MQTT Broker Congestion Mitigation Using Huffman Deep Compression

Ammar Nasif, Zulaiha Ali Othman, Nor Samsiah Sani, Yousra Abudaqqa
Department of Centre of Artificial Intelligence Technology-Faculty of Information Science & Technology,
Universiti Kebangsaan Malaysia, 43600 Bangi Selangor, Malaysia

***Abstract*—This study presents an improved MQTT protocol designed to address broker congestion and connection overflow in large-scale IoT networks. The proposed method integrates Huffman Deep Compression (HDC) at the publisher side to mitigate network traffic and latency. Unlike standard MQTT, which suffers from broker overload, our approach applies efficient data compression on resource-constrained sensor devices prior to publishing. The proposed approach was validated on a real-world air pollution dataset collected from the Tanjung Malim monitoring station in Malaysia, using ESP8266-based IoT nodes. Experimental results demonstrated that broker congestion was reduced by 84.26% for QoS 0 and 79.6% for QoS 1, significantly outperforming both standard MQTT and the state-of-the-art MRT-MQTT (58% and 45%, respectively). The method attained a high compression ratio of 2.62, which directly led to a dramatic reduction in power consumption from 2,664,864 to 63,216 mA (QoS 0) and from 3,155,760 to 49,168 mA (QoS 1). This substantial saving in current consumption contributes to extended device lifetime and enhanced energy efficiency. The findings highlight the potential of this enhanced protocol to support massive IoT deployments by minimizing network overhead at the broker.**

Keywords*—Compression; network congestion; connection overflow; deep learning; IoT; broker congestion; IoT network; sensor; latency reduction; publishers; broker; MQTT; power consumption

I. Introduction

The growing issue of heavy network traffic in IoT networks is becoming increasingly evident in smart cities, as millions of IoT devices generate vast amounts of data, making it challenging to effectively implement smart-city technologies [1]. Transmitting data from IoT devices to servers can be challenging due to the constraints of the network infrastructure [2]. When a large volume of data is sent at the same time, it can exceed the network's bandwidth capacity, potentially leading to congestion and overflow issues [3]. Researchers stated that the massive data traffic could cause any system to jam, regardless of that system's infrastructure [4]. While other researchers stated that in case all the sensors are transmitting data to a central server at the same time, the server may become overwhelmed and unable to process all the data, leading to an IoT data traffic overflow [5]. Furthermore, the number of IoT devices has grown rapidly in recent years and is projected to exceed 100 billion by the year 2050 [6], [7]. Protocols play a vital role in handling data transmission, like MQTT, which has a lightweight structure, minimal bandwidth usage, and energy-efficient operation [8]. MQTT is particularly beneficial in IoT environments where devices operate under low-power constraints and unreliable network conditions. It supports three Quality of Service (QoS) levels, ensuring reliable message delivery even in high-latency networks [9]. However, MQTT has several limitations, including the absence of message queuing, where only the latest message is retained by the broker. Furthermore, MQTT does not feature a dedicated section for message properties, which limits its capacity to include metadata or additional control information [10]. Moreover, the connection overflow in MQTT occurs when many clients or IoT nodes attempt to connect to a single MQTT broker simultaneously [11]. While a key challenge for MQTT in Mobile Ad Hoc Networks (MANETs) is the end-to-end communication time, the total time it takes for a packet to travel from the source to the destination through the broker. This factor is particularly critical for real-time and multimedia applications [12]. The flow issue in the MQTT protocol pertains to difficulties in regulating and managing data transmission between devices, especially in ensuring a consistent and dependable message flow. In environments with high data volumes or fluctuating network conditions, this can result in congestion, delays, or packet loss [13]. Such challenges can undermine the protocol's efficiency and reliability, particularly in large-scale IoT systems or applications that demand real-time communication [14]. Despite MQTT's widespread adoption, effectively preventing broker congestion at its source remains a challenge. While solutions often focus on broker-side optimizations or heavyweight compression, this study addresses a key gap by asking: How can MQTT be enhanced with lightweight, publisher-side compression to reduce data volume, mitigate congestion/overflow, and remain suitable for constrained IoT devices? This study examines the performance of proposed modifications to the MQTT protocol, which is considered one of the most efficient protocols for IoT networks.

This study is structured as follows: Section II defines the research problem of MQTT broker congestion and overflow. Section III reviews related work and state-of-the-art solutions. Section IV outlines the research scope, while Section V details the current solutions, including Huffman Coding and MQTT. Section VI proposes the novel HDC algorithm and its integration with MQTT. Section VII describes the experimental setup. Section VIII presents and analyzes the results against benchmarks. Section IX summarizes the core contributions of this work. Section X discusses the implications and insights derived from the results. Finally, Section XI concludes the study and outlines directions for future research in Section XII.

II. Research Problem

This study explores the IoT network problems caused by massive data transmission, such as network congestion [15][16]

[17] and the connection overflow [18] [19]. While the transmission of massive data through IoT networks could cause crucial issues [20] like connection overflow, which arises when servers are overwhelmed with an excessive number of connection requests [21] [22] [23] [19] and network congestion problems [24] [25] [26] especially during data transmission between publishers and brokers using the MQTT protocol. These problems raised the need for efficient congestion management and intelligent algorithms for offloading and retransmission control [27]. Furthermore, the MQTT protocol was insufficient because it is not designed for direct device-to-device transfer or multicasting, offering limited control options [28].

III. Related Work

Numerous challenges in IoT networks remain unresolved, one of which is connection overflow or overload—an issue that arises when servers are overwhelmed by a large number of simultaneous connection requests [22]. This can severely impact system performance and reliability [21] [19]. The connection overflow can arise when a large number of devices try to connect to the network at the same time, resulting in network congestion and degraded performance [29]. Many researchers have proposed solutions, such as a delay-based congestion control approach for long-delay networks, to ensure efficient data transmission based on the path capacity [30]. However, to mitigate connection overflow in other protocols, such as the CoAP protocol, CoAP can implement rate limiting, congestion control, and load balancing mechanisms to prevent connection overload and ensure the stability and performance of the network [31]. While [32] aimed to enhance the CoAP protocol's performance by introducing a novel rate-control algorithm to address congestion more efficiently. Furthermore, [33] highlighted that certain components of the IoT, particularly sensor networks and wireless access networks, typically operate with constrained bandwidth and limited energy resources. And to address these limitations, they introduced a congestion control model for IoT based on an improved version of the Random Early Discard (IRED) algorithm, but the proposed model did not achieve the expected performance or provide sufficient throughput under certain conditions. However, [27] noted that congestion is a prevalent issue in IoT networks. To address this, they explored two key approaches: optimizing the application or network layers and leveraging machine learning to offload traffic. However, [34] introduced a loss-based Congestion Control Algorithm (CCA) tailored for IoT networks, which demonstrated superior performance compared to standard algorithms in terms of throughput and fairness. Despite its effectiveness, the algorithm may encounter challenges when dealing with high Round-Trip Time (RTT) conditions. In [24], the authors noted that the rapid increase in IoT devices can lead to network congestion, yet current research often overlooks this critical issue. Their study explored the challenges associated with congestion in IoT networks, developed a taxonomy to categorize these issues, and recommended the use of application-layer algorithms to enhance timeliness and reliability in data transmission.

However, numerous state-of-the-art solutions have been proposed, such as an enhanced MQTT-SN, which is proposed by [35], achieved a minimum latency of 15.5 ms, a peak message transfer rate of 620 messages per second, and demonstrated minimal energy consumption. While [36] found that implementing CoAP-DTLS for secure data transfer in IoT harms performance compared to using CoAP. Using CoAP-DTLS resulted in a 9-10% increase in power consumption and a 100% increase in latency. Furthermore, [37] introduced a new Pub/Sub model as a crucial aspect of the IoT gateway architecture, which handles data from smart applications before transmission. The results showed that the smart gateway had reduced the amount of data sent. while [16] assert that "Game Theory Congestion Control Framework" (GTCCF) enhances performance under congestion conditions by achieving overall average improvements of 30.45% in throughput, 39.77% in end-to-end delay, 26.37% in energy consumption, 91.37% in the number of lost packets, and 13.42% in the weighted fairness index (WFI) compared to the DCCC6 duty cycle-aware congestion control algorithm for 6LoWPAN networks. Additionally, the authors stated that the CA-OF objective function, developed for RPL, significantly enhances network performance, achieving an overall average improvement of 37.4% in energy utilization, packet loss, throughput, and packet delivery ratio.

Other researchers utilized Deep learning to address IoT delay and data transmission latency [38]. Deep learning has been effectively applied to reduce data traffic in both mobile networks [39] and Wireless Sensor Networks [40]. Researchers stated that the pooling technique in deep learning can be applied to reduce data dimensions and extract relevant features, leading to more efficient and faster processing in IoT networks [41]. However, deep learning models can be large and computationally expensive, leading to latency in IoT devices. Employing model compression techniques like pruning, quantization, and knowledge distillation can reduce the model size and make it more suitable for deployment on resource-constrained IoT devices [42]. Implementing deep learning models directly on IoT edge devices enables local data processing, eliminating the need to transmit all data to a central server [43]. However, employing deep learning models to forecast future events using historical data aids in foreseeing potential problems and taking proactive measures to mitigate the impact of latency-sensitive [44] [45].

Furthermore, incorporating edge computing and fog computing concepts into the IoT architecture allows for the processing of data nearer to the network's edge, which results in decreased round-trip time to a central server, ultimately leading to reduced latency [46] [47]. Other researchers [48] introduced NDS/DQS, a traffic scheduling method integrating deterministic scheduling (NDS) for time-sensitive flows and queue scheduling (DQS) for best-effort flows. Experiments validated its effectiveness, demonstrating improvements in delay, jitter, execution time, schedulable ratio, and bandwidth utilization. Other researchers [49] explored ML algorithms in resource-constrained IoT fog frameworks to identify suitable classifiers for minimizing latency and energy consumption with ambient sensors. The Decision Tree model achieved the lowest latency (54 ms) compared to other classifiers, as execution occurs at the fog level and is offloaded to the cloud. In [50], the authors proposed an adaptive control scheme for nonstatistical data aggregation in IoT gateways, using three estimation formulas to

optimize aggregation based on time-varying arrival rates. Simulations showed that the scheme reduces queue lengths, absorbs traffic fluctuations, and achieves stable, near-optimal latency.

Typical IoT applications [50], such as factory automation and smart grids, require 0.25–10 ms latencies and 3–20 ms, respectively. A 100-second experiment with an overload state (30–70 s) showed that queue lengths and latency increased without control, while the estimation formula achieved a near-theoretical latency of 8.0 ms. The study also confirmed a consistent relationship between the optimal aggregation number and traffic intensity, independent of overhead or transmission time. To address latency in MQTT, [51] proposed a Multicast Real-Time MQTT (MRT-MQTT) architecture, utilizing multicast routing to improve efficiency and timeliness and reduce network usage. Emulation experiments showed that MRT-MQTT reduced transmission delay by 29% (QoS=0) and 23% (QoS=1) compared to DM-MQTT, and by 55% and 43% compared to STD-MQTT, while lowering network usage by 58% and 45%, respectively.

Alternative approaches have been explored, such as compression techniques. Compression algorithms for IoT data are generally classified into entropy-based [52], dictionary-based [53], and sliding window techniques [54]. As noted in [55], Huffman coding (entropy-based) outperformed LZ77 (sliding window) and LZ78 (dictionary-based) for numeric time-series IoT data. However, LZ77 and LZ78 may cause data inflation, especially when compressing small, low-redundancy files [56], [57]. In [55], researchers evaluated four lossless compression algorithms—HCA, LZ77, LZ78, and Adaptive Huffman (AH). Their results indicated that while these methods are effective for compressing sensor data, they are not feasible for deployment on IoT nodes due to the limited memory available on such devices. Several studies have explored data compression in sensor networks. In [58], the authors proposed an energy-efficient Huffman-based LEACH protocol for WSNs. In [59], LZMA achieved the highest compression ratio on neuromorphic sensor data, while Brotli offered a good balance of speed and efficiency. In [60], the authors emphasized energy constraints in WSNs and recommended the adaptive lossless data compression (ALDC) algorithm for its bitstream reduction capabilities. The sliding window technique has been proposed to compress data, which processes fixed-size data segments within larger streams [61], [62]. Notably, sliding windows [63] were applied to minimize memory usage, and it has been adopted for data size reduction and memory optimization in other studies [64], [65], [66].

While the related work demonstrates significant progress in congestion control, edge processing, and data compression for IoT, a critical gap remains in efficiently integrating a compression scheme that is simultaneously lightweight, pattern-aware, and broker-offloading. Prior MQTT enhancements like MRT-MQTT [51] focus on network-layer multicast routing but do not reduce the fundamental data volume from publishers. Conversely, generic compression algorithms (e.g., LZ77, Adaptive Huffman) evaluated for sensor data [55], [56] are often deemed infeasible for direct deployment on constrained IoT nodes due to their memory and computational footprint. Other IoT compression studies [58], [59], [60] operate at the network or application layer independently of the MQTT protocol, missing the opportunity for tight, publisher-side integration that proactively prevents broker overload.

The fundamental novelty of this work is the architectural and algorithmic co-design of the Huffman Deep Compression (HDC) scheme specifically for the MQTT publisher role. Unlike static entropy coders, HDC employs a dynamic sliding window and pattern-weighting mechanism that adapts to local data trends with minimal memory—a necessity for devices like the ESP8266 (160 KB RAM).

Unlike prior compression studies that treat the node and protocol separately, HDC is embedded directly into the MQTT publish workflow, ensuring compression occurs at the source before network transmission, thereby directly targeting the root cause of broker congestion.

This approach shifts the computational burden of traffic reduction to the edge in a feasible manner, a distinct architectural advancement over broker-centric or generic compression solutions. Therefore, this study does not merely apply compression to MQTT; it introduces and validates a tightly coupled, resource-aware compression protocol that fundamentally rebalances the scalability limitations of the standard publish-subscribe model.

IV. Research Scope

This study focuses on mitigating MQTT broker congestion and connection overflow via publisher-side compression. The scope is defined as follows:

- Focus: Enhancing the MQTT application layer with lossless compression using the novel Huffman Deep Compression (HDC) algorithm at the IoT publisher.
- Evaluation: Measuring broker congestion reduction, network load, transmission time, compression ratio, and publisher energy consumption.
- Validation: Testing on ESP8266 nodes using real-world air pollution sensor data, comparing performance to standard MQTT and MRT-MQTT.
- Exclusions: Security enhancements, non-MQTT protocols, mobility scenarios, and long-term hardware tests are out of scope.

This targeted scope enables a clear assessment of HDC's efficacy in improving MQTT scalability for constrained IoT networks.

V. The Current Solutions

A. The Huffman Coding Algorithm

Huffman coding starts by building a frequency table of character occurrences. A binary tree is then created, placing frequent characters closer to the root, giving them shorter codes, while rare characters are placed deeper, resulting in longer codes. Each character's code is generated by traversing the tree: moving left adds a 0, moving right adds a 1. Fig. 1 illustrates a simple Huffman coding process.

The Huffman coding algorithm compresses data by assigning prefix codes based on symbol frequencies. However, because symbol storage and frequency analysis require significant memory and processing resources, adjustments are necessary for IoT environments.

B. The MQTT Protocol

MQTT is used for communication in IoT networks, as illustrated in Fig. 2. The data flow process in MQTT can be included in the following steps [67]:

- The publisher connects to the MQTT broker, a server.
- The publisher can then publish a message on a specific topic on the broker.
- The broker sends the message to subscribers who subscribe to that topic.
- The subscribers can receive the message and take appropriate action.

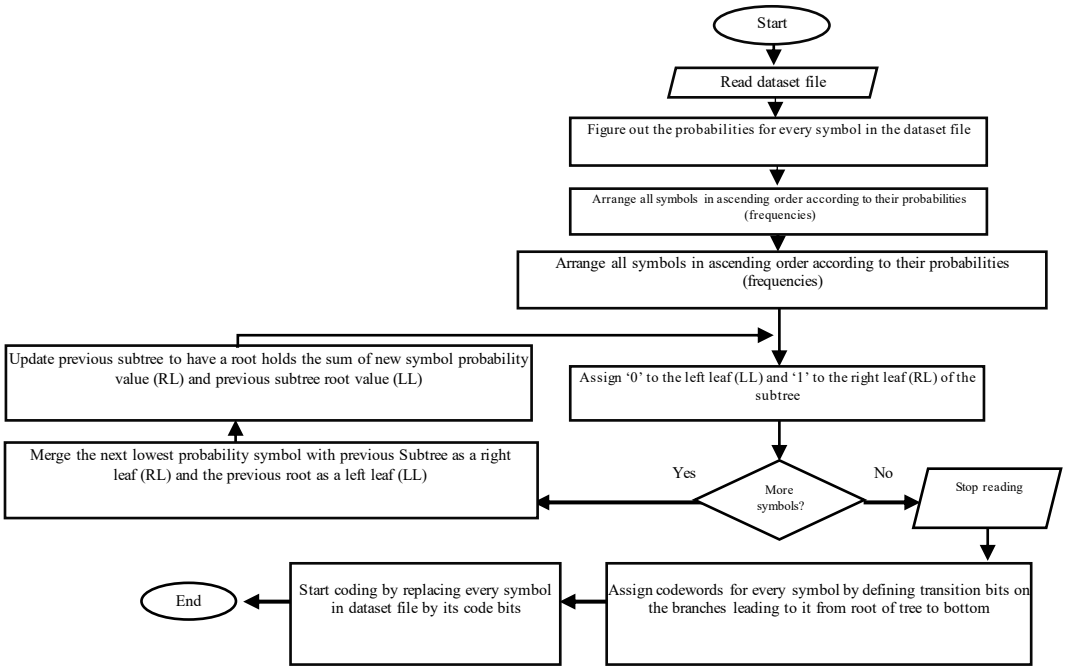

Fig. 1. Huffman coding algorithm (HCA) flowchart.

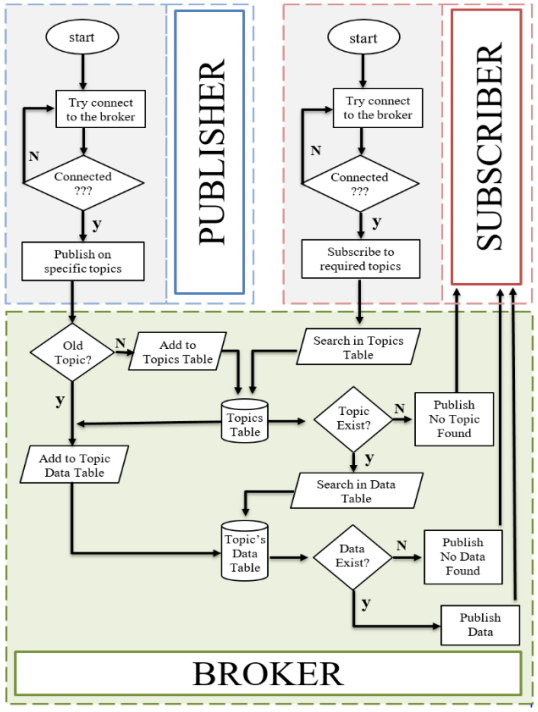

Fig. 2. MQTT publish-subscribe flowchart.

VI. Proposed Solutions

A. Proposed HDC Algorithm

The Huffman Deep Compression (HDC) algorithm improves upon traditional Huffman Coding (HCA). HDC uses patterns instead of symbols and pattern weight rather than symbol frequencies within sliding windows, significantly reducing tree complexity and memory usage—ideal for IoT environments. Each data segment is processed independently using polling and pruning to extract optimal patterns and generate efficient Huffman trees. Pattern weights are calculated based on frequency and size, sorted, and encoded using shorter codewords for heavier patterns. Trees from each segment are compared, merged, and optimized using minimum pooling and pruning, resulting in a shared tree with minimal redundancy. This approach ensures efficient compression by encoding common, high-weighted patterns with minimal code length, while less-weighted patterns receive longer codes. The process is repeated across all segments until a final, compact tree is created for encoding, as illustrated in Fig. 3.

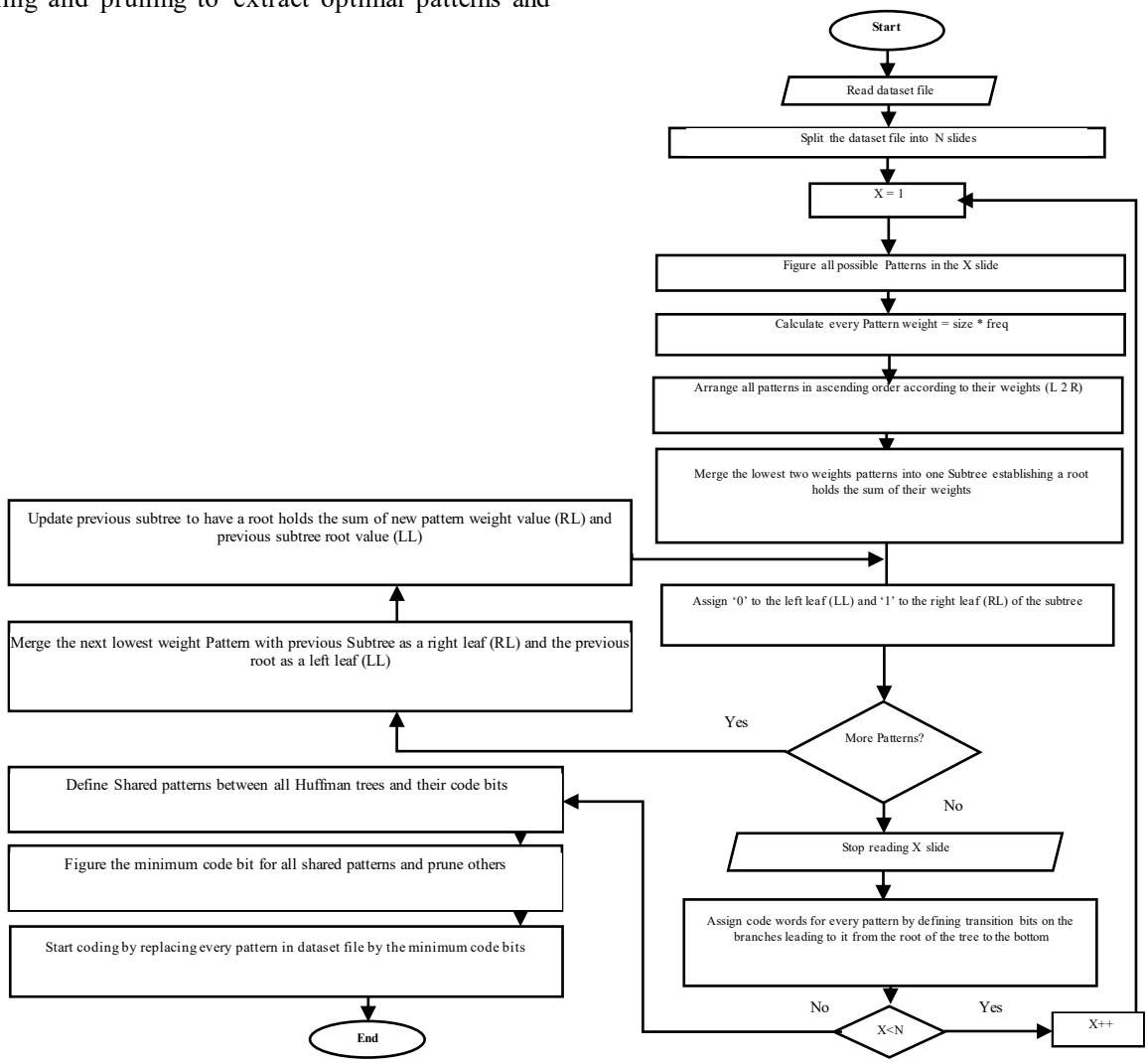

Fig. 3. The HDC algorithm flowchart.

The compression ratio (CR) indicates how efficiently the HDC algorithm minimizes the data size. Eq. (1) shows how to measure the CR, where S_{orig} presents the original data size and S_{comp} presents the compressed data size. A higher CR indicates better compression efficiency.

$$CR = \frac{S_{orig}}{S_{comp}} \quad (1)$$

The HDC algorithm has been previously tested and validated in an earlier published study, demonstrating its effectiveness in reducing memory usage and data traffic on IoT nodes. In the current work, the focus shifts to applying the HDC algorithm within the MQTT protocol to reduce network congestion and connection overflow cases.

B. Proposed MQTT Improvement Based on the HDC Compression Algorithm

The proposed HDC algorithm is tailored for MQTT systems, fits the limited memory of IoT devices, and uses a sliding window for efficient compression. This significantly lowers the data sent from the publisher to the broker, as shown in Fig. 4.

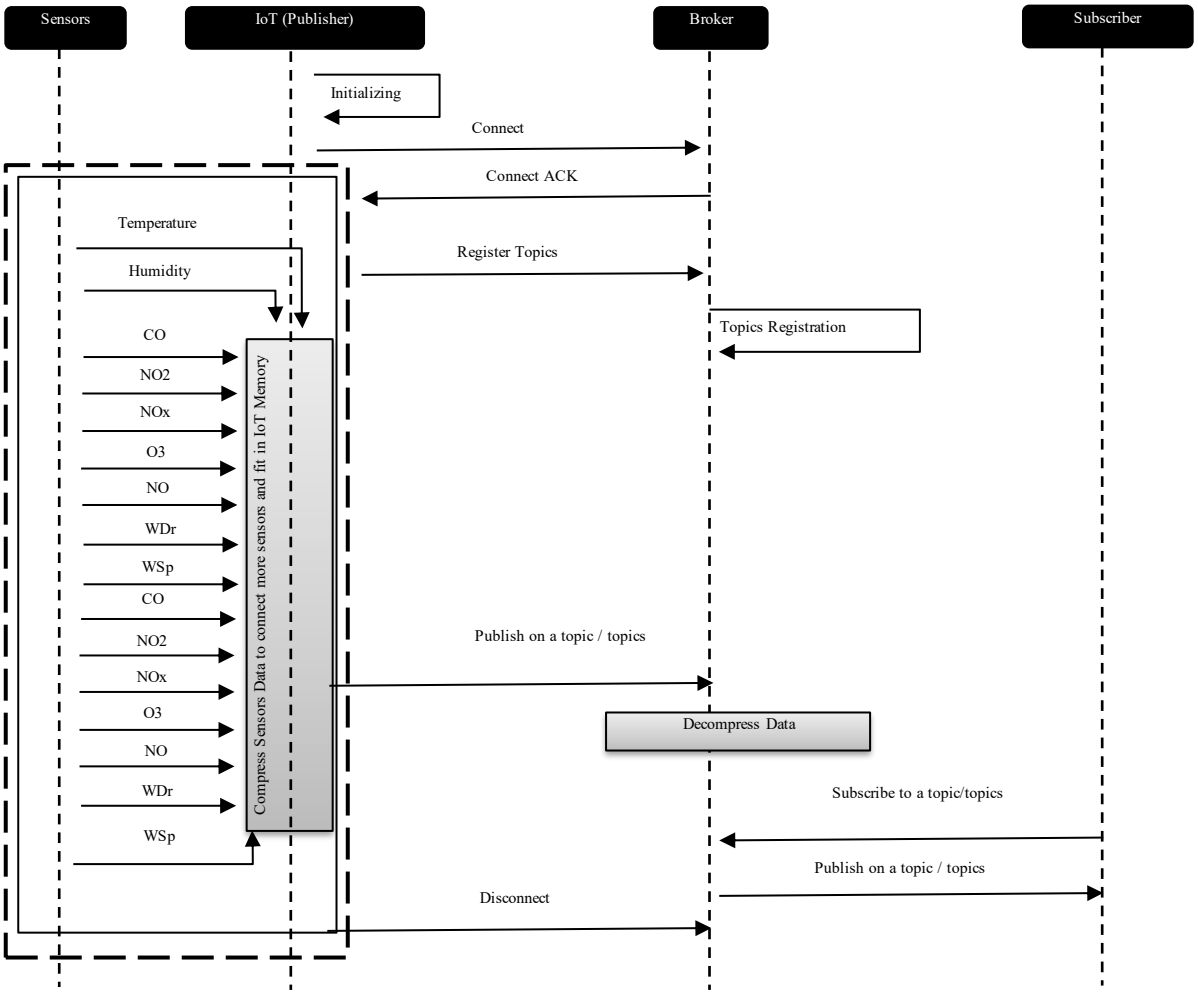

Fig. 4. The proposed MQTT diagram based on the compression approach.

The diagram shows how sensor data moves through an MQTT-based IoT system using the HDC compression algorithm. Sensors first collect data like temperature, humidity, etc. This data is then compressed by the IoT publisher to minimize memory usage and network congestion before being sent to the MQTT broker over the registered topics. Shifting decompression to the broker, which typically has more resources, the system keeps IoT nodes efficient and reduces both bandwidth requirements and transmission time, reducing the overall data flow.

Eq. (2) shows the transmission time reduction ($T_{reduction}$), which presents the difference between data transmission time using HDC and without it.

$$T_{reduction=}\ T_{Orig} - T_{comp} \quad (2)$$

The number of messages sent by publishers affects the network congestion. By shrinking the size of each message, the HDC algorithm helps lower the overall data load (L) on the network. Eq. (3) shows how to measure the network load. Where N is the number of messages sent and S is the message size.

$$L=N\times S \quad (3)$$

With HDC, the reduced load becomes, as in Eq. (4):

$$L_{comp=}\ N * S_{comp} \quad (4)$$

The reduction in network load can be calculated, as in Eq. (5):

$$L_{reduction=}\ L_{orig} - L_{comp} = N * (S_{orig} - S_{comp}) \quad (5)$$

Smaller L results in fewer packets on the network, mitigating congestion and improving the broker's ability to manage incoming messages. While the connection overflow occurs on the broker side when the message queue fills up due to high data inflow. The HDC algorithm helps by decreasing the data size (S_{comp}) of each message, and reducing the arrival rate (R_{comp}) of data packets. Eq. (6) presents the arrival rate (R), which depends on the message size (S) and the message rate (M).

$$R=M\times S \quad (6)$$

With compression, the new arrival rate is presented in Eq. (7):

$$R_{comp=}\ M * S_{comp} \quad (7)$$

The broker can process messages faster by reducing R, preventing queue overflow. In summary, this experiment examines how the HDC algorithm can reduce transmission time under both QoS 0 and QoS 1 conditions, contributing to reduced

network congestion and preventing connection overflow on brokers.

VII. EXPERIMENT

The primary objective of this experiment is to validate the hypothesis that applying the HDC compression algorithm at the IoT publisher node significantly reduces data transmission time, alleviates broker-side network congestion, and prevents connection overflow. The end-to-end process is illustrated in Fig. 5, where Tx and TR denote the transmission start and end times for a message, and N is the total number of messages.

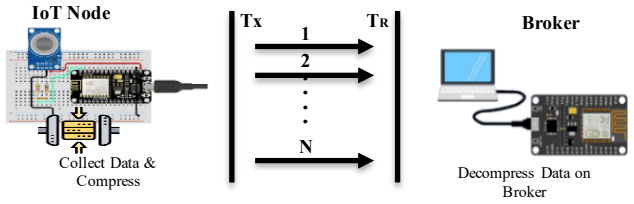

Fig. 5. The experiment process for data transmission from the publisher to the broker using an improved MQTT based on the HDC algorithm.

A. Hardware and Software Configuration

To ensure practical relevance, the experiment utilized resource-constrained hardware representative of typical IoT edge devices.

- IoT Publisher Node: An ESP8266 (NodeMCU v1.0) microcontroller was employed. Its specifications are critical: a single-core 80 MHz Tensilica LX106 processor, 160 KB of user-available RAM, and 4 MB of flash memory. It was connected via 802.11n Wi-Fi to a local network.
- MQTT Broker: The Eclipse Mosquitto™ broker (version 2.0.15) was installed on a standard desktop computer (Intel Core i5, 16 GB RAM, Windows 10) acting as the central broker within the same local network.
- Subscriber and Measurement Host: A C++ script utilizing the Paho-MQTT v1.6.1 client library ran on a separate laptop. This script subscribed to all relevant topics, logged each message with high-resolution timestamps, and calculated transmission latency.
- Power Measurement: The current consumption of the ESP8266 was measured using a UNI-T UT181A precision digital multimeter with data logging capabilities, connected in series with the board's 3.3V power supply.

B. Dataset Characterization and Preparation

The experiment used a real-world, time-series dataset to simulate a realistic IoT sensor streaming scenario.

- Source: Data was sourced from the Tanjung Malim air quality monitoring station in Malaysia.
- Content and Structure: The dataset comprises 4,383 complete records. Each record contains synchronous readings from eight environmental sensors: Temperature (°C), Nitrogen Oxides (NOx, ppm), Relative Humidity (%), Nitrogen Dioxide (NO2), Nitric Oxide (NO), Ozone (O3), Wind Direction (degrees), and Wind Speed (m/s).

C. Experimental Procedure

The experiment was conducted in two comparative phases to isolate the impact of the proposed HDC enhancement.

Phase 1. Baseline Protocol Performance (MQTT vs. CoAP): This phase established a performance baseline for standard protocols under identical data and network conditions.

- Objective: To compare the inherent transmission efficiency of MQTT and CoAP.
- Scenarios:
  - MQTT QoS 0 (at-most-once delivery) vs. CoAP Non-confirmable (NON) messages.
  - MQTT QoS 1 (at-least-once delivery) vs. CoAP Confirmable (CON) messages.
- Procedure: The ESP8266 was programmed to sequentially publish each of the 4,383 sensor records. For MQTT, each sensor type was published to a unique topic (e.g., sensors/temp). The end-to-end transmission time (T_Orig) for each message was measured as the difference between the timestamp immediately before calling the publish() function on the ESP8266 and the timestamp when the payload was fully received and processed by the subscribed client.

Phase 2. Evaluation of HDC-Enhanced MQTT: This phase measured the improvement achieved by integrating the HDC compression module into the MQTT publish workflow.

- Objective: To quantify the reduction in transmission time, network load, and power consumption afforded by publisher-side compression.

Scenarios:

- Standard MQTT (QoS 0 & QoS 1) without compression.
- HDC-enhanced MQTT (QoS 0 & QoS 1) with compression applied before publishing.

Procedure:

- Compression: On the ESP8266, the data for each sensor record was passed through the HDC algorithm prior to the MQTT publish call, producing a compressed payload (S_comp).

  Transmission: The compressed payload was published to the same MQTT topics as in Phase 1.
- Decompression: To maintain standard MQTT semantics for subscribers, a lightweight decompression plugin was added to the Mosquitto broker. This plugin intercepted messages on the designated topics, decompressed the payload using HDC to regenerate the original data.

- Measurement: The transmission time with compression (T_comp) was measured identically to Phase 1. Additionally, the total current consumption for the entire transmission cycle (compression, transmission, and Decompression) was logged by the multimeter for both compressed and uncompressed scenarios.

VIII. Results

The experimental results are divided into five sections.

A. The Performance of HDC

Fig. 6 presents the compression ratio results, comparing HDC to state-of-the-art solutions, showing that HDC achieves the highest compression ratio (2.62), outperforming BDC (2.47) [68], Delta+RLE (1.898) [69], Sahu & Panda (0.89) [70], and LZW+X.509 (0.71) [71], confirming HDC's superior efficiency.

B. MQTT vs. CoAP

Table I displays the transmission time recorded, measured in milliseconds, for the Tanjung Malim station sensors' data, for both scenarios: QoS 0 and QoS 1 in MQTT, and non-confirmable and confirmable modes in CoAP.

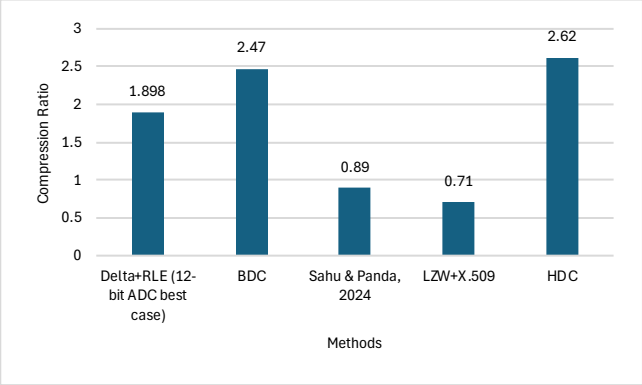

Fig. 6. HDC vs. State-of-the-art research results on compression ratio.

TABLE I. Transmission Time (MS) Comparison Between MQTT and CoAP

	Non-confirmable	QoS 0	Confirmable	QoS 1
Sensor	**CoAP**	**MQTT**	**CoAP**	**MQTT**
Temp	284	442.461	374	460.857
NOx	244	227.448	339	335.711
Humidity	247	214.537	350	335.817
NO2	236	216.483	358	333.08
O3	242	217.965	328	331.201
NO	259	218.061	365	332.993
WDr	215	215.521	348	334.654
WSp	244	215	351	330.429

CoAP shows higher transmission times than MQTT, with the exception of temperature data, where the overhead of TCP connection establishment in MQTT results in longer delays. Despite this, MQTT generally demonstrates superior performance in terms of data transmission efficiency.

C. MQTT with HDC vs. Standard MQTT

Transmitting data from the sensors to the Tanjung Malim station (broker), where every sensor is considered a publisher.

TABLE II. Transmission Time (MS) Comparison Between Standard MQTT and the Improved MQTT Using Compression (HDC)

	QoS 0		QoS 1	
Sensor	**Time with compression**	**Time without compression**	**Time with compression**	**Time without compression**
Temp	43.5602	442.461	77.5269	460.857
NOx	39.878	227.448	72.8804	335.711
Humidity	38.2013	214.537	73.1911	335.817
NO2	39.5145	216.483	71.0191	333.08
O3	40.305	217.965	74.5532	331.201
NO	39.5492	218.061	73.6148	332.993
WDr	27.0295	215.521	56.7491	334.654
WSp	41.7017	215	71.0177	330.429

The experimental results demonstrate that applying compression significantly improves transmission performance in both QoS 0 and QoS 1 settings. The impact of HDC compression on transmission latency is detailed in Table II. Under QoS 0, compression reduced the total transmission time from 1967.48 ms to 309.74 ms, achieving a Network Congestion Reduction (NCR) and Broker Overflow Prevention of 84.26%. Similarly, in QoS 1, compression lowered the total transmission time from 2794.74 ms to 570.55 ms, with an NCR and Broker Overflow Prevention of 79.6%. These findings indicate that the compression approach effectively minimizes transmission delays, reduces network congestion, and alleviates broker load across different QoS levels.

D. Compression Time

Table III presents the recorded times used to calculate the total data transmission duration from the publisher to the broker, incorporating both compression at the publisher and decompression at the broker, under the two transmission scenarios: QoS 0 and QoS 1.

TABLE III. Time (MS) for Data Transmission From the Publisher to the Broker Using the HDC Algorithm

	Compression Time	Transmission Time	Decompression Time
QoS 0	1458.589	309.7394	63.6218
QoS 1	1458.589	570.5523	63.6218

Data transmission using the HDC algorithm shows different performance for each QoS level. QoS 0 achieves a total time of ~1831.95 ms, while QoS 1 increases to ~2092.76 ms due to longer transmission time. Since compression and decompression times remain unchanged, the additional delay in QoS 1 is due to its increased reliability mechanism.

E. Power Consumption

The IoT node draws 26 mA at idle. Compression with HDC increases consumption to 44 mA (an extra 18 mA), while decompression requires 40 mA. Transmission consumes 102 mA with QoS 0 and 116 mA with QoS 1. The table below summarizes these values.

TABLE IV. CURRENT CONSUMPTION FOR EACH PROCESS IN MAMPS

Operation	Consumption (mAmps)
Compress (per sliding window)	18
Decompress (per sliding window)	14
QoS 0 Transmission	76
QoS 1 Transmission	90

The table shows current consumption for each operation, as measured and listed in Table IV. Compression and decompression require 18 mA and 14 mA per sliding window, respectively, while data transmission draws 76 mA for QoS 0 and 90 mA for QoS 1 due to extra acknowledgments. Table V shows how many mAmps in total were needed to transmit data from the publisher to the broker.

TABLE V. POWER CONSUMPTION (MAMPS) FOR THE DATA TRANSMISSION FROM THE PUBLISHER TO THE BROKER

	With Compression	Without Compression
QoS 0	63216	2664864
QoS 1	49168	3155760

The table indicates that using compression significantly reduces the total current consumption during data transmission for both QoS levels. The results demonstrate that compression not only reduces data volume but also leads to substantial energy savings, especially beneficial for battery-powered IoT nodes.

IX. CONTRIBUTIONS

This work makes the following distinct contributions to the field of IoT communication protocols:

- A Novel Protocol Integration Architecture: A new publisher-side compression framework for MQTT is proposed and validated, shifting the primary computational burden of traffic reduction from the broker to the edge. This architectural shift establishes a design principle: proactive, source-side data minimization is a scalable alternative to reactive broker-side optimizations for congestion control.
- An Algorithm Optimized for Constrained MQTT Publishers: A resource-aware adaptation of the Huffman Deep Compression (HDC) algorithm is introduced, specifically tailored for the MQTT publisher role on devices with severe memory constraints (e.g., 160 KB RAM). The key innovations are its dynamic sliding-window pattern matching and lightweight tree-generation process, which maintain a high compression ratio of 2.62 while operating within the limits of IoT edge hardware.
- Quantified Performance Advancement and New Congestion Insights: Through empirical validation, this study demonstrates that the integrated system reduces broker-side congestion by 84.26% for QoS 0 and 79.6% for QoS 1, significantly outperforming the state-of-the-art MRT-MQTT (58% and 45%, respectively). These results provide new insights into congestion dynamics, quantitatively proving that reducing per-message payload size is disproportionately effective in alleviating queue growth—more so than optimizing broker processing speed alone.
- Substantial Energy Efficiency Gains and a Reusable Benchmark: The approach delivers transformative energy savings, reducing total current consumption from 2,664,864 to 63,216 mA (QoS 0) and from 3,155,760 to 49,168 mA (QoS 1). Furthermore, a complete, reproducible methodology for evaluating compression-enhanced MQTT is provided, establishing a new performance benchmark for the community.

These contributions collectively advance the field by providing not just a faster protocol, but a new architectural pattern, an optimized algorithm, fundamental insights into congestion mechanics, dramatic energy improvements, and a reproducible benchmark for scalable IoT communication systems.

X. DISCUSSION

The results validate the HDC-enhanced MQTT protocol. The core insight is that broker congestion is fundamentally a data volume issue. Solutions like MRT-MQTT manage traffic more efficiently at the broker, but our publisher-side compression prevents the overload at its source. The higher congestion reduction for QoS 0 (84.26% vs. 79.6% for QoS 1) confirms this, as the acknowledgment overhead in QoS 1 presents a fixed bottleneck.

The dramatic energy reduction—from millions to tens of thousands of mA—stems from shorter radio transmission times for compressed data. This demonstrates a key principle: optimizing network efficiency in IoT inherently optimizes energy efficiency, directly extending device lifetime in large-scale deployments.

Achieving a compression ratio of 2.62 on a device with 160 KB RAM also addresses prior work that deemed such algorithms infeasible on constrained nodes. This work establishes a practical design pattern: the Intelligent Publisher, where edge nodes preprocess data to minimize their network footprint. This shifts system scalability from being a broker-centric challenge to a shared responsibility, offering a clear path for building sustainable, massive IoT networks. The main trade-off is added compression latency, which the significantly reduced transmission time offsets for most applications.

XI. CONCLUSION

This study has demonstrated that integrating Huffman Deep Compression (HDC) at the MQTT publisher is a fundamental strategy for scalable IoT. The significant performance gains—84.26% congestion reduction, a 2.62 compression ratio, and over 97% energy savings—are not merely numerical

improvements, but represent critical advancements for real-world systems.

The reduction in broker congestion directly translates to enhanced system reliability and stability; brokers can handle orders of magnitude more devices without queue overflow, enabling true massive-scale deployments. The drastic drop in energy consumption extends device lifetime from months to years, reducing maintenance costs and enabling deployment in inaccessible locations. Finally, by minimizing the data load, this approach reduces bandwidth requirements and operational costs for large sensor networks.

While validated on environmental data, the proposed architectural principle, proactive source-side minimization, provides a generalizable framework for enhancing publish-subscribe protocols. Future work will test this framework with diverse data and in dynamic networks. Ultimately, this work moves beyond optimizing a protocol to providing a practical blueprint for building sustainable, large-scale IoT infrastructure. A noted limitation is the focus on numerical time-series data within a stable local network. Future work will therefore investigate HDC's performance on diverse data formats, its integration with security layers like MQTT-S, and its robustness in dynamic, large-scale network topologies to further generalize its utility for next-generation IoT ecosystems.

XII. Future Work

Future research on the Huffman Deep Compression (HDC) algorithm should focus on enhancing its performance in large-scale IoT environments. Integrating loss compression techniques may significantly reduce data size while preserving essential information, leading to lower transmission latency. Maximizing throughput and ensuring scalability in high-density IoT networks, such as smart city infrastructures, is another critical direction. Furthermore, implementing the enhanced MQTT protocol within next-generation IoT architectures, including 6G networks, can unlock new opportunities for ultra-reliable and low-latency communication. Finally, developing innovative strategies to reduce network overhead will be vital for sustaining efficient and reliable data transmission in massive IoT ecosystems.

Acknowledgment

This work is part of a research project funded by the FRGS/1/2019/ICT02/UKM/02/7, the Ministry of Higher Education of Malaysia.

References

[1] M. Alipio and M. Bures, "Deep Reinforcement Learning Perspectives on Improving Reliable Transmissions in IoT Networks: Problem Formulation, Parameter Choices, Challenges, and Future Directions," Internet of Things, vol. 23, no. May, p. 100846, 2023, doi: 10.1016/j.iot.2023.100846.

[2] R. L. Delfanti et al., "EPIIC: A NOVEL ENCODING PLUGGABLE LOSSLESS DATA COMPRESSION ALGORITHM A," 2018. doi: 10.1056/nejmoa1407279.

[3] M. Caporuscio, F. Edrisi, M. Hallberg, A. Johannesson, C. Kopf, and D. Perez-Palacin, Architectural concerns for digital twin of the organization, vol. 12292 LNCS. 2020. doi: 10.1007/978-3-030-58923-3_18.

[4] A. N. Jaber Al, "Efficient Visualization Framework for Real-Time Monitoring Network Traffic of High-Speed Networks," Proc. - 2021 IEEE Int. Conf. Big Data, Big Data 2021, pp. 5839–5842, 2021, doi: 10.1109/BigData52589.2021.9671915.

[5] P. M. E. Ramakrishnan, S. Hari Prasath, L. Harish, and E. KrishnaKumar, "IoT based Industrial Automation for Various Load using ATmega328p Microcontroller," Proc. - 2022 6th Int. Conf. Intell. Comput. Control Syst. ICICCS 2022, no. Icicccs, pp. 471–475, 2022, doi: 10.1109/ICICCS53718.2022.9788258.

[6] G. G. K. W. M. S. I. R. Karunarathne, K. A. D. T. Kulawansa, and M. F. M. Firdhous, "Wireless communication technologies in internet of things: A critical evaluation," 2018 Int. Conf. Intell. Innov. Comput. Appl. ICONIC 2018, no. June, 2019, doi: 10.1109/ICONIC.2018.8601226.

[7] A. Almansoori, C. Ncube, and S. A. Salloum, "Internet of Things Impact on the Future of Cyber Crime in 2050," Res. Gate, no. May, pp. 643–655, 2021, doi: 10.1007/978-3-030-76346-6_57.

[8] P. Chhikara, R. Tekchandani, N. Kumar, and M. S. Obaidat, "An Efficient Container Management Scheme for Resource-Constrained Intelligent IoT Devices," IEEE Internet Things J., vol. 8, no. 16, pp. 12597–12609, 2021, doi: 10.1109/JIOT.2020.3037181.

[9] R. Doshi, S. Inamdar, T. Karmarkar, and M. Wakode, "Distributed MQTT Broker: A Load-Balanced Redis-Based Architecture," 2024 Int. Conf. Emerg. Smart Comput. Informatics, ESCI 2024, pp. 1–6, 2024, doi: 10.1109/ESCI59607.2024.10497427.

[10] S. Gruener, H. Koziolek, and J. Ruckert, "Towards Resilient IoT Messaging: An Experience Report Analyzing MQTT Brokers," Proc. - IEEE 18th Int. Conf. Softw. Archit. ICSA 2021, pp. 69–79, 2021, doi: 10.1109/ICSA51549.2021.00015.

[11] C. C. Kao, C. W. Chang, C. P. Cho, and J. Y. Shun, "Deep Learning and Ensemble Learning for Traffic Load Prediction in Real Network," 2nd IEEE Eurasia Conf. IOT, Commun. Eng. 2020, ECICE 2020, pp. 36–39, 2020, doi: 10.1109/ECICE50847.2020.9302005.

[12] A. S. Sadeq, "A QOS APPROACH FOR INTERNET OFTHINGS (IOT) ENVIRONMENT USING MQTT PROTOCOL," in 2019 International Conference on Cybersecurity (ICoCSec), IEEE, 2019, pp. 59–63. doi: https://doi.org/10.1109/ICoCSec47621.2019.8971097.

[13] S. Lakshminarayana and P. S. Thilagam, "Next-Generation DDoS Attacks on IoT Deployments: Targeting the Advanced Features of MQTT v5.0 Protocol," IEEE Internet Things J., vol. PP, p. 1, 2025, doi: 10.1109/JIOT.2025.3549784.

[14] J. Ahn, J. Y. Park, D. Park, J. Paek, and J. G. Ko, "Convolutional neural network-based classification system design with compressed wireless sensor network images," PLoS One, vol. 13, no. 5, 2018, doi: 10.1371/journal.pone.0196251.

[15] P. Chanak and I. Banerjee, "Congestion Free Routing Mechanism for IoT-Enabled Wireless Sensor Networks for Smart Healthcare Applications," IEEE Trans. Consum. Electron., vol. 66, no. 3, pp. 223–232, 2020, doi: 10.1109/TCE.2020.2987433.

[16] V. K. Jain, A. P. Mazumdar, P. Faruki, and M. C. Govil, "Congestion control in Internet of Things: Classification, challenges, and future directions," Sustain. Comput. Informatics Syst., vol. 35, no. April 2020, p. 100678, 2022, doi: 10.1016/j.suscom.2022.100678.

[17] L. P. Verma and M. Kumar, "An IoT based Congestion Control Algorithm," Internet of Things (Netherlands), vol. 9, p. 100157, 2020, doi: 10.1016/j.iot.2019.100157.

[18] H. Akram, S. G. Abbas, and G. A. Shah, "Preventing MQTT Vulnerabilities Using IoT-Enabled Intrusion Detection System," Sensors (Switzerland), vol. 22(2), no. january 2022, p. 27, 2022, doi: https://doi.org/10.3390/s22020567.

[19] U. Pandey and B. K. Chaurasia, "IoT edge based framework for sybil and buffer overrow detection IoT edge based framework for sybil and buffer overflow detection," Res. Sq., no. June 2nd, 2022, pp. 0–16, 2022, [Online]. Available: https://doi.org/10.21203/rs.3.rs-1693583/v1

[20] R. Krishnamurthi, A. Kumar, D. Gopinathan, A. Nayyar, and B. Qureshi, "An overview of iot sensor data processing, fusion, and analysis techniques," Sensors (Switzerland), vol. 20, no. 21, pp. 1–23, 2020, doi: 10.3390/s20216076.

[21] M. S. Harsha, B. M. Bhavani, and K. R. Kundhavai, "Analysis of vulnerabilities in MQTT security using Shodan API and implementation of its countermeasures via authentication and ACLs," 2018 Int. Conf.

Adv. Comput. Commun. Informatics, ICACCI 2018, pp. 2244–2250, 2018, doi: 10.1109/ICACCI.2018.8554472.

[22] O. Jukic, R. Filjar, I. Hedi, and E. Cirikovic, "MQTT-like Network Management Architecture," 2021 44th Int. Conv. Information, Commun. Electron. Technol. MIPRO 2021 - Proc., pp. 459–463, 2021, doi: 10.23919/MIPRO52101.2021.9596894.

[23] P. Kortoçi, L. Zheng, C. Joe-Wong, M. Di Francesco, and M. Chiang, "Fog-based Data Offloading in Urban IoT Scenarios," Proc. - IEEE INFOCOM, vol. 2019-April, pp. 784–792, 2019, doi: 10.1109/INFOCOM.2019.8737503.

[24] A. Gerodimos, L. Maglaras, M. A. Ferrag, N. Ayres, and I. Kantzavelou, "IoT: Communication protocols and security threats," Internet Things Cyber-Physical Syst., vol. 3, no. January, pp. 1–13, 2023, doi: 10.1016/j.iotcps.2022.12.003.

[25] H. A. Khalek and L. Mhamdi, "Light-Weight Congestion Control in Constrained IoT Networks," Proc. - IEEE Glob. Commun. Conf. GLOBECOM, pp. 6265–6270, 2022, doi: 10.1109/GLOBECOM48099.2022.10000648.

[26] R. Sukjaimuk, Q. N. Nguyen, and T. Sato, "An Efficient Congestion Control Model utilizing IoT wireless sensors in Information-Centric Networks," 2021 Jt. 6th Int. Conf. Digit. Arts, Media Technol. with 4th ECTI North. Sect. Conf. Electr. Electron. Comput. Telecommun. Eng. ECTI DAMT NCON 2021, pp. 210–213, 2021, doi: 10.1109/ECTIDAMTNCON51128.2021.9425753.

[27] A. Maheshwari and R. K. Yadav, "Analysis of congestion control mechanism for IOT," Proc. Conflu. 2020 - 10th Int. Conf. Cloud Comput. Data Sci. Eng., pp. 288–293, 2020, doi: 10.1109/Confluence47617.2020.9058058.

[28] K. Phung, H. Tran, and V. Tran-quang, "Service Platform for Integration of various M2M / IoT system," vol. 144, pp. 17–21, 2020.

[29] C. Suwannapong and C. Khunboa, "Congestion control in CoAP observe group communication," Sensors (Switzerland), vol. 19, no. 15, pp. 1–14, 2019, doi: 10.3390/s19153433.

[30] L. Verma, I. Verma, and M. Kumar, "An Adaptive Congestion Control Algorithm," Model. Meas. Control A, vol. 92, no. 1, pp. 30–36, 2019, doi: 10.18280/mmc_a.920105.

[31] V. K. Jain, A. P. Mazumdar, and M. C. Govil, "Toward Adaptive Range for Parallel Connections in CoAP," Arab. J. Sci. Eng., vol. 46, no. 4, pp. 3595–3611, 2021, doi: 10.1007/s13369-020-05215-w.

[32] Y. Cui and D. Lei, "Optimizing Internet of Things-Based Intelligent Transportation System's Information Acquisition Using Deep Learning," IEEE Access, vol. 11, no. January, pp. 11804–11810, 2023, doi: 10.1109/ACCESS.2023.3242116.

[33] J. Huang et al., "Modeling and analysis on congestion control in the Internet of Things," 2014 IEEE Int. Conf. Commun. ICC 2014, pp. 434–439, 2014, doi: 10.1109/ICC.2014.6883357.

[34] H. H. Hasan and Z. T. Alisa, "Effective IoT Congestion Control Algorithm," Futur. Internet, vol. 15, no. 4, 2023, doi: 10.3390/fi15040136.

[35] E. Shahri, P. Pedreiras, and L. Almeida, "Enhancing MQTT with Real-Time and Reliable Communication Services," IEEE Int. Conf. Ind. Informatics, vol. 2021-July, pp. 1–6, 2021, doi: 10.1109/INDIN45523.2021.9557514.

[36] F. Siddiqui, J. Beley, S. Zeadally, and G. Braught, "Secure and lightweight communication in heterogeneous IoT environments," Internet of Things (Netherlands), vol. 14, no. 1, p. 100093, 2021, doi: 10.1016/j.iot.2019.100093.

[37] H. R. A. Ameer and H. M. Hasan, "Enhanced MQTT Protocol by Smart Gateway," Iraqi J. Comput. Commun. Control Syst. Eng., vol. 1, no. October, pp. 53–67, 2020, doi: 10.33103/uot.ijccce.20.1.6.

[38] M. Ateeq, F. Ishmanov, M. K. Afzal, and M. Naeem, "Predicting Delay in IoT Using Deep Learning: A Multiparametric Approach," IEEE Access, vol. 7, pp. 62022–62031, 2019, doi: 10.1109/ACCESS.2019.2915958.

[39] S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," in Published as a conference paper at ICLR 2016 DEEP, 2016, pp. 1–14. doi: abs/1510.00149/1510.00149.

[40] S. Teotia and V. Sharma, "A Review on Deep Learning Models for Wireless Sensor Networks," vol. 8, no. 4, pp. 268–273, 2019.

[41] L. Huan, B. Tang, and C. Zhao, "Global Composite Compression of Deep Neural Network in Wireless Sensor Networks for Edge Intelligent Fault Diagnosis," IEEE Sens. J., vol. PP, no. Xx, p. 1, 2023, doi: 10.1109/JSEN.2023.3290153.

[42] X. Xiao, Z. Wang, and S. Rajasekaran, "AutoPrune: Automatic network pruning by regularizing auxiliary parameters," Adv. Neural Inf. Process. Syst., vol. 32, no. NeurIPS, 2019.

[43] S. Jha and D. Tripathy, "Low Latency Consistency based Protocol for Fog Computing Systems using CoAP with Machine Learning," 2023 2nd Int. Conf. Innov. Technol. INOCON 2023, pp. 1–6, 2023, doi: 10.1109/INOCON57975.2023.10101176.

[44] I. Kok and S. Ozdemir, "DeepMDP: A Novel Deep-Learning-Based Missing Data Prediction Protocol for IoT," IEEE Internet Things J., vol. 8, no. 1, pp. 232–243, 2021, doi: 10.1109/JIOT.2020.3003922.

[45] Y. Sharma*, V. Tyagi, and P. Datta, "IOT Based Smart Agriculture Monitoring System," Int. J. Innov. Technol. Explor. Eng., vol. 9, no. 9, pp. 325–328, 2020, doi: 10.35940/ijitee.i7142.079920.

[46] J. Liu, K. Luo, Z. Zhou, and X. Chen, "ERP: Edge resource pooling for data stream mobile computing," IEEE Internet Things J., vol. 6, no. 3, pp. 4355–4368, 2019, doi: 10.1109/JIOT.2018.2882588.

[47] I. Kok, B. H. Corak, U. Yavanoglu, and S. Ozdemir, "Deep Learning based Delay and Bandwidth Efficient Data Transmission in IoT," Proc. - 2019 IEEE Int. Conf. Big Data, Big Data 2019, pp. 2327–2333, 2019, doi: 10.1109/BigData47090.2019.9005680.

[48] Y. Lu et al., "An Intelligent Deterministic Scheduling Method for Ultralow Latency Communication in Edge Enabled Industrial Internet of Things," IEEE Trans. Ind. Informatics, vol. 19, no. 2, pp. 1756–1767, 2023, doi: 10.1109/TII.2022.3186891.

[49] N. K. Suryadevara, "Energy and latency reductions at the fog gateway using a machine learning classifier," Sustain. Comput. Informatics Syst., vol. 31, no. August 2020, p. 100582, 2021, doi: 10.1016/j.suscom.2021.100582.

[50] H. Yoshino, K. Ota, and T. Hiraguri, "Adaptive Control of Nonstatistical Sensor Data Aggregation to Minimize Latency in IoT Gateways," 2019 29th Int. Telecommun. Networks Appl. Conf. ITNAC 2019, pp. 0–5, 2019, doi: 10.1109/ITNAC46935.2019.9077988.

[51] E. Shahri, P. Pedreiras, L. Almeida, and J. Sousa, "Scalable SDN-based MQTT Real-Time Communications for Edge Networks," IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA, vol. 2023-Septe, pp. 1–8, 2023, doi: 10.1109/ETFA54631.2023.10275671.

[52] R. A. Venkat and C. Vaidyanathan, "Lossless Video Compression Using Bayesian Networks and Entropy Coding," Proc. 2019 IEEE Reg. 10 Symp. TENSYMP 2019, vol. 7, pp. 254–259, 2019, doi: 10.1109/TENSYMP46218.2019.8971209.

[53] S. Kanda, K. Morita, and M. Fuketa, "Practical String Dictionary Compression Using String Dictionary Encoding," Proc. - 2017 Int. Conf. Big Data Innov. Appl. Innov. 2017, vol. 2018-Janua, pp. 1–8, 2018, doi: 10.1109/Innovate-Data.2017.9.

[54] W. Feng, H. Luo, B. Sun, and C. Gui, "Performance analysis of sliding window network coding for energy efficient in MANETs," Proc. 2017 IEEE 7th Int. Conf. Electron. Inf. Emerg. Commun. ICEIEC 2017, pp. 219–222, 2017, doi: 10.1109/ICEIEC.2017.8076548.

[55] A. Nasif, Z. A. Othman, and N. S. Sani, "The deep learning solutions on lossless compression methods for alleviating data load on iot nodes in smart cities," Sensors, vol. 21, no. 12, 2021, doi: 10.3390/s21124223.

[56] V. Kulkarni, A. Mahalunkar, B. Garbinato, and J. D. Kelleher, "Examining the limits of predictability of human mobility," Entropy, vol. 21, no. 4, Apr. 2019, doi: 10.3390/e21040432.

[57] A. Y. Tuama, M. A. Mohamed, A. Muhammed, and Z. M. Hanapi, "A new compression algorithm for small data communication in wireless sensor network," Int. J. Sens. Networks, vol. 25, no. 3, pp. 163–175, 2017, doi: 10.1504/IJSNET.2017.087712.

[58] D. Mechta and S. Harous, "HC-LEACH: Huffman Coding-based energy-efficient LEACH protocol for WSN," 2020 11th IEEE Annu. Ubiquitous Comput. Electron. Mob. Commun. Conf. UEMCON 2020, pp. 0932–0938, 2020, doi: 10.1109/UEMCON51285.2020.9298061.

[59] K. Iqbal, N. Khan, and M. G. Martini, "Performance Comparison of Lossless Compression Strategies for Dynamic Vision Sensor Data," ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc., vol. 2020-May, pp. 4427–4431, 2020, doi: 10.1109/ICASSP40776.2020.9053178.

[60] K. L. Ketshabetswe, A. M. Zungeru, B. Mtengi, C. K. Lebekwe, and S. R. S. Prabaharan, "Data Compression Algorithms for Wireless Sensor Networks: A Review and Comparison," IEEE Access, vol. 9, pp. 136872–136891, 2021, doi: 10.1109/ACCESS.2021.3116311.

[61] S. Kavita and G. Dakshayani, "A Sliding Window Blockchain Architecture for the Internet of Things," in 5th IEEE International Conference on Advances in Science and Technology, ICAST 2022, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 45–48. doi: 10.1109/ICAST55766.2022.10039664.

[62] C. Y. Wu, "Improved LZ77 Compression," Data Compression Conf. Proc., vol. 2021-March, no. Dcc, p. 377, 2021, doi: 10.1109/DCC50243.2021.00066.

[63] P. J. Zhao, G. B. Hu, and L. W. Wang, "A Sliding Window Data Compression Method for Spatial-Time DOA Estimation," Int. J. Antennas Propag., vol. 2021, 2021, doi: 10.1155/2021/9705617.

[64] A. M. Ghosh and K. Grolinger, "Edge-Cloud Computing for Internet of Things Data Analytics: Embedding Intelligence in the Edge with Deep Learning," IEEE Trans. Ind. Informatics, vol. 17, no. 3, pp. 2191–2200, 2021, doi: 10.1109/TII.2020.3008711.

[65] W. Liu and J. OuYang, "Clustering algorithm for high dimensional data stream over sliding windows," Proc. 10th IEEE Int. Conf. Trust. Secur. Priv. Comput. Commun. Trust. 2011, 8th IEEE Int. Conf. Embed. Softw. Syst. ICESS 2011, 6th Int. Conf. FCST 2011, pp. 1537–1542, 2011, doi: 10.1109/TrustCom.2011.213.

[66] M. Tareq, E. A. Sundararajan, M. Mohd, and N. S. Sani, "Online clustering of evolving data streams using a density grid-based method," IEEE Access, vol. 8, pp. 166472–166490, 2020, doi: 10.1109/ACCESS.2020.3021684.

[67] D. Borsatti, W. Cerroni, F. Tonini, and C. Raffaelli, "From IoT to cloud: Applications and performance of the MQTT protocol," Int. Conf. Transparent Opt. Networks, vol. 2020-July, pp. 20–23, 2020, doi: 10.1109/ICTON51198.2020.9203167.

[68] S. H. Hwang, K. M. Kim, S. Kim, and J. W. Kwak, "Lossless Data Compression for Time-Series Sensor Data Based on Dynamic Bit Packing," Sensors (Basel)., vol. 23, no. 20, Oct. 2023, doi: 10.3390/s23208575.

[69] A. Hanumanthaiah, A. Gopinath, C. Arun, B. Hariharan, and R. Murugan, "Comparison of Lossless Data Compression Techniques in Low-Cost Low-Power (LCLP) IoT Systems," Proc. 2019 Int. Symp. Embed. Comput. Syst. Des. ISED 2019, pp. 63–67, 2019, doi: 10.1109/ISED48680.2019.9096229.

[70] M. Sahu and J. Panda, "A Proposed IOT-based Smart Healthcare Management Framework for Performing Lossless Data Compression using Concept of Ontology," J. Sci. Ind. Res. (India)., vol. 83, no. 3, pp. 282–291, Mar. 2024, doi: 10.56042/jsir.v83i3.6153.

[71] S. Karthikeyan and T. Poongodi, "Secured Data Compression and Data Authentication in Internet of Thing Networks Using LZW Compression Based X.509 Certification," in IEEE International Conference on Data Science and Information System, ICDSIS 2022, Institute of Electrical and Electronics Engineers Inc., 2022. doi: 10.1109/ICDSIS55133.2022.9915855.

AUTHORS' PROFILE

AMMAR SAID NASIF earned his bachelor's degree in Computer Information Systems from Alquds Open University, Palestine, in 2003, and a Master's degree from Arab Academy for Banking & Financial Sciences, Jordan, in 2006. Currently pursuing a PhD at UKM University, Malaysia, his research focuses on IoT networks, particularly optimizing the MQTT protocol. Ammar is an experienced Applications Developer proficient in .NET platform, Flutter, ASP.NET, PHP, and WordPress. He continues to serve as a Lecturer at the Palestinian Technical University, imparting knowledge across various computer science disciplines. Ammar's professional journey includes significant contributions to the industry, specializing in programming controller systems for applications such as soil irrigation, traffic monitoring, smart grids, smart homes, and security systems. His expertise extends to providing diagnostic, programming, and testing services in automotive workshops. Passionate about technological innovation, Ammar remains dedicated to advancing both academic research and practical applications in computer science.

ZULAIHA ALI OTHMAN received a Ph.D. degree in computer science from Sheffield Hallam University in 2004. Since 2003, she has been working on various intelligent system projects, especially in developing intelligent art-based techniques. She is an Associate Professor with the Center for Artificial Intelligence Technology, University of Applied Sciences, and Universiti Kebangsaan Malaysia (UKM). Her comprehensive expertise in framework development, algorithm development, and applied artificial intelligence (AI) solutions to different fields, such as network intrusion detection, human talent, poverty, and air and weather pollution. A number of state, industrial, and foreign ventures have been undertaken. She has published more than 200 articles in many local and international journals, including Expert Systems with Applications, Applied Intelligence, Intelligent Data Analysis, and Applied Soft Computing. She has also supervised more than 30 Ph.D. students from all over the world. In addition to progress in academic work, she has contributed her expertise to the community's progress. (Based on a document published on 20 May 2024).

NOR SAMSIAH SANI received her bachelor's degree in Information Technology from Universiti Tenaga Nasional (UNITEN), Malaysia. She then continued her PhD at the University of Sheffield, United Kingdom, where her thesis focused on machine learning, deep learning, and evolutionary algorithms. She holds the position of Senior Lecturer at the Faculty of Information Science and Technology, Center for Artificial Intelligence Technology (CAIT), Universiti Kebangsaan Malaysia (UKM). She has been invited as a speaker for several seminars on Machine Learning (ML) and Deep Learning (DL). She has also been invited to conduct short courses on similar subjects. She leads several projects involving collaboration with Malaysian government sectors and industries at the national and international levels. Her research interests primarily lie in predictive analytics, machine learning, deep learning, evolutionary algorithms, data mining, optimization, and chemo informatics.

YOUSRA ABUDAQQA is a Ph.D. student at the Universiti Kebangsaan Malaysia (UKM), a research scholar in the Center for Artificial Intelligence Technology (CAIT), Faculty of Information Science & Technology.