between massive amounts of data and interactive presentation by accelerating the process and increasing performance.

As a result, edge computing, edge AI, and AQP are critical to enabling efficient, low-latency, secure, and energy-aware processing of the massive data streams generated by IoT devices. By moving computation and decision-making to the data source, they address key challenges of centralized cloud architectures and unlock the full potential of real-time IoT applications across various domains.

4 | Lightweight and Efficient Data Structures

Processing the massive, continuous data streams generated by IoT devices requires the use of data structures that are not only efficient in terms of computational speed but also remarkably lightweight in terms of memory and energy consumption. This is mainly due to the inherent hardware limitations of many IoT and edge devices, which often operate with limited processing power, low memory capacity, and tight energy budgets. Traditional, memory-intensive data structures designed for powerful servers may be impractical or impossible to deploy effectively in these resource-constrained environments. Therefore, there is a significant need for data structures that can provide valuable insights from data streams with minimal overhead.

4.1 | Probabilistic Data Structures

To overcome the challenge of limited resources while still supporting fast queries on large data streams, PDS offers an attractive solution. PDSs are data structures that provide approximate answers to queries, typically with a probabilistic guarantee of error, while utilizing significantly less memory and computational resources than deterministic data structures that provide exact answers. They are particularly well-suited for scenarios where a level of inaccuracy is acceptable in exchange for a significant increase in performance.

A BF is a space-efficient PDS used to test whether an element is a member of a set [14]. A key feature of BFs is that they may yield false positives (indicating the presence of a component when it is not) but never false negatives (an aspect that is in the set is always reported as present). This one-sided error guarantee is critical in many applications. BFs are renowned for their compactness and efficiency in determining membership and are widely utilized in various fields, including IoT, networking, databases, and bioinformatics. Their space efficiency and fast membership queries make them valuable in scenarios where memory is a significant constraint. In the context of IoT, BFs are widely used, especially in wearable electronics, where battery life is a considerable concern.

A recently proposed class is the Learned Bloom Filter (LBF), which combines the CBF with a learning model LBF aims to provide the same one-sided error guarantee as CBF but with potentially better performance for the same memory budget [15]. A Standard Learned Bloom Filter (SLBF) uses a learning model as a prefilter before a supporting CBF. The supporting CBF is queried only for inputs that the learning model predicts are not in the set, which ensures that the critical completeness property (no false negatives) is maintained throughout the entire structure.

While efficient, the use of BFs in IoT and other applications also raises concerns about privacy. Research has shown that CBCs can be vulnerable to set reconstruction attacks, where an attacker with access to the internal state of the filter may be able to infer the original set with high probability. Addressing the privacy of data stored in a BF has been identified as a significant open problem. Recent work provides a detailed differential privacy-based analysis of BFs. It is also noted as the first work to analyze and address the privacy of learned BFs under any exact model. Privacy-preserving algorithms, such as Nickel and Dime [16], have been proposed based on random response mechanisms. These algorithms modify the input set stored by the BF rather than modifying the internal state. Nickel adds privacy without compromising the one-way error guarantee of BF, while Dime introduces a significant trade-off in terms of false negative probability.

Challenges in BF design include managing the trade-off between the false positive rate and memory space, selecting optimal hash functions, accommodating their predefined size that is difficult to change for large or growing datasets, and handling element deletion. Despite these challenges, efforts continue to refine these structures for better performance and security in resource-constrained environments. An enhanced Bloom filter (eBF) has been developed specifically for intrusion detection in IoT networks, demonstrating significant memory savings over standard BFs and other variants while maintaining fast and accurate performance. A lightweight BF accelerator for IoT applications has also been implemented using hardware such as FPGAs. It is.

4.2 | Other Possible Data Structures and Schemas

Beyond BFs, several other data structures are well-suited to handling the continuous and often extensive nature of data streams, particularly those used in AQP and data summarization. Sketching data structures is a family of structures designed to summarize large data sets in a small amount of space [17]. They are valuable for providing approximate answers to queries and for data compression. Sketching-based data structures are a common technique for estimating frequencies in data streams, including the Count Sketch (CS), the CMS, and the Count-Mean-Min Sketch (CMMS). For example, the Count-Min Sketch effectively trades off accuracy for a significant reduction in memory consumption, making it a powerful solution for estimating item frequency in large datasets where precise storage is impossible. However, CMS can suffer from overestimation due to hash collisions, and its accuracy depends on the quality of the hash functions used. Count-Sketch is a viable alternative that supports decrement operations and negative counts, which CMS lacks. These constructs are critical for querying continuous data streams with limited resources, as they maintain compact summaries for efficient estimation of query results.

Another important PDS is the HLL. HLL is a mighty approximation algorithm used to estimate the cardinality (number of unique elements) of a large set [18]. It provides reasonably accurate estimates with much less memory than methods that require storing all elements. HLL is utilized in network monitoring, web analytics, data analysis, and database management. Like CMS, HLL has limitations, such as not supporting element deletion; however, variants like the sliding HLL address this problem. An extension called KHyperLogLog (KHLL) is used to estimate re-identifiability and joinability risks in large databases.

5 | Algorithmic and Hardware Design

Approximate Calculations (AC) is a paradigm that significantly impacts the design of both algorithmic and hardware systems, particularly in resource-constrained IoT environments [2]. Given the inherent hardware limitations of many IoT and edge devices, such as limited processing power, memory, and energy budgets, traditional computational approaches designed for more powerful systems are often impractical. AC offers a way to enable complex computational tasks on these devices by strategically sacrificing a degree of accuracy in exchange for increased efficiency, including speed, reduced energy consumption, and reduced resource usage.

5.1 | Hardware Design

Processing massive, continuous data streams can be applied at different levels of the hardware stack to improve the performance of IoT applications:

Circuit level: a critical focus is on redesigning basic arithmetic units such as adders, multipliers, and dividers. These units are central to many fault-tolerant applications. By designing approximate versions, significant reductions in power consumption, area, and latency can be achieved. Examples include various types of approximate adders, multipliers, and dividers. Hardware implementations of primitive and activation functions, which are critical to many algorithms, including those used in IoT, also benefit from approximation techniques. Approaches such as lookup tables, polynomial approximation, shift-and-add algorithms (such as CORDIC), and hybrid methods are employed, often sacrificing speed/area for improved accuracy [19].

Architecture level: design of specialized approximate processing units and memory systems is essential.

I. Approximate memory: techniques are being explored to make memory systems more energy efficient. These include voltage scaling for SRAMs, reducing refresh rates for DREAMs, and compressing or encoding data before storage. These methods leverage fault tolerance in applications. For IoT devices, especially those that rely on battery power or operate with limited memory, approximate memory designs such as fault-tolerant configurable SRAMs and refresh-free DRAM techniques are being developed.

II. Approximate processors and accelerators: these are integrated units designed with controlled imprecision to increase computational efficiency. Low-power architectures such as ARM and open source RISC-V are being explored for IoT approximate processors, specifically for tasks such as machine learning. Accelerators such as TPUs, which use reduced precision, represent an architectural approach.

III. DVFS management: techniques such as DVFS, Voltage Over Scaling (VOS), dynamic power management (DPM), and near-threshold voltage (NTV) operation are widely used in IoT to save energy. These methods use idle times or reduce voltage/frequency at the cost of potential faults, which can be managed in fault-tolerant applications.

Data level (hardware aspects): the use of approximate data types and data structures directly affects the hardware requirements for storage and manipulation. Techniques such as precision scaling (using fixed-point or floating-point with lower precision) and quantization reduce memory and computational costs. These are critical for deploying models on resource-constrained edge devices. Hardware accelerators such as GPUs and TPUs are designed to handle reduced-precision formats efficiently.

5.2 | Algorithm Design

AC also impact algorithm design, making them suitable for implementation on resource-constrained IoT hardware:

Software level: algorithms can be designed or modified to introduce approximations into their code. This includes identifying non-critical computations where accuracy can be degraded. Techniques include computation skipping (e.g., perforation loops or selective skipping of instructions), iterative refinement (early stopping iterations), function approximation (replacing exact functions with approximate functions, possibly using neural networks), pruning (removing redundant parts of models such as neural networks), and lazy synchronization in parallel algorithms to reduce waiting times. These methods trade off performance and error. Approximate memoization techniques can store and reuse approximate results for similar inputs, reducing redundant computation.

Data level (algorithmic aspects): designing algorithms that operate on approximate data representations is key. Data sampling algorithms analyze representative subsets of large datasets or streams to provide faster, approximate results with defined error bounds. PDSs are inherently algorithmic and are designed for approximate queries with minimal memory overhead. BFs are used for approximate membership testing, while sketch data structures (e.g., CMS, HLL) are employed for frequency estimation and cardinality counting in data streams, and MinHash is used for set similarity estimation. These structures enable efficient data processing and summarization on devices with limited resources.

5.3 | Co-Design and Frameworks

Effective use of AC for IoT often requires a cross-layer co-design approach, integrating techniques at the hardware, architecture, software, and data levels. Frameworks and tools are being developed to support this, helping designers manage the trade-off between accuracy and efficiency and automating the process of identifying approximation opportunities and applying techniques. For example, Approximate Logic Synthesis (ALS) tools automate the design of approximate digital circuits.

In essence, AC enables the design of specialized and resource-efficient hardware components and systems, while also guiding the development of algorithms that can operate effectively with controlled imprecision on these limited platforms. This dual effect is critical to enabling the deployment of complex data processing and AI tasks on the growing number of IoT devices.

Table 1. Comparison of high-level approaches to approximate calculations.

Level of Abstraction	Common Techniques	Key Benefits	Common Exchange	Typical Application Areas
Fault-tolerant applications, power-constrained IoT devices, digital circuits	Loss of accuracy (controllable), design complexity	Significant reduction in power consumption, area and latency	Redesign of arithmetic units (adders, multipliers, approximate dividers), elementary functions/approximate activation (lookup tables, CORDIC)	Circuit
Resource-constrained IoT devices, machine learning at the edge, battery-life-critical applications	Controlled inaccuracy, potential errors, verification challenges	Energy efficiency, computational efficiency, reduced memory footprint	Approximate memory (SRAM voltage scaling, DRAM refresh rate reduction), approximate processors/accelerators (low-power IoT processors, reduced-precision TPUs), DVFS management	Architecture
Deploying AI models on edge devices, processing big data	Loss of accuracy, need for hardware support	Reduced memory and computing costs, efficient storage	Precision scaling (fixed/floating point with lower precision), quantization	Data (Hardware Aspects)

Table 1. Comparison of high-level approaches to approximate calculations.

Level of Abstraction	Common Techniques	Key Benefits	Common Exchange	Typical Application Areas
Machine learning algorithms, parallel algorithms, analytical programs	Loss of precision, error handling, coding complexity	Increased performance, reduced overhead, faster execution time	Skipping computations (e.g. loop perforation), iterative refinement (early stopping iterations), function approximation, pruning, lazy synchronization, approximate memoization[7]	Software
Data flow analysis, approximate queries, resource-constrained devices, membership testing, frequency estimation	Possible errors, controlled imprecision, need for optimal hash functions	Faster approximate results, minimal memory overhead, efficient data processing/summarization	Data sampling algorithms, PDS such as BFs (for membership queries), sketching structures such as CMS (for frequency/cardinality estimation), MinHash	Data (Algorithmic Aspects)

AC (AxC) is a growing paradigm that strategically trades off accuracy in exchange for higher performance (speed, energy efficiency, memory footprint). This approach is particularly well-suited for resource-constrained IoT environments and fault-tolerant applications. The following table compares high-level approaches to AC based on their level of abstraction, standard techniques, main advantages, trade-offs, and application areas, using information provided in the references.

6 | Challenges and Future Directions

AC and related efficiency paradigms offer significant potential for enhancing the performance and energy efficiency of systems, especially in resource-constrained IoT environments. However, fully realizing the benefits of these approaches involves overcoming several complex challenges and identifying promising avenues for future research.

6.1 | Existing Challenges

Despite advances in approximate computation and efficiency paradigms, several challenges remain, especially when applied to the dynamic and resource-constrained nature of IoT:

I. Scalability, security, and privacy in IoT data SP: the rapid adoption and development of IoT devices is leading to the growing trend of edge computing, which is changing the way data is processed. Edge computing brings more innovative solutions to various application scenarios in IoT. However, the resource and energy efficiency of edge computing has certain limitations. With the increasing use of IoT end devices, a massive amount of sensor data is being generated. Managing this significant volume of data is a major challenge. Data centers, such as cloud computing storage spaces, are struggling to provide the necessary resources for transmission due to the explosion in data volume. The growing number of connected IoT devices is increasing exponentially. This growth brings new challenges for data processing and transmission.

Security and privacy issues are significant concerns in IoT environments. The integration of edge computing and AI can significantly enhance the data privacy and security concerns that are of most concern to individuals. Edge computing enables local data processing, reduces the transmission of sensitive data, and thereby enhances data security. Edge computing provides unprecedented opportunities for many application scenarios, including improved data privacy and secure management. Strong security measures are increasingly necessary in applications such as autonomous driving, innovative furniture, and industrial automation.

Existing decentralized architectures for large-scale IoT networks have not consistently addressed security and privacy concerns. Traditional cloud servers employ a centralized architecture that connects IoT devices for processing and storage, which can lead to transmission issues, security vulnerabilities, or a single point of failure. A decentralized architecture is required to manage security and energy-related issues in IoT-based networks. Blockchain technology has been widely researched due to its decentralized nature and its potential to protect IoT devices from security threats and rogue service providers.

The security implications of approximation computations, especially in sensitive applications, require careful consideration. AC can complicate reverse engineering, but it can also introduce new target areas for hardware trojans in circuits that control approximation levels. Assessing the security of approximation circuits is challenging because their defenses against passive side-channel attacks can vary with voltage-frequency settings. Approximation circuits may be vulnerable to fault injection attacks, especially at operational limits, although the full effects and effectiveness of countermeasures are still unclear. Processing-In-Memory (PIM) introduces changes to security models due to architectural modifications, programming models, side-channel vulnerabilities, device reliability concerns, and potential hardware trojans. Developing secure and privacy-preserving AC methods is a critical area of future research.

II. Heterogeneous and incomplete data management: anomaly detection in bright environments, often based on IoT data streams, is challenging due to complex dynamics. Commonly encountered challenges include scarcity (e.g., lack of labeled data, unbalanced datasets) and complexity (e.g., noisy data, high dimensionality, conceptual drift). Contextualization challenges arise from the need to consider complex relationships between influencing factors to distinguish abnormal events from regular events under specific conditions. Data streams can originate from various sources in different formats, making their processing unwieldy, especially in real-time applications. The diversity of data that can be received increases the complexity of working with big data. Managing out-of-order and delayed data is a fundamental challenge in SP.

Data transmission delays due to distributed sensors and diverse communication channels are a critical challenge in industrial IoT anomaly detection, making models essential to account for these delays to ensure accurate real-time detection. Managing the temporal aspects of data, particularly the order of streams and handling out-of-order data, is crucial for accurate and timely data analysis. Integrating stored (historical) and streaming data is also a key aspect of data management.

III. Ensuring Quality of Service (QoS) in dynamic environments: satisfying Quality of Service (QoS) under workload changes has been a long-standing research challenge in SP systems. Stream processors lack control over the rate of incoming events and must be adaptable to handle the load. Ensuring predictable results and maintaining real-time response are essential requirements for real-time SP. Scalability, the ability of a system to handle growing workloads without compromising performance, is critical and can be achieved through scale-in, scale-out, and scale-up strategies.

Fault tolerance and rapid recovery are crucial for maintaining high availability and data integrity in SP. Consistent SP is an open research problem due to the challenging nature of processing unbounded streams in a distributed setting. High availability is addressed through various replication approaches.

Approximate computations essentially involve a trade-off between accuracy and efficiency (speed and energy consumption). Managing this trade-off is a key challenge in approximate programming. For approximate memory, a careful evaluation is necessary to achieve a balance between the benefits of minimizing accuracy and the degree of precision required for a particular application. ALS faces the challenge of accommodating varying accuracy requirements while managing power and delay variations. Designing high-quality, configurable circuits that can be tuned to different levels of accuracy in real-time is required.

In dynamic WSN scenarios, the computational efficiency of query processing relies on lightweight optimization approaches that ensure consistency. Ensuring efficient and reliable data delivery services, as well as longer network lifetime, are critical issues in mobile sensor networks. This requires a balance between efficient data transmission and energy conservation.

6.2 | Future Research Directions

Based on the current landscape and existing challenges, several promising avenues for future research emerge:

6.2.1 | Deeper integration of AI and machine learning with stream processing at the edge

The combination of edge computing and AI is changing the way data is processed and providing unprecedented opportunities for many application scenarios. AI-based edge computing is driving the development of the IoT. Deploying AI algorithms on edge devices to process data can greatly achieve low latency, high performance, and high privacy protection.

AI-based edge computing is having a significant impact on the development of the IoT. AI plays a fundamental role in delivering scalable results on real-world big data. Machine learning enhances IoT by analyzing vast data sets for actionable insights, which is critical for applications such as wearables and smart devices. Embedded processing close to the sensor is often preferred over cloud computing due to privacy, latency, and bandwidth constraints.

Future research emphasizes the need for scalable and lightweight solutions for anomaly detection to adapt to environments with limited computing power, ensuring efficient and real-time performance on edge devices. AC is well-suited for machine learning and AI applications at the edge, where small precision losses can be tolerated for significant performance gains. Future research directions include the development of real-time analytics and algorithms for Big Data time series streams, particularly the application of machine learning and deep learning algorithms for online education to facilitate real-time analysis, prediction, and anomaly detection.

6.2.2 | Further development of PDS and AQPs for more complex IoT scenarios

PDS that provide approximate answers to membership queries, such as BFs, are common. Extending privacy-preserving structures to other PDSs, such as (CMS), is a direction for future work.

Approximate data structures, such as BFs or (CMS), are helpful in saving resources and time by providing probabilistic capabilities. These structures enable efficient data processing and summarization on devices with limited resources. Sampling techniques play an essential role in stream analysis, providing faster and approximate results with estimated error limits. AQP provides fast and approximate answers to queries on large datasets using techniques such as sampling.

Future research should prioritize the efficiency of query processing techniques in WSNs by developing methods that adjust to changing network conditions and dynamic application needs, ensuring continuous optimization in dynamic IoT contexts.

New approaches to co-optimization of energy, latency, and accuracy: Approximate computation is a promising paradigm that achieves significant improvements in reducing overhead costs (energy, area, and latency) at the cost of acceptable accuracy degradation. Error metrics are emerging as a new design parameter that can be traded off to increase performance or reduce energy consumption. AC offers significant improvements in power and performance by reducing numerical parity for fault-tolerant applications.

Effective use of AC requires a cross-layer co-design approach that integrates techniques at the hardware, architecture, software, and data levels. Cross-layer approaches have emerged as a powerful tool for intelligently combining approximation techniques to maximize efficiency gains while meeting user-defined quality constraints. These can lead to significant improvements in speed, power consumption, and overall optimization.

Approximate adders are designed to optimize power consumption, area, and latency while allowing for controlled imprecision. Approximate multipliers aim for low power, high performance. Approximate dividers solve the problems of high latency, large area, and power consumption through imprecise calculations.

DVFS management techniques dynamically adjust voltage and frequency to optimize power consumption and performance based on the workload. AxC techniques, such as voltage scaling and reduced precision in memory and processing units, contribute to energy efficiency.

Shannon-inspired statistical computing leverages the statistical properties and integrated computation in memory/sensor arrays to create robust and energy-efficient systems that ensure reliability even in the presence of hardware noise/errors, and enable operation at lower SNR levels. Integrating this with AC increases robustness, consistency, and energy efficiency.

Dynamic and adaptive approximation techniques that can adjust the approximation level based on application requirements, input data characteristics, and available resources are a new future direction. This ensures an optimal trade-off between accuracy and performance.

In the context of query processing in WSNs for IoT, dynamic strategies aim to increase scalability, reliability, and energy efficiency by optimizing sink mobility and balancing energy consumption. DSQPS aims to minimize query processing latency, reduce average energy consumption, and increase network lifetime and throughput. Future research should explore sophisticated algorithms for low-power routing of moving wells.

Addressing these challenges and exploring the aforementioned future directions will be crucial to unlocking the full potential of approximation computing and other efficiency paradigms in the complex and dynamic IoT landscape.

7 | Conclusion

In conclusion, this review provided a comprehensive analysis of the state-of-the-art algorithms and data structures for processing and managing the vast data streams generated by the IoT. We established that the exponential growth of connected devices has necessitated a paradigm shift from traditional centralized cloud architectures to decentralized models like edge and fog computing. These paradigms were shown to be crucial for reducing latency, optimizing bandwidth, enhancing privacy, and lowering the computational load on cloud servers, thereby enabling real-time applications in domains such as autonomous driving and industrial automation. To operate within the resource-constrained nature of IoT and edge devices, we identified several essential techniques. Key strategies examined included AC, which traded controlled accuracy for significant gains in energy and performance, and the use of lightweight PDS like BFs and sketches to manage massive data volumes with minimal memory overhead. Furthermore, we explored how the integration of Edge AI enabled autonomous data analysis and decision-making directly on devices, reducing dependency on the cloud and allowing for more intelligent and responsive systems.

Despite these significant advancements, our analysis highlighted that several challenges persist, defining the trajectory for future research. Ensuring data privacy and security remains a primary concern, particularly as decentralized architectures and approximate data structures introduce new vulnerabilities. The inherent complexity of managing heterogeneous, incomplete, and out-of-order IoT data streams continues to pose difficulties for real-time analytics. Future work, therefore, should focus on several promising directions to address these issues. This includes the deeper integration of AI and machine learning at the network edge, enabling more sophisticated and low-latency analytics. There is also a critical need to develop more advanced PDS and AQP techniques that can handle complex IoT scenarios while offering stronger privacy guarantees. Finally, a continued emphasis on cross-layer co-design approaches—holistically optimizing hardware, software, and algorithms—will be essential to adaptively balance the trade-offs between accuracy, performance, and energy efficiency, ultimately unlocking the full potential of real-time IoT data analytics.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Data Availability

All data are included in the text.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] Yadav, S. R., Bhushan, A., Kumar, P. R., & Quadri, M. N. (2022). *Internet of things and its applications*. AG Publishing House (AGPH Books). https://www.flipkart.com/internet-things-its-applications/p/itm22f5ca49abf4b

[2] Dalloo, A. M., Humaidi, A. J., Al Mhdawi, A. K., & Al-Raweshidy, H. (2024). Approximate computing: Concepts, architectures, challenges, applications, and future directions. *IEEE access*, *12*, 146022–146088. https://doi.org/10.1109/ACCESS.2024.3467375

[3] Kumar, M., & Singh, A. (2022). Probabilistic data structures in smart city: Survey, applications, challenges, and research directions. *Journal of ambient intelligence and smart environments*, *14*(4), 229–284. https://doi.org/10.3233/AIS-220101

[4] Allioui, H., & Mourdi, Y. (2023). Exploring the full potentials of IoT for better financial growth and stability: A comprehensive survey. *Sensors*, *23*(19), 8015. https://doi.org/10.3390/s23198015

[5] Ali, I., Ahmedy, I., Gani, A., Munir, M. U., & Anisi, M. H. (2022). Data collection in studies on internet of things (IoT), wireless sensor networks (WSNs), and sensor cloud (SC): Similarities and differences. *IEEE access*, *10*, 33909–33931. https://doi.org/10.1109/ACCESS.2022.3161929

[6] Wójcicki, K., Biegańska, M., Paliwoda, B., & Górna, J. (2022). Internet of things in industry: Research profiling, application, challenges and opportunities—A review. *Energies*, *15*(5), 1806. https://doi.org/10.3390/en15051806

[7] Abdulhussain, S. H., Mahmmod, B. M., Alwhelat, A., Shehada, D., Shihab, Z. I., Mohammed, H. J., ... & Hussain, A. (2025). A comprehensive review of sensor technologies in IoT: Technical aspects, challenges, and future directions. *Computers*, *14*(8), 342. https://doi.org/10.3390/computers14080342

[8] Kuchuk, H., & Malokhvii, E. (2024). Integration of IoT with cloud, fog, and edge computing: A review. *Advanced information systems*, *8*(2), 65–78. https://doi.org/10.20998/2522-9052.2024.2.08

[9] Devan, S. (In Press). Edge computing for distributed IoT data management in smart cities. *Computational engineering and technology innovations*. https://doi.org/10.48314/ceti.vi.46

[10] Shwe, T., & Aritsugi, M. (2024). Optimizing data processing: A comparative study of big data platforms in edge, fog, and cloud layers. *Applied sciences*, *14*(1), 452. https://doi.org/10.3390/app14010452

[11] Ficili, I., Giacobbe, M., Tricomi, G., & Puliafito, A. (2025). From sensors to data intelligence: Leveraging IoT, Cloud, and edge computing with AI. *Sensors*, *25*(6), 1763. https://doi.org/10.3390/s25061763

[12] Bourechak, A., Zedadra, O., Kouahla, M. N., Guerrieri, A., Seridi, H., & Fortino, G. (2023). At the confluence of artificial intelligence and edge computing in IoT-based applications: A review and new perspectives. *Sensors*, *23*(3), 1639. https://doi.org/10.3390/s23031639

[13] Ma, Q., Shanghooshabad, A. M., Almasi, M., Kurmanji, M., & Triantafillou, P. (2021). *Learned approximate query processing: Make it light, accurate and fast* [presentation]. Conference on innovative data systems, (CIDR21). https://wrap.warwick.ac.uk/id/eprint/159826/1/WRAP-Learned-approximate-query-processing-make-light-accurate-fast-2021.pdf

[14] Cishugi, E. S., Smit, T. T., Forlin, B., Cazzaniga, C., Chen, K. H., & Ottavi, M. (2025). Bloom filters for soft error detection: Neutron and fault injection validation. *2025 IEEE 31st international symposium on on-line testing and robust system design (IOLTS)* (pp. 1–7). IEEE. https://doi.org/10.1109/IOLTS65288.2025.11116913

[15] Tirmazi, H. (2025). *A privacy model for classical & learned bloom filters*. https://arxiv.org/abs/2501.15751

[16] Jung, F. (2022). Nickel-and-diming? *ATZ worldwide*, *124*(10), 3. https://doi.org/10.1007/s38311-022-0877-9

[17] Marchet, C., Boucher, C., Puglisi, S. J., Medvedev, P., Salson, M., & Chikhi, R. (2021). Data structures based on k-mers for querying large collections of sequencing data sets. *Genome research*, *31*(1), 1–12. https://genome.cshlp.org/content/31/1/1.short

[18] Karppa, M., & Pagh, R. (2022). *Hyperlogloglog: Cardinality estimation with one log more* [presentation]. Proceedings of the 28th acm sigkdd conference on knowledge discovery and data mining (pp. 753–761). https://doi.org/10.1145/3534678.3539246

[19] Kulkarni, K., Joshi, N., Lale, I., & Nirmal, N. G. (2024). CORDIC algorithm implementation using FPGA. *International conference on information and communication technology for competitive strategies* (pp. 167–182). Springer. https://doi.org/10.1007/978-981-96-5751-3_15