scientific reports

Check for updates

OPEN

LLM-enabled adaptive scheduling in IoT sensing for optimized network performance

Muhammad Nawaz Khan[1]✉, Sokjoon Lee[1]✉, Sang Su Lee[2], Mohsin Shah[3]✉, Inam Ullah[3], Shakila Basheer[4,6] & Ali Kashif Bashir[5]

The use of numerous sensors on edge devices, combined with the emergence of AI techniques, makes the IoT environment more intelligent and interactive. The resulting paradigm encompasses device-centric systems that operate instantly and remotely with zero clicks. However, with these advantages, many functional challenges affect remote sensing, including incomplete data, communication delay, lack of context awareness, and dynamically switching topology. To address these challenges, we have proposed a novel scheme, "LLM-Enabled Adaptive Scheduling in IoT Sensing for Optimized Network Performance (LLM-AS)." This scheme uses LLM to adjust the system's sensing to avoid redundant and useless data sending and enhance decision-making for optimized network resources. First, LLM-AS is trained with a defined data set for different parameters, such as packet loss trends, time-based fluctuations, event triggers, network failure patterns, and congestion signals with contextual decisions. Then, this scheme is deployed in a dynamic remote monitoring system for learning and updating task descriptions to utilize the feedback for future decisions and enhance the system performance. Evaluation of LLM-AS on various parameters using the CASAS dataset shows that the optimization functions of LLM are useful and make the IoT more usable. The LLM-AS optimization function confirms an improvement of 57.8% to 60% in MTP, a decrease of 26% to 60% in median delay, and an optimized energy solution with a confidence interval of 95% and a very small error margin. It also indicates that the precision score is about 0.86, the recall score is about 0.82, and the RMSE is about 0.21; all these values suggest high separability for varying conditions of IoT systems in dynamically changing situations.

Keywords Optimized sensing, Energy efficiency, Artificial intelligence, LLM, IoT, Cognitive sensor, Adaptive scheduling

Internet of Things (IoT)[1–3] revolutionized the era of conventional, unintelligent, and passive user-centric approaches into a smart, intelligent, and interactive approach. The system automates each process in a uniform layer of resources for the end user and delivers services according to the user's preferences[4,5]. Data is now carried by IoT-connected devices, allowing them to observe, process, disseminate, and act in response to any event. Most devices are connected using these tiny and pluggable gadgets, where they have turned these memoryless devices into intelligent and interactive smart objects[6]. These smart gadgets are commonly used in healthcare and remote patient monitoring[7–9], agricultural and animal monitoring[10], manufacturing and energy management[11,12], and transportation and logistics[13]. They are usually used to build smart homes, smart campuses, and smart cities[14,15].

With the advent of these devices and IoT applications, our lives have become more comfortable and convenient than ever[16–18]. However, with the increasing number of devices and applications, many functional and technical challenges can arise that degrade system performance. Some of the leading factors are congestion and packet loss, delays and jitter, bandwidth inefficiencies, and lack of contextual understanding[19,20]. Continuous sensing and broadcasting redundant data also affect system resources, especially network energy, where sensors

[1]Department of Smart Security, Gachon University, 1342 Seongnam-daero, Seongnam-si, Gyeonggi-do 13120, Republic of Korea. [2]Cyber Security Research Division, ETRI, Daejeon 34129, Republic of Korea. [3]Department of Computer Engineering, Gachon University, 1342 Seongnam-daero, Seongnam-si, Gyeonggi-do 13120, Republic of Korea. [4]Department of Information Systems, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, 116771 Riyadh, Saudi Arabia. [5]Centre for Research Impact & Outcome, Chitkara University Institute of Engineering and Technology, Chitkara University, Rajpura, Punjab 140401, India. [6]A.K Bashir These authors contributed equally to this work. ✉email: muhammadnawaz@gachon.ac.kr; junny@gachon.ac.kr; symnshah@gachon.ac.kr

are installed for remote data collection[21–23]. Conventional analytical and statistical models did not address these problems. Another issue with IoT is that creating a mesh network of sensors and embedded devices generates a huge amount of data[24,25]. These data packets are always broadcast and circulated on the network, creating extra overhead and consuming a good amount of energy. With limited sensor resources, a vigilant strategy is always desired for careful consumption of energy. In many cases, some functional vulnerabilities affect sensing operations, including incomplete data, delay in communication, lack of context awareness, and dynamically changing structure. In addition, if the right approach is not used when establishing the IoT system, the sensors' energy sources will quickly deplete, and the system will stop working before completing the task. All of these factors not only have an impact on system performance, but also keep it busy, wasting resources on unnecessary operations. With the emergence of AI, the IoT is integrated with some AI techniques, such as LLM, deep learning, reinforcement learning, or federated learning, for decision-making and optimization. By merging AI and IoT, the new models can perform better and increase the efficiency of the system[26–28].

LLMs (Large Language Models), like other fields in AI, have reshaped AI mechanisms by enabling complicated reasoning. It uses some optimization techniques for contextualization of sensor data using different parameters[29–31]. LLM's main aim is to seek and understand the different complex patterns in any dataset and generate context-aware responses by utilizing the predefined constants. They are used equally and are learned in different domains. To address different issues in IoT, LLM uses different generative techniques to improve the decision-making process by controlling and adjusting different parameters[32–34]. With the integration of both LLMs and IoT, the new system is more beneficial in terms of resource management and network optimization. The data obtained from IoT sensors can be used to transfer large volumes to a contextually linear form. This information is fine-tuned for further analysis and optimized network resources according to the context[35–37]. The detected data has been fed into LLMs, and after applying different transforming and tuning techniques, they learn about it and produce more precise and accurate results. With LLM, IoT now reacts in time and with greater accuracy to any context-aware system[38,39]. Using the advanced framework, it handles the inherent problems in IoT, such as handling massive volumes of data, heterogeneity, and human-device interaction. With the integration of LLMs, IoT is now more interactive, efficient in user response, and adaptive to dynamic network conditions[40–43].

To address the above challenges, in this article, we have proposed a novel technique of an LLM-enabled IoT system to optimize IoT performance. It enables the system to make better decisions using the contextual meaning of prompting data based on sensor data. Using LLM inference for decision making improves the scalability, efficiency, and interactive response time of the IoT system.

Problem statement: The main problem with sensing is data loss due to a lack of reliable links and the dynamically changing structure of the IoT. Another issue in IoT is the creation of huge volumes of data with many repeating bit streams and redundant patterns. This repetitive sensing results in excessive energy consumption and keeps all resources on the network. Resource restrictions and overloaded sensor data lead to many additional problems that greatly affect system performance, such as congestion, packet loss, delay, and jitter. Using traditional recovery mechanisms and static network parameters never meets the requirements of fully connected IoT applications in sensing. There is always a need for an AI-based mechanism (LLM) that manages data on the sensing side and improves decision-making for adaptive scheduling. It should learn from pre-defined parameters and data sets. It makes contextual decisions in response to packet loss trends, time-based fluctuations, event triggers, and network failure patterns. To address all these issues, an LLM-based IoT mechatronic that works on contextual meaning and optimizes system performance is always desired.

The main contributions of this research work are as follows:

- Analyze all LLM-based schemes for scheduling IoT sensors, covered and addressed the issues related to optimization and energy management in sensing. We mentioned the implementation of these schemes and how to deal with various performance parameters such as energy, latency, packet loss, and fluctuating network signals.
- Propose a novel LLM-based scheme that is trained on a defined dataset to obtain different parametric values that are concerned with the above challenges. The technique learns more about adaptive scheduling and defines an objective function to adjust network resources. Optimize network resources to adjust different functions and improve sensor scheduling.
- Implement the basic structure of LLM-AS in Python with extensive libraries, with GPT-J-6B for contextual decision making, and check its applicability with various parameters. Finally, the system is tested with different parameters for validation and its applicability in various IoT scenarios.
- Define and utilize different optimization functions on a defined dataset for average energy consumption, median transmission power, and delay with other functional features such as context awareness, ROC AUC, RMSE, and precision recall curve.

The organization of this paper is as follows.

Section-2 is the literature review of LLM-based models deployed in IoT systems; Section-3 is the details of the proposed model of LLM-AS; Section 4 is the evaluation part; and finally, the article is concluded in Sect. 5.

LLM-based models in IoT for optimized sensing

In this section, we have discussed and reviewed all the schemes that have been developed for LLM-based inferencing and are primarily used for optimization techniques. The optimization techniques are used for textual reasoning and high-level instructions, while we need a scheme that works in real-time conditions with adaptive and anomaly scheduling.

LLMSense[44] is a prompting framework that works with low-level and high-level raw data from sensors using LLM logic. It uses selective integration and summarizes the contents of the sensors. Edge devices and cloud servers are used for reasoning and privacy preservation. The main aim was to use LLM for high-level reasoning between sensor raw data, but further a refinement engine is needed, and some low-level techniques are needed to locate these traces. **IoT-LLM**[45] is an LLM-based IoT model that uses perception and a knowledge base to improve capability and scalability. They convert raw IoT data into a format, improve the comprehension for context learning, and create prompts using their knowledge. The model has been tested on five tasks with different kinds of data and complexity. Using the reasoning technique improves the logic, but further experimentation is still needed for scalability in congested user environments. **LC-LLM**[46] is a light LLM-based IoT that uses knowledge distillation to prune simply while preserving interpretive capabilities. It uses a rich dataset to test the capabilities of the model by configuring anomaly detection, predictive maintenance, and user analytics. Reduce inference time and computational overhead on edge devices, but appears to be vague in streamlining the contextual understanding of LLM-based IoT.

LLM-CAT[47] is a framework for helping senior citizens by using IoT integrated with LLMs. Different sensors are embedded to monitor each activity and streamline the information in a voice format. Based on previous data, an IoT box is constructed for checking these activities. It used reasoning to detect low channel data and interpret the GPT-4 interface to provide better QoS, but the system needs further refinement for real-world implementation and should be evaluated for scalability. **LLMEdge**[48] is an LLM-based IoT to address the issues related to computational and memory requirements in dense IoT scenarios. It uses web applications and quantized LLMs to quantize at edge devices. It claims to consume less energy and enhance data privacy. It is more scalable with low latency and with strong and intelligent edge devices. However, it does not explicitly address the issues of computational cost. **EdgeShard**[49] is an LLM-based IoT system proposed to improve efficiency and achieve greater accuracy. Integrate edge devices with a cloud server to provide an easy-to-use pool of distributed services. It uses an adaptive joint device selection model to infer latency and throughput based on dynamic programming. A better approach for reducing latency and no accuracy loss is needed, but it still necessitates more memory for the edge device, which affects the system efficiency.

CoLLM[50] has proposed handling the hierarchical partitioning models with "Compute Bound" that are relevant to the devices with limited resources, like in IoT. It works as a tensor parallelization with a balanced energy consumption. Reduces inference latency by providing an adaptive load-balancing approach. Based on device resources and network conditions, it uses a balanced power consumption and dynamically adapts to new states. Achieve better results in limited devices, but for scalability and dynamic scenarios like real-time IoT, it reduces their effectiveness and efficiency. *Wi-Chat*[51] is an activity identification system based on LLM inference. It utilizes Wi-Fi sensing data for prompt creation. It uses a physical model instead of conventional signal processing for channel state information. It utilizes the concept of zero-shot activity detection by using high-level LLM inferences. It uses practical Wi-Fi sensing for various activities, but the system also does not have an explicit explanation for the computational cost. **Penetrative AI**[52] uses IoT sensors and actuators to collect data and infer the reasoning of LLM for decisions. Sensor data is used as input to the LLM core layer and drives the decisions according to the dynamics of the network. It uses the expertise of knowledge based on the real world and integrates humans into cyber-physical systems. It minimizes many layers of activities that are applied in text-based approaches, but it still needs task dependency and hinders many real-world applications.

LightLLM[53] is a light-based IoT sensor model to optimize historically learned LLMs. A new fusion layer will be used to merge and prepare a contextual prompt. From the pre-trained data, different inputs are integrated to create a lightweight and trainable module. Adjusts to a new task with the same setting and no need for further refinement, which reduces complexity. It uses three sources of light for implementation; however, this needs further refinement, and scalability issues arise in implementation. **LLM-EAI**[54] is using LLM for integration in IoT and event abstraction. It uses raw sensor data in a single prompt that is combined with the logs and applies process mining applications. It works for analysis and for finding high-level activity detection. The best is a simple scenario, but a complex scenario might still require diagnosis and resolution of congestion. **CLAF-IoT**[55] is an LLM-based IoT model to address the issues in dynamic and heterogeneous IoT ecosystems. It is an authentication framework that authenticates the IoT device in a dynamic environment. It uses the LLM-based reasoning for authentication with some historical data. The user behavior and context sensing with federated learning and LLM are integrated for long-term authentication. It ensures authentication accuracy in different IoT contexts, but it still has a greater false acceptance rate.

CASIT[56] is an LLM agent-based IoT that is being used to apply to a wide variety of applications. It uses many intelligent LLM agents to utilize collective intelligence. They have also created a memory mechanism to address the input sensors' data more effectively. The method uses past data for the prompt's local knowledge to obtain good results, but inference time may hinder its use in practical scenarios. **TaskSense**[57] uses a coordinated sensory system for answering the complex user question. Some predefined vocabularies and grammatical rules are derived from the core model of LLMs, and you can learn about TaskSense's sensor language. It uses some task-based dependencies and validates them according to user input. It ensures the sensor's data is timely to enhance

robustness. It uses six LLM layers to be implemented in real-world scenarios such as smart homes. **DrHouse**[58] is an LLM-based multi-turn consultation virtual doctor system. It works in three ways: it improves accuracy and dependability, provides medical information, and is a diagnostic algorithm. Enhances the diagnosis using multi-turn interactive data and helps decide the next stage and action. Better in data acquisition in health, but still faces many functional and technological challenges. There are many technical and functional challenges based on scalability, applications for the real implementation of LLM-based IoT systems[59]. To direct future advances in LLM research, it also identifies research gaps, current trends, and promising avenues.

In all of these schemes, the primary goal is to use LLM as a facilitator for intermediate activities and to align all tasks in a user-understandable format. They cannot use sensor data for decision-making and optimization techniques in the management of the functional properties of IoT. LLM inference can be used to interpret real-world decisions and optimize the function of the system by applying knowledge-based stored and historical data. All of the above schemes have been summarized in Table 1 with their main idea, advantages, and drawbacks.

LLM-enabled adaptive scheduling in IoT sensing for optimized network performance (LLM-AS)

This section provides an overview of the fundamentals and methodology of LLM-AS. It also introduces some preliminary architectural concepts of the system using appropriate notation and formulas.

System architecture of LLM-AS

The main system consists of four layers that perform different tasks, as shown in Fig. 1. Each layer performs specific functions and also organizes the data flow sequentially. In the first layer, cognitive sensors are provided to measure various activities in an open scenario. These sensors collaboratively detect an environment and send the data to a central controller with other sensors in neighboring areas. When different sensors collect data, they send it to gateways at the gateway layer, where all data is combined into packets for further processing. After aggregation at the gateway, the data is sent to the Edge Node layer. The edge node is the place where LLM starts working locally. The data is passed through various procedures for refinement and extraction. Here, the local training module has applied a classification technique to create prompts and fine-tune. This local module shows versatile capabilities for classification tasks that range from textual network commands to dynamic content. The layer's output is passed to the final layer, the 5G edge layer. This layer applies several optimization techniques to determine hyperparameters for the IoT and is also responsible for analyzing the optimization capabilities of LLM.

Network model of LLM-AS

The IoT network consists of a vector of cognitive sensors (S_{cn}), with measurements of different parameters such as temperature, humidity, proximity, and location. Let these S_{cs} sense different values of data; we represent it as a vector of sensing readings as follows.

$$V_{Sn}(t) = V_{S1}(t) + V_{S2}(t) +, ..., +V_{Sn+1}(t) \tag{1}$$

While $V_{Sn}(t)$ is the vector of sensing values is at time *t*. The S_{cn} involvement of continuous streams of bits, these total values are summed up and sent to the gateway layer.

Schemes	Main contribution	Advantages	Limitations
LLMSense[44]	Prompting framework integrating low/high-level sensor data with LLM	Reasoning and privacy via edge-cloud integration	Needs a refinement engine and trace localization
IoT-LLM[45]	Converts raw data for context learning and knowledge-based prompts	Validated on 5 diverse tasks; improves logic	Scalability untested in dense settings
LC-LLM[46]	Lightweight LLM with knowledge distillation for IoT tasks	Low inference time and edge device load	Lacks contextual understanding streamlining
LLM-CAT[47]	Voice-based elderly care using LLM-integrated IoT	Improved QoS and reasoning with GPT-4	Needs refinement and real-world evaluation
LLMEdge[48]	Web-based, quantized LLMs for edge deployment	Low latency, energy-saving, privacy-preserving	Lacks detail on computational cost
EdgeShard[49]	Adaptive device-cloud joint model for inference	Reduces latency, maintains accuracy	Requires large memory on edge devices
CoLLM[50]	Tensor-parallel LLM for adaptive load on IoT	Balanced power use and lower latency	Ineffective in dynamic real-time scenarios
Wi-Chat[51]	LLM-driven activity detection via Wi-Fi sensing	Zero-shot inference, high-level prompt use	Computational cost not explained
Penetrative AI[52]	Human-integrated cyber-physical IoT via LLM	Simplifies layered processing, real-world logic	Task dependency hinders applications
LightLLM[53]	Lightweight contextual prompt fusion layer	Reduces complexity, adapts without retraining	Needs refinement; scalability unclear
LLM-EAI[54]	Event abstraction using logs and raw sensor prompts	Simple high-level activity detection	Complex scenarios need congestion handling
CLAF-IoT[55]	Dynamic LLM-based IoT authentication framework	Accurate in varied contexts with federated learning	High false acceptance rate
CASIT[56]	Agent-based LLM with memory for task performance	Effective local prompt formation	Long inference time for real cases
TaskSense[57]	Multi-layer LLM with task-based sensor coordination	Enhances sensor timing and robustness	Task dependencies need validation
DrHouse[58]	Multi-turn LLM doctor for consultation and diagnostics	Improves diagnosis with interactive reasoning	Faces tech and function limitations

Table 1. Summary of literature in LLM-based IoT schemes.

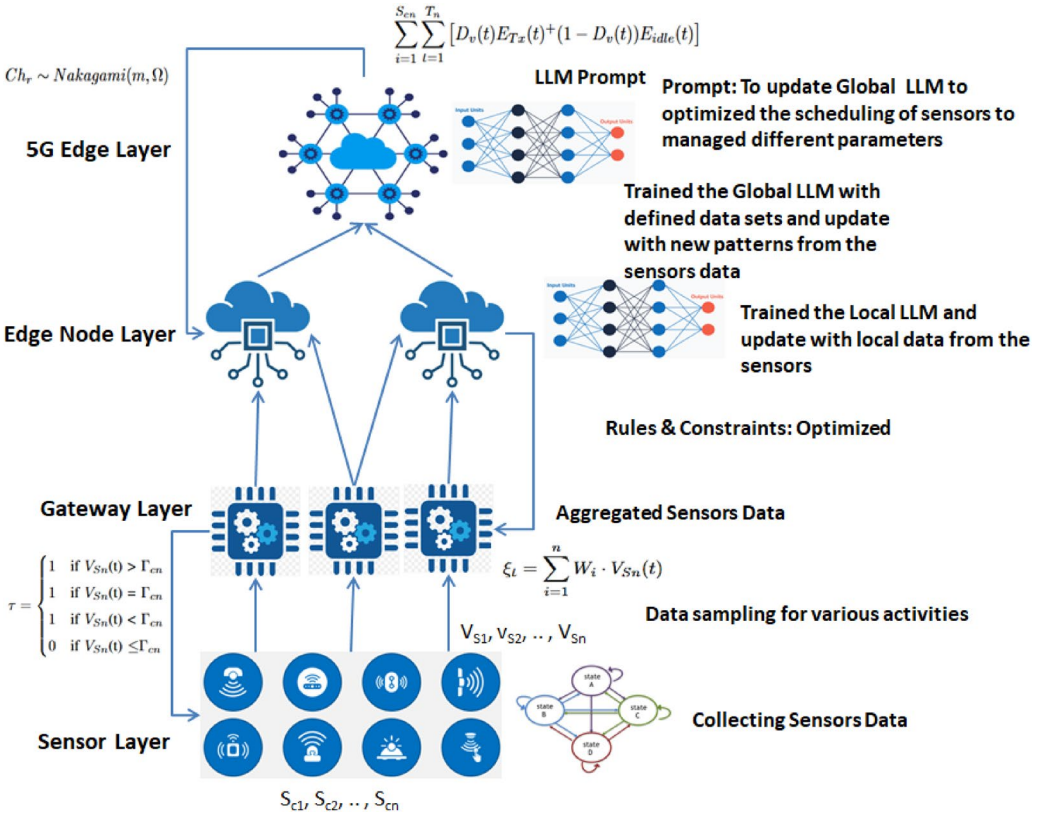

Fig. 1. LLM-AS core functions in Layered-based approach.

$$\xi_t = \sum_{i=1}^{n} W_i \cdot V_{Sn}(t) \tag{2}$$

Where the ξ_t, is the sensed values and W_i, are the weighted values assigned to the nth S_{cn}. Each S_{cn} individually incorporates the sensed data packets and dynamically adjusts the weights. This normalized factor quantifies these weights as follows.

$$W_i = \frac{V_{Sn}(t)}{N_i(t)} \tag{3}$$

$$N_i(t) = \sum_{i=1}^{n} W_i \tag{4}$$

We have defined a threshold value to detect a change in the network Γ_{cn}, which is based on these four cases of S_{cn}. Γ_{cn} is applied, $V_{Sn}(t)$, to calculate the change (τ) in the network operation as per S_{cn}. The change in the network configuration is in the form.

$$\tau = \begin{cases} 1 & \text{if } V_{Sn}(t) > \Gamma_{cn}, \\ 1 & \text{if } V_{Sn}(t) = \Gamma_{cn}, \\ 1 & \text{if } V_{Sn}(t) < \Gamma_{cn}, \\ 0 & \text{if } V_{Sn}(t) \leq \Gamma_{cn}, \end{cases} \tag{5}$$

Calculating the temporal variation in the IoT is the change in the data, the status of S_{cn}, traffic conditions, or any environmental change. Most of the data is periodic time series, event-driven, or periodic. To ensure the robustness of these detected packets using these techniques in the IoT, it can be found as follows.

$$\bar{\xi}_t = \frac{1}{W} \sum_{i=0}^{W-1} \xi_{t-i} \tag{6}$$

In Eq. 6, the W is the time frame window, and the ξ_{t-i} is the network change rate concerning a previous time. These values are important in LLM-AS for handling many parameters, such as redundancy of sensed data packets, data validation for missing bits, and predictive maintenance of all potential failures in IoT. Using the above values of S_{cn} in equation 5, determine the abrupt change or any event detection that is altered after some sensing. Here, the first case is applied on true when there is traffic and all S_{cn} are in a firing state. In the second case, when S_{cn} is equal to the defined threshold, then only transmission takes place in that specific sensor. In the third case, when S_{cn} is less than the threshold, the system is operating normally, and each S_{cn} is sensing normally and sending data. In the last case, when S_{cn}, the less-than-or-equal condition changed, the sensor state was shifted only with the pick data and forwarded. An abrupt change in the network conditions can be summarized as follows.

$$\xi_t = \sum_{i}^{n} W_i \cdot V_{Sn}(t) \geq \nu_t \cdot V_{Sn}(min) \tag{7}$$

Where the ν_t represents the total weighted value of S_{cn}. To find out the exact change in the network conditions of the IoT, any functional changes that lead to deterioration of the system performance are calculated as follows.

$$\xi_t \geq \nu_t \frac{\Gamma_{cn}}{\nu_t} = \Gamma_{cn} \tag{8}$$

From equations 7 and 8, if the C_{cn} conditions are satisfied, that is, $V_{Sn}(t) \geq \frac{\Gamma_{cn}}{\nu_t}$ and it follows that $V_{Sn}(t) \geq \Gamma_{cn}$, Then the system needs to be checked because the IoT system is sensing changes under normal conditions.

Problem formulation of LLM-AS

Many factors need to be optimized in sensed data packets, such as managing a large volume of data, incomplete data due to packet loss, and delay in normal communication. Other functional features, such as a lack of context awareness and the rapidly changing dynamic structure of the IoT, can also require an optimized solution. The main goal of the LLM-AS is to optimize the collection of data to adjust traffic conditions and improve decision-making through contextual LLM techniques. To optimize all these parameters according to the context, the IoT is dynamically scheduled based on S_{cn}. For any S_{cn}, the time tm_i at which the data are transmitted and the network configuration $\tau_{cn}(t)$. Compared with the pre-defined threshold, optimization of the IoT communication schedules provides a uniform solution.

$$min_{t_n, \tau_{cn}(t)=[1,0]} \sum_{t}^{n} tm_i + \tau_{cn}(t) \tag{9}$$

Here, the variations in the values of $\tau_{cn}(t)$ are applied in equation 5. Using this notation and formulas, we can define each parameter.

Delay Optimization: Using the Nakagami-m model[60], many factors are identified, and estimates of loss and delay due to padding. This shows the reliability factor of the traffic of each S_{cn}. The gain in the channel Ch_r is calculated through this process.

$$Ch_r \sim Nakagami(m, \Omega) \tag{10}$$

Where Ch_r is the channel gain in fading, *m* and Ω are the power consumed over time *tm*. The noise in the channel is calculated using a normal distribution with variance σ^2.

$$N_s \sim (0, \sigma^2) \tag{11}$$

In most cases, each S_{cn} independently managed the channel occupation; therefore, the packet loss is individually measured. When the gain is Ch_r=1, the probability density function for Gaussian noise is calculated as follows.

$$f(Ch_r) = \frac{2m^m Ch_r^{2m-1}}{\pi(m)\Omega^m} . e^{-\frac{mCh_r}{\Omega}} \tag{12}$$

Here, $\pi(m)$ is the Gaussian function, and *m* is used to adjust the level of fading; a lower value is important. The lower the values of *m*, the higher the fading feels, which means that there is a high level of noise present in the channel. Due to the dynamic structure of the IoT, the connections are fragile and reconfigured more

frequently, and the values m in the Nakagami distribution must decrease. In Algorithm 1, the details of the process of calculating the delay have been shown.

Input: Set of sensors S_{cn};
Vector of sensor data streams V_{s_n};
Channel estimate $Ch_i \sim \text{Nakagami}(m, \Omega)$
Output: Traffic-adjusted decision Dt_i
for *each sensor* S_{cn_i} **do**
  Estimate channel state CH_r from parameters (m, Ω);
  if $Ch_i < \delta_{low}$ **then**
    Delay is detected;
  else
    if $\tau_i > V_{s_n}(t)$ ***and*** $CH_r > \delta_{high}$ **then**
      Schedule transmission;
    else
      Schedule based on deadline and fairness criteria;
  Update feedback to the local model;
return *Traffic schedule vector for the global model*

Algorithm 1. Delay Optimization for Traffic Adjustment in IoT

Generating $\delta_i(t)$ and Deciding $D_v(t)$: In LLM-AS, the GPT-J-6B model works as a context-aware module and provides some basic functionalities. The main task is to check the condition of IoT and, based on the current state, it creates a suggestion probability $\delta_i(t)$. There are two concerned values of "0" and "1", which are the confidence of LLM for transmission of data in time *t*. In energy optimization and optimization objective function algorithms are based on these as key constraints $(C1 : D_v(t) \leq \delta_i(t))$ for binary decision $D_v(t) \in 1, 0$. For the practical use of GPT-J-6B, we have to look at a single sensor and generalize for the whole IoT system in the following steps.

- Input into LLM-AS: At the start of time *t*, data is collected about real-time events and historical logs. This information is formatted as prompts for the LLM in the context of decision-making. The credentials of the sensor are: $S_{cn} = S_i$; current SNR: $S_{NR}(t) = 5\,\text{dB}$; energy of sensor S_i is $E_{cs}(t) = 50\,\text{mJ}$; history logs: $[D_v(t-3),\ D_v(t-2),\ D_v(t-1)]$; data type: "VitalSigns"; delay: $\tau_i(t) = 2$ (waiting for two time slots to reach the deadline). In the next step, all this information is formatted as a prompt.
- LLM-AS Internal Processing and $\delta_i(t)$ output: This prompt is passed from GPT-J-6B and is analyzed for various tasks. Here, the "reasoning" based on the training data is utilized for various parts of the prompt. S_{NR}(t) = 5 dB is higher than the threshold, the energy level $E_{cs}(t)$ = 50 mJ is not low, and the data are "VitalSigns". By weighing these values, LLM finds the importance of data with SNR and computes a high probability, but not a certainty. The LLM produces the output: $\delta_i(t)$ = 0.85; it means that, "It is strongly recommended to transmit, with an 85% confidence level".
- Input $\delta_i(t)$ into algorithms: The algorithm for optimization takes this value: $\delta_i(t)$ = 0.85. The algorithm-3 makes the decision based on these values in slot t. *Step-1 Initialization:* Let algorithm-3, take this input for decision as $D_v(t)$ = 1 and start transmission. As the data is "VitalSigns", it needs to be transmitted without delay. *Step-2 Apply C1:* $D_v(t) \leq \delta_i(t)$: "Is $1 \leq 0.85$? No, it is false." Therefore, the algorithm must apply the constraint as:

$$\text{if } D_v(t) > \delta_i(t),\ \text{then set } D_v(t) \leftarrow \delta_i(t).$$

Here, GPT-J-6B utilized and returned complete contextual information based on proper reasoning, such as: $E_{sc}(T)$=50 mJ, $\tau_i(t)$=2, and historical parameters such as S_{cs}, $D_v(t-3)$, $D_v(t-2)$, $D_v(t-1)$ with VitalSigns. All of these are embedded with a prompt, and the model is instructed to produce single output values representing the "confidence to transmit". It is a regression-based function: $\delta_i'(t) = f_0(x_i(t))$, and normalized for valid range:$\delta_i(t) = \min(1, \max(0, (\delta_i'(t))))$. Using all these parameters, the model output is $\delta_i(t)$=0.85, and it is interpreted as "As 85% confidence, so recommend transmission". *Step-3 Action:* The values of $D_v(t)$ are 0.85, but it must be binary, and the current value is interpreted as for transmission. Therefore, set the binary decision at $D_v(t)$ = 1. *Step-4 Check Other Constraints:*

$$C2 : (S_{NR}) : D_v(t) * S_{NR}(t) = 1 * 5 \geq 1 * 3 - > 5 \geq 3 - > True.Pass.$$

In C3/C4, the algorithm tests whether $E_{cs}(t) \geq E_{Tx}(t)\,\Delta t$. For instance, if $E_{Tx}(t)\,\Delta t = 10\,\text{mJ}$ and $50\,\text{mJ} \geq 10\,\text{mJ}$, the condition passes, and the energy is updated as $E_{cs}(t+1) = 50 - (1 \times 10) = 40\,\text{mJ}$. For C5, the sum of $D_v(t)$ over the interval $[t,\ t+\tau]$ is checked to ensure that at least one transmission occurs. Since this condition is satisfied, the final decision for time *t* is $D_v(t) = 1$, meaning transmission should proceed. The same process will be used in different situations. If the S_i detects a weak signal of 2 dB with a low battery (15 mJ) and non-priority data, it will be handled accordingly. LLM will analyze this context, but it recognizes no transmission because a transmission attempt would likely fail. The output is very low with $\delta_i(t)$ = 0.10. In this case, the algorithm advises against transmitting.

Energy optimization: For energy in the IoT sensors, all relevant parameters are assigned to minimize the use of energy, which is consumed during data transmission. At the same time, LLM-AS also ensures the reliability of context-based decisions. With some of the constraints, the LLM-AS works on energy optimization and properly manages it at a lower level. The first constraint is $D_v(t) \leq \delta_i(t)$, in which LLM-AS will make a decision when it meets the criteria. The second constraint is $D_v(t) \cdot S_{NR}(t) \geq D_v(t) \cdot S_{NR}\text{min}$, with the calculation of the values of S_{NR}. Another constraint is $E_{cn} = E_{cn}(t) - D_v(t)E_{Tx}(t)\Delta t - (1 - D_v(t))P_{idle}(t)\Delta t$, for energy, when the criteria are met, it will change the network conditions. For the calculation of the delay, another constraint is $\sum_{t'=t}^{t+\tau} D_v(t') \geq 1, \quad \forall i$. Using these constraints, the LLM-AS uses the following method for energy optimization.

$$\min_{D_v(t)} \quad \sum_{i=1}^{S_{cn}} \sum_{t=1}^{T_n} \left[D_v(t) E_{Tx}(t)^{+} (1 - D_v(t)) E_{idle}(t) \right]$$
$$\begin{cases} C1 : D_v(t) \leq \delta_i(t), \\ C2 : D_v(t) \cdot S_{NR}(t) \geq D_v(t) \cdot S_{NR}\text{min}, \\ C3 : E_{cn} = E_{cn}(t) - D_v(t)E_{Tx}(t)\Delta t - \\ (1 - D_v(t))P_{idle}(t)\Delta t, \\ C4 : E_{cs}(t+1) \geq 0, \\ C5 : \sum_{t'=t}^{t+\tau} D_v(t') \geq 1, \quad \forall i \end{cases} \tag{13}$$

Equation 13 is used to measure the energy parameters of the IoT that are based on certain inherent values and constants. The procedure of optimizing energy in IoT networks based on these constraints is shown in the following algorithm 2.

Input: Sensors S_{cn}, SNR $S_{NR}(t)$, LLM output $\delta_i(t)$, transmission energy $E_{Tx}(t)$, idle energy $E_{idle}(t)$
Output: Optimal Transmission Decision $D_v(t)$
for *each communication node* $i = 1$ *to* S_{cn} **do**
  for *each time slot* $t = 1$ *to* T_n **do**
    Estimate SNR $S_{NR}(t)$ and available energy $E_{cs}(t)$;
    Query LLM for decision suggestion $\delta_i(t)$;
    if $D_v(t) \leq \delta_i(t)$ ***and*** $D_v(t) \cdot S_{NR}(t) \geq D_v(t) \cdot S_{NR_{min}}$ **then**
      if $E_{cs}(t) \geq E_{Tx}(t) \cdot \Delta t$ **then**
        Transmit data using energy $E_{Tx}(t)$;
        Update energy: $E_{cs}(t+1) = E_{cs}(t) - D_v(t) \cdot E_{Tx}(t) \cdot \Delta t$;
      else
        Defer transmission, remain idle;
        Update energy: $E_{cs}(t+1) = E_{cs}(t) - E_{idle}(t) \cdot \Delta t$;
    else
      Defer transmission due to low S_{NR} advice;
      Update energy accordingly;
return *Optimized transmission schedule* $D_v(t)$

Algorithm 2. Energy Optimzation for Scheduling in IoT

Optimization Objective: Using the optimization for LLM-AS in Equation 10 for delay and equation 13 for energy, based on these equations, we want to create another optimization objective. This will be the generic

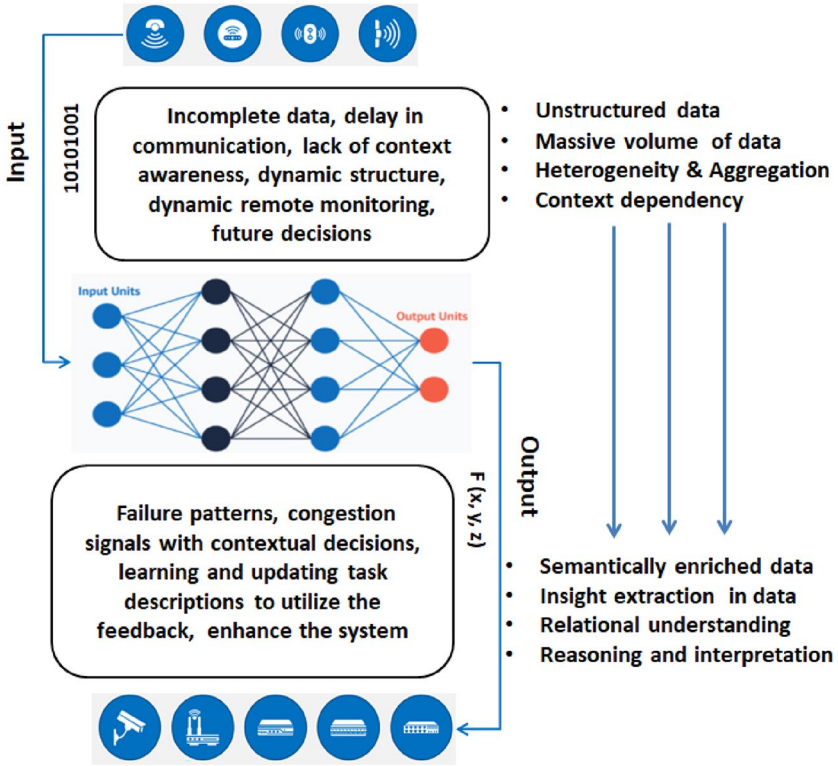

Fig. 2. Optimization operation at the Core model of LLM-AS.

structure for handling all the issues that are already linked to energy and delay. The main aim of the LLM-AS is to optimize a set of functions in terms of addressing the issue of incomplete data due to fading, delay, and jitter in communication; lack of context awareness; dynamic remote monitoring; and future decision-making.

$$\min_{D_v(t)} \sum_{i=1}^{S_{cn}} \sum_{t=1}^{T_n} \Big[D_v(t) E_{Tx}(t) + (1 - D_v(t)) E_{idle}(t) + \alpha_1 DL_i + \alpha_2 C_i(t) \Big] \tag{14}$$

Where $D_v(t) \in \{1, 0\}$ is either 1 for transmission or 0 for non-transmission. This DL_i is the delay penalty used to measure the delay in transmission. $C_i(t)$ is the uncertainty cost of any data completeness or loss. $\alpha_1 \alpha_2$ are weighting factors for delay and uncertainty. These main optimization functions are performed at core optimization modules, as shown in Fig. 2. The values of α_1 and α_2 are important for the delay penalty and uncertainty cost. The main aim is to balance multiple conflicting objectives in IoT, such as minimizing energy consumption and delay. Higher values of α_1, determine more priority to minimizing delay, where timely data delivery is essential, and lower values reduce the emphasis on delays. In the same way, α_1 is an uncertainty weighting factor that measures data completeness, reliability, or potential loss during transmission. In LLM-AS integrates with IoT, these are not fixed but are adaptive coefficients that are learned dynamically. GPT-J-6B dynamically adjusts using this equation.

$$\begin{aligned} \alpha_1(t+1) &= \alpha_1(t) + L_{Rate-1} \cdot \frac{\partial \text{ Loss}(t)}{\partial f(Ch_r)} \\ \alpha_2(t+1) &= \alpha_2(t) + L_{Rate-2} \cdot \frac{\partial \text{Uncertainty}(t)}{\partial D_v(t)} \end{aligned} \tag{15}$$

Where L_{Rate-1} and L_{Rate-2} are learning rates with loss and uncertainty. The objective function with all steps and related operations is explained in Algorithm 3.

Input: Vector Sensors S_{cn}, time horizon T_n, Thresholds $\delta_i(t)$, $S_{NR}^{\min}$, delay DL_i, Uncertainty cost $C_i(t)$
Output: Optimized scheduling decision $D_v(t)$
foreach *sensors* $S_i \in S_{cn}$ **do**
  foreach *time slot* $t \in [1, T_n]$ **do**
    Compute Energy: $E_i(t) \leftarrow D_v(t)E_{Tx}(t) + (1 - D_v(t))E_{idle}(t)$;
      Compute total cost in Energy:
       $J(t) \leftarrow E_i(t) + \alpha_1 DL_i + \alpha_2 C_i(t)$;
    if $D_v(t) > \delta_i(t)$ **then**
      $D_v(t) \leftarrow \delta_i(t)$; // r.t. constraint
    if $D_v(t) \cdot S_{NR}(t) < D_v(t) \cdot S_{NR}^{min}$ **then**
      continue ; // Skip due to signal pading
      Update energy:
       $E_{cn}(t+1) \leftarrow E_{cn}(t) - D_v(t)E_{Tx}(t)\Delta t - (1 - D_v(t))E_{idle}(t)\Delta t$;
    if $E_{cn}(t+1) < 0$ **then**
      $D_v(t) \leftarrow 0$; // Prevent depletion
      Ensure future guarantee:
       if $\sum_{t'=t}^{t+\tau} D_v(t') < 1$ **then**
        Schedule must satisfy: $D_v(t') \leftarrow 1$ for some $t' \in [t, t+\tau]$;
return *Final optimized scheduling vector* $D_v(t)$ *for all sensors*

Algorithm 3. Optimized Scheduling for Objective Function in IoT

For LLM-AS memory footprints, PEFT-based fine-tuning is used to reduce the memory requirement compared to other models. It reduces approximately 57% to 65%. The models are deployed in full size on the edge device and need 2-3 GB, which is feasible and can be used on such platforms. Here, all the processing is done on the edge device rather than on sensors with limited resources. For the latency cost, the inference latency is measured with respect to the edge devices. The latency here is calculated with the LLM-based decision system activated with contextual ambiguity occurring. This activation of the LLM-AS scheduler incorporates 8–11 ms of overhead. This is a lower latency than it saves through optimized scheduling and energy-aware planning. In terms of energy cost, as the inference occurs at the edge devices and not at the sensor, it does not affect the energy at the sensors. The LL-AS optimization objective reduces redundant transmissions and minimizes idle-state, which results in a 12% reduction in overall transmission energy. The main reason for LLM inference to occur is that many scheduling and task coordination in IoT involve contextual ambiguity, incomplete data, and the handling of multiple parameters. These parameters are energy with latency and energy with packet loss. Other models do not handle these parameters and can not adapt to dynamically changing wireless conditions, uncertainty, and complex reasoning over multi-modal signals.

Experimental setup and evaluation

LLM-AS core optimization functions are evaluated for functional parameters such as energy, mean power transmission, delay, accuracy, and efficiency. The system is implemented in Python by integrating the GPT-J-6B[61] into LLM-AS to interpret raw sensor data and suggest energy-efficient or delay-optimized actions. It is a language model based on the Transformer architecture with 28 layers and 6 billion parameters. The model has support scheduling with adaptive rules from contextual data such as temperature, energy, and latency. We have used a trial-and-error-based simulation and finally obtained many optimum solutions. The system is properly evaluated on a widely used dataset, CASAS[62]. This dataset is obtained from sensors that are used for various purposes, such as smart homes, smart farming, and other applications. The following are some of the parameters used to evaluate the LLM-AS and check its applicability and reliability. The system used in simulation is an Intel Core i5-14600KF processor (20 cores) at 3.5 GHz, 32 GB RAM, and a dedicated NVIDIA RTX 5070 Ti GPU. For the edge node, we have used the NVIDIA Jetson AGX Xavier edge module, suitable for embedded AI and edge computing. All these details are mentioned in Table 2 with main components of the model, parameters and description of each parameter.

LLM-AS is fine-tuned to enhance energy efficiency and reduce latency in IoT environments. The hyperparameters used in LLM-AS are mentioned in Table 3, in which the "Learning Rate" is a fine-tuning rate commonly used for LLM. The warm-up steps are the gradual warm-up used in stabilizing the early process. The "Batch Size" is 16 with 3 epochs to adapt to the data without overfitting and 0.01 weight decay to avoid overfitting.

Component	Parameter/Setting	Description
Base model	GPT-J-6B	Transformer-based LLM, Adaptive decision rules and Optimization strategies
Fine-tuning method	PEFT (Parameter-Efficient Fine-Tuning)/FP16 precision for edge	Lightweight adaptation to IoT data, Reduces computational cost and memory requirements
Training dataset	CASAS[62]	energy consumption, transmission power, latencys
Optimized metrics	Energy Consumption, Delay, MTP, PDR	IoT performance parameters to evaluate system efficiency
Prompt engineering	Context-Aware State Encoding	Converts IoT sensor states into structured textual prompts for GPT-J-6B. Example: "Node A: energy=0.45J, delay=10ms, objective: minimize delay and energy."
Learning objective	Adaptive Rule Generation and Decision Support	To generate adaptive control rules for transmission scheduling and power optimization based on dynamic IoT states.
Baseline system	Static Scheduling and Non-Adaptive Configuration	Represents the IoT system before GPT-J-6B integration, using fixed duty-cycling and static transmit power

Table 2. LLM-AS used parameters for IoT optimization Using GPT-J-6B with PEFT.

Hyperparameter	Value used
Learning rate (α)	3e-5
Warmup steps	500
Batch size	16
Epochs	3
Weight decay	0.01
Gradient clipping (max norm)	1.0
LoRA rank	16
LoRA alpha (scaling)	32
Learning rate scheduler	linear decay
Validation split	10 %
Early stopping patience	2 epochs

Table 3. Fine-tuning hyperparameters for LLM-AS on CASAS IoT dataset.

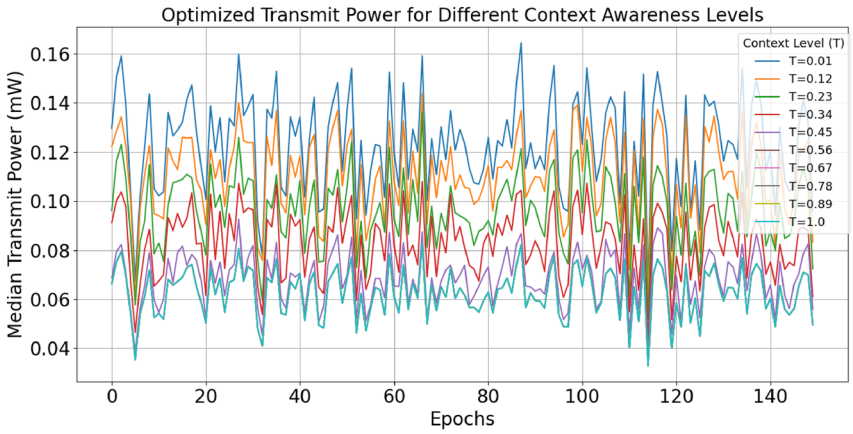

Fig. 3. MTP over variable context awareness levels in LLM-AS.

Optimized transmit power (MTP) with different context awareness level

This parameter is used to check *MTP* for changing the context level with increasing epochs. The LLM-AS has used the inference for optimization, and the optimized values are then mapped to check the behavior of the scheme. The values of each epoch have been observed, and then they are finally mapped with *MTP* against the context level. In the higher level of context awareness (*T*), the system behaves with greater consumption but works more intelligently, as shown in Fig. 3. The system is evaluated at 150 epochs with a change of values of *T* from 0.01 to 1.0. The higher values of *T* (0.8 to 1.0) exhibit a consistent *MTP* around 45 mW, resulting in a 60% reduction in the median delay compared to the baseline (initial) values of *T*. The values are calculated by taking the average of all and then comparing with the initial values of *T*. It may be the initial values such as *T*=0.1, 0.2. These observations show that after applying the LLM-AS optimization function for *MTP*, the more aware the sensors in LLM-AS are, the less energy they consume on average over time, which is the main objective of

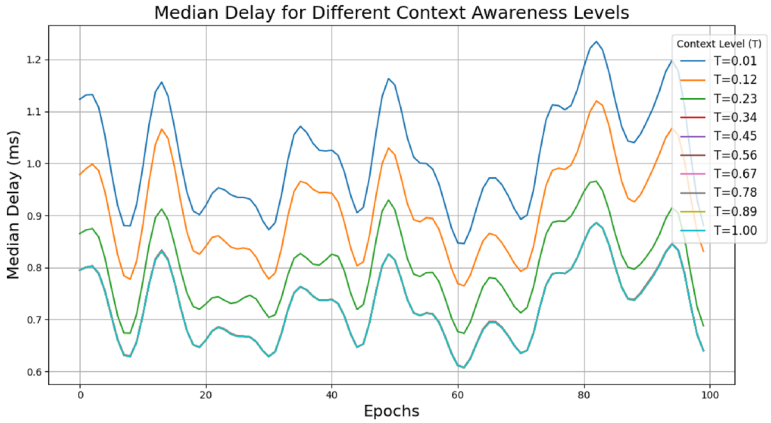

Fig. 4. Delay over variable context awareness levels.

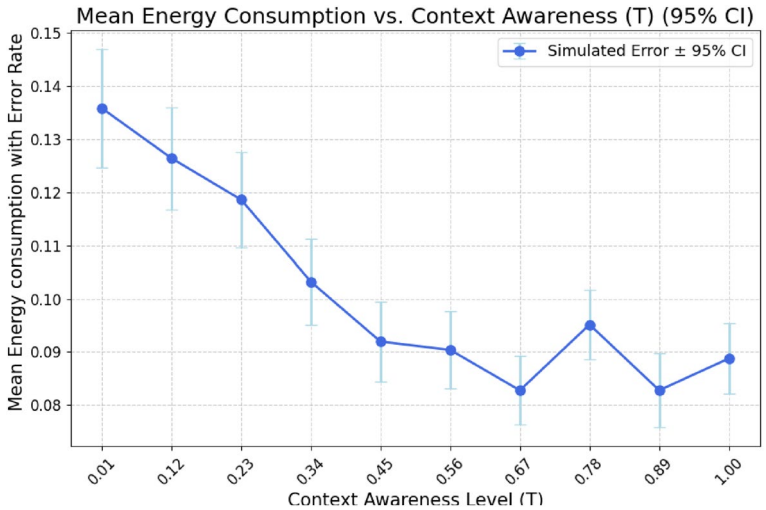

Fig. 5. Energy consumption with variable context awareness in LLM-AS.

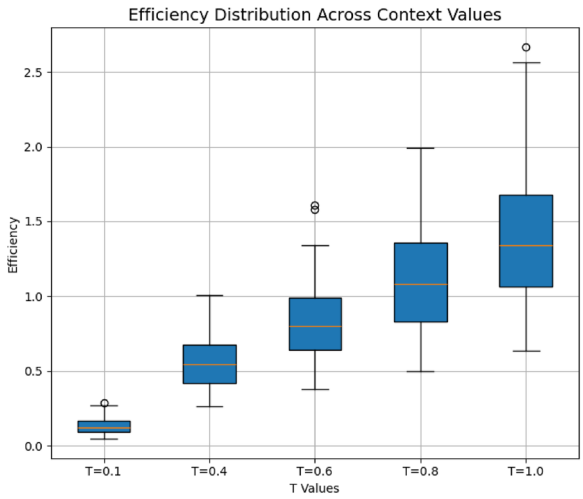

Fig. 6. Efficient distribution across values in LLM-AS.

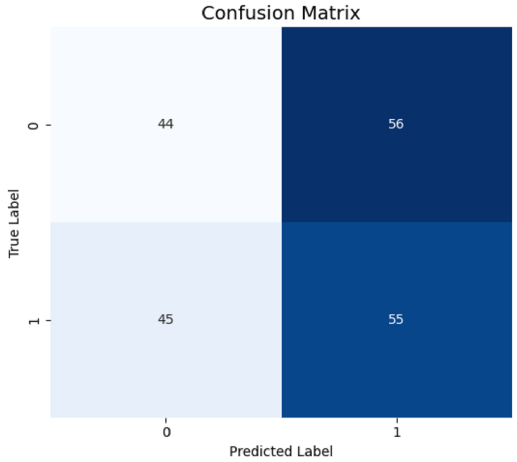

Fig. 7. Confusion Matrix in LLM-AS for calculating Accuracy.

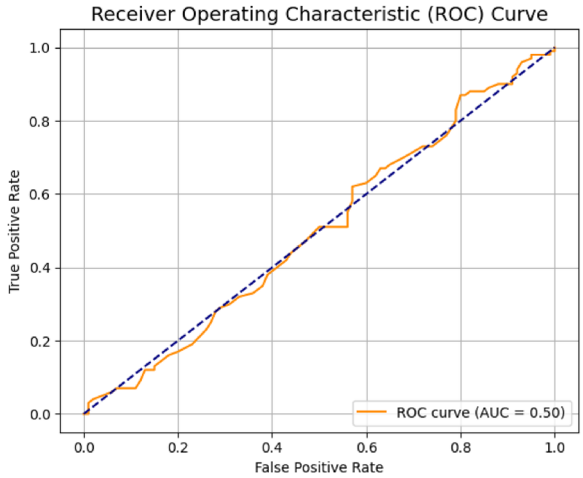

Fig. 8. ROC curve over AUC=0.5.

optimization of the proposed functions. By calculating the average increase, it is shown that the characteristic values of *MTP* improve by approximately 57.8% compared to the context awareness of LLM-AS.

Median delay with varying level of the context awareness

The LLM-AS delay has been calculated with varying values of the level of context awareness, and a final median delay has been determined. Based on the data the system used from the dataset, the level of context awareness enhances the system's intelligence and adaptability for quick decision-making, thereby optimizing the scheduling of C_{cn}. The behavior of the LLM-AS has been mapped in Fig. 4, in which different values of T ranging from 0.01 to 1.0 have been checked with a delay of 100 epochs. In these experiments, the median delay decreases with time, but the values coincide after T= 0.45, and all values remain in the same pattern. The main reason for this behavior is the uniform delay across all context levels after certain gains in these values. The delay changes with context; starting with T=0.01, the value is 26% while at very high sensitivity, such as T=1.0, the value is a 60% delay compared to the baseline. Here, the baseline means the initial values of $T = 0.1$, which evaluates the LLM-AS in the same scenario with the same parameters. The average of all these values is calculated and compared with the baseline (initial) values. In this figure, when T=0.01, the delay value is greater at 1.12 *ms* with variations; however, when T=0.56 and above, the delay is smaller and remains consistently at the same points. All this applies due to LLM-AS, where inference makes the system adaptively modify for optimized decisions. These experiments prove that LLM-AS functions not only optimize functional values but also significantly reduce the delay with efficient performance.

Energy consumption in LLM-AS with the effect of context awareness

LLM-AS's core function is energy optimization in sensing to infer the decision of the cooperative IoT system. Here, the system has been experimented with at some levels of context awareness with energy consumption. The LLM-AS has been checked for different values, and its behavior is shown in Fig. 5. In this figure, the

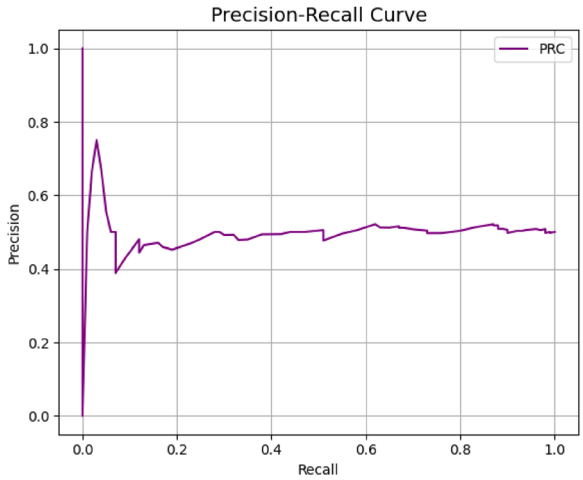

Fig. 9. Precision recall curve in LLM-AS.

Metric	BERT	PALM	OPT-350M	OPT-1.3b	LLM-AS
Precision	0.9058	0.8965	0.9065	0.9172	0.8939
Recall	0.8932	0.8917	0.8904	0.9083	0.8991
F1-Score	0.8741	0.8939	0.8793	0.8878	0.8733
Accuracy	0.8932	0.8921	0.8971	0.9054	0.8901
Latency (ms)	115.48	165.83	88.14	215.36	101.90
Energy Eff.	519.40	361.90	680.70	278.00	249.11

Table 4. Metric-wise performance comparison on the LLM-AS with other models.

mean energy consumption with T is derived from the defined data set. With increasing levels of T, the energy consumption decreases because of the optimization function defined in LLM-AS. After 0.56 of T value, the system is consistently uniform and exhibits a uniform level of energy used. The range T ranges from 0.01 to 1.0. These results confirm optimized functions with increasing energy consumption with increasing T. As the system becomes more intelligent at its operational level, the system uses the same energy consumption uniformly. Ensures the confidence interval is up to 95%, and with a very minor margin of error. This value confirms the statistical reliability for all these values. At T=0.5, the energy consumption is around 5.0 ± 0.4, with a 95% chance that these values will be between 4.6 and 5.4.

Efficiency in context awareness in LLM-AS

Efficiency is calculated from the factors of delay and transmit power with a constant permittivity factor. Here, in LLM-AS, it is calculated as $E_{eff} = \frac{1}{MPT+Delay+\epsilon}$ and, putting the values at different intervals, the response of the system is shown in Fig. 6. In this figure, it is elaborated that the efficacy of the system increases with increasing values of T. With a lower T value, the system shows a slight change, while with a higher T value, it exhibits higher efficiency. It shows the environmental sensitivity and efficiency over an equal number of epochs. With higher T values, the system is more context-aware and exercises more efficiency in data collection. This higher sensitivity is always desired in IoT systems because these systems need a quick response and a data stream from sensors over time.

Here, we have linked the optimization Objective in Eq. 14 with the efficiency evaluation of LLM-AS. This equation is directly linked with Fig. 6, which is used to minimize the composite cost of transmission energy $E_{Tx}(t)$, delay DL_i, and uncertainty cost $C_i(t)$. This is all governed by the adjusted values of $D_v(t)$. As we have proved in simulations that this function, with a set of applied constraints it effectively minimizes the redundant transmissions with idle-state energy losses. It also handled data incompleteness and applied a trade-off between communication and energy consumption. From these experiments, the mean E_{Tx} is decreased to 12%, with a delay penalty of 9% and uncertainty cost dropping by 9%. In $E_{eff} = \frac{1}{MPT+Delay+\epsilon}$, the efficiency is inversely proportional to MTP, Delay and ϵ. Increasing these values (combined/individual) has decreased efficiency and vice versa. As we have measured about a 10-14% increase in efficiency. The rising trend of efficiency with increasing factor "T". It advocates and validates that reducing the objective function in equation 14 effectively improves total energy consumption at the edge nodes, minimizes the latency, and increases contextual awareness for restricted IoT environments.

Scheme	Pre-train-ing	Fine-tuning	Inferen-cing	Data-set	Reas-oning	Cont-Awar	Priv-acy	Del-ay	Ene-rgy	P-Loss	Acc-ura-cy
LLMSense[44]	✓	✗	✓	✓	✓	✓	✓	✗	✗	✗	✓
IoT-LLM[45]	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓
LC-LLM[46]	✓	✓	✓	✓	✓	✗	✗	✓	✓	✗	✓
LLM-CAT[47]	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓
LLMEdge[48]	✓	✗	✓	✓	✓	✗	✓	✓	✓	✗	✓
EdgeShard[49]	✓	✗	✓	✓	✓	✗	✗	✓	✗	✗	✓
CoLLM[50]	✓	✗	✓	✗	✓	✗	✗	✓	✓	✗	✗
Wi-Chat[51]	✓	✗	✓	✓	✓	✗	✗	✗	✗	✗	✓
Pen-AI[52]	✓	✗	✓	✓	✓	✓	✗	✗	✗	✗	✗
LightLLM[53]	✓	✓	✓	✓	✓	✓	✗	✗	✓	✗	✓
LLM-EAI[54]	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓
CLAF-IoT[55]	✓	✗	✓	✓	✓	✓	✓	✗	✗	✗	✓
CASIT[56]	✓	✗	✓	✓	✓	✓	✗	✗	✗	✗	✓
TaskSense[57]	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓
DrHouse[58]	✓	✗	✓	✓	✓	✗	✗	✗	✗	✗	✓
LLM-AS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 5. Feature comparison of LLM-AS with other baseline LLM-based IoT integration schemes.

Confusion matrix, ROC AUC, precision-recall curve, and RMSE

The LLM-AS accuracy and reliability of LLM-AS are measured by applying different parameters in the collection of the dataset with optimization functions. These parameters are the 2x2 confusion matrix, the ROC AUC based on precision-recall, and RMSE. The packet loss at any stage is calculated from the confusion matrix, which has a direct impact on IoT efficiency. The power and delay parameters have already been calculated, and on the basis of these values, the ROC AUC has been calculated. which proves that this model, after optimization, separates positive and negative outcomes for effective adjustment of power and delay. True positive, false negative, true negative, and false positive values make the system more resilient to the precision-recall curve. With an imbalanced dataset, the optimized function is applied uniformly, applying RMSE for robust accuracy. The confusion matrix to verify the results is shown in Fig. 7, which is mapped from the optimized functions that are applied to the dataset. The values obtained suggest that the reliability level of LLM-AS correctly predicted packet loss with a high true positive and true negative rate.

Upon obtaining the ROC AUC score from the predictive MTP and delay, it was around 0.89, as shown in Fig. 8. These values of 0.89 suggest high separability for varying conditions of IoT systems in dynamically changing situations.

LLM-AS confirms that the precision score is about 0.86 and the recall score is about 0.82, reflecting its effectiveness in detecting the true packet loss. It also ensures a low prediction error, with an RMSE of about 0.21, as shown in Fig. 9.

Computational overhead analysis of LLM-AS at the edge node

The edge devices are used in LLM-AS for fine-tuning, and they provide an interface between low-power IoT sensors and the cloud server. Due to the limited computational and processing powers of IoT sensors, it does not host the LLM. They transmit light-weight context vectors such as SNR, energy level, and uncertainty score. The edge device utilized GPT-J-6B inference to produce the transmission-related values $\delta_i(t)$. For LLM-AS, for the edge node layer, we have quantified the memory footprint, processing latency, and energy consumption with its feasibility in resource-constrained IoT.

Memory footprint: Edge node is responsible for executing the fine-tuned GPT-J-6B module, which was not possible to run at IoT sensors.

LLM-AS Model Size: GPT-J-6B has approximately 6B parameters, which are stored in FP16 precision. For storing in memory, it requires 12 GB of storage because 6B $\times$ 2 bytes/parameter. This was not possible for lower-range micro-controllers such as sensors.

Data Compression: For feasible deployment, post-training quantization needs to be applied. This reduces the precision from FP16 to INT4. After applying compression, the model size condenses to approximately 3 GB.

Required Memory: In inference, INT4 must be loaded into RAM with at least 3 GB of space. Some additional memory may be needed as a cache during token generation. The total approximated memory would be 4 GB.

Processing Time: The average inference latency is calculated here. Edge Setup: The quantized model in the form INT4 is deployed on an NVIDIA Jetson AGX Xavier edge module. This module is suitable and recommended for autonomous machines and edge devices.

Latency: δ_i is an output produced from a single forward pass, which requires a window of 512 tokens. Latency calculated in this case is around 150 milliseconds. *Energy Overhead:* The energy consumed by the edge device is considered here to determine how it affects the model.

Energy Measurement: There are many types of sensors used, but here we use the values of the onboard power sensors of the Jetson module. The power consumed during the LLM inference is calculated. Normally, these modules utilize an idle power is about 8W. In 150 ms of active inference, it consumes about 25W. To calculate

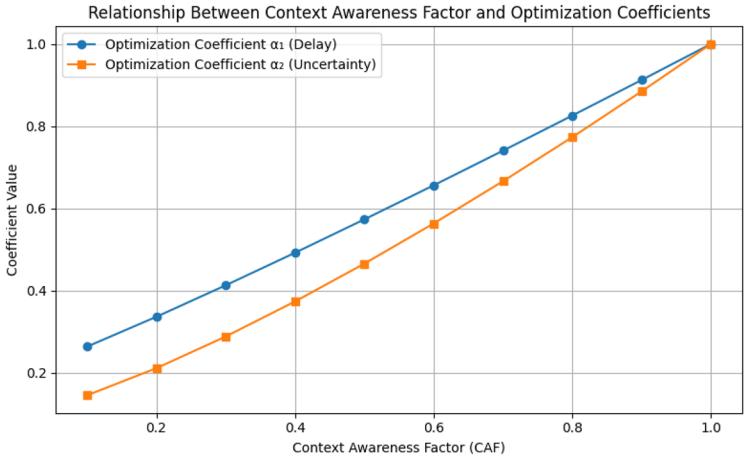

Fig. 10. Relationship between context awareness and optimization coefficients in LLM-AS.

Level	$\delta_i(t)$	α_1 (Delay)	α_2 (Data-Loss)	System Behavior
Low (0.1–0.3)	Low confidence	↓ low	↓ low	System avoids transmission and saves energy.
Medium (0.4–0.6)	Balanced	medium	medium	Balanced delay–energy scheduling.
High (0.7–1.0)	High confidence	↑ high	↑ high	Immediate transmission; protects important packets.

Table 6. Relationship between context-awareness level and optimization coefficients.

the additional energy per decision is E_{Edge}, and it can be calculated as: $E_{Edge} = P_{inf} - P_{idel}$ (t) = (25W - 8W) $\times$ 0.15 s = 2.55 Joules.

From these experimental values for the edge node, we calculate the computation overhead. The 8-16 GB RAM is not possible for sensors, and LLM-AS resides on the edge node only. For latency, 150 ms per decision is acceptable in sensing scenarios. While 2.55 Joules is a substantial expense for a tiny sensor, it is spent at the Edge Node, which is usually powered by an accurate power source.

Comparative analysis of LLM-AS

LLM-AS has been checked and compared with other models for various critical performance parameters, which justifies the use of LLM in IoT systems. For comparison, we have used some published results of already deployed models, with results obtained after running LLM-AS. The base model of LLM-AS is GPT-J-6B, where other schemes use other models. These values are presented in Table 4, for various parameters such as precision, recall, F1-score, accuracy, latency, and energy efficiency. We have utilized the published results of BERT, PALM, and OPT-350 for analysis with LLM-AS[63]. In this analysis, BERT gets a precision of 0.9058, a recall of 0.8932, an F1-score of 0.8741, and an accuracy is about 0.8932. It utilizes transformer depth and limited contextual scaling. The PALM provides a low precision is about 0.8965, with a higher F1-score is about 0.8939, accuracy is about 0.8921, and a higher latency is about 165.83 ms. It is due to the complex attention procedure. OPT-350M with smaller parameterization and lighter architecture, achieves a precision of 0.9065, an accuracy is about 0.8971, a lower latency of about 88.14, and an F1-score is 0.8793. OPT-1.3b also has low computational overhead, with the highest precision being about 0.9172, recall is 0.9083, accuracy is about 0.9054, and latency is 215.36 ms with energy efficiency 278. This analysis concludes that LLM-AS maintains a better accuracy of about 0.8901, a recall is 0.8991, and experiences a latency of 101.90 ms with an energy efficiency of 249.11. It also shows a lower precision of 0.8939, and the F1-score is 0.8733. LLM-AS utilizes adaptive learning mechanisms and optimized attention layers, leading to better efficiency and reduced overhead.

A comparative analysis of LLM-AS with other baseline schemes was also performed based on functional properties. These are LLM-based features like pre-training, fine-tuning, and inferencing, and some others are general features as shown in Table 5. The generic features include whether or not a dataset is used, whether or not reasoning is applied, and a variety of other factors such as context awareness, privacy and security, delay, energy efficiency, packet loss, and accuracy. The four features of LLM-AS are the core functionality that is achieved by the optimization objective in Equation 14. These are context-aware event handling, delay optimization, energy optimization, and packet loss. In context-awareness, LLM-AS leverages fine-grained sensor context for node-level scheduling, such as network load and edge-node behavior. The values of $D_v(t)$ in 14 are used to derive states with uncertainty cost $C_i(t)$ to find incomplete, inconsistent, or noisy data. From the experiments, it is proven that a 9% reduction in uncertainty cost which useful in terms of reasoning even in a limited IoT.

In delay optimization, LLM-AS uses a delay reduction objective that is linked directly to its task-scheduling method, while other schemes do not consider delay as an optimization target. In Equation 14, the delay factor DL_i utilizes the time-based latency and consistently adapts scheduling decisions to minimize latency. There

is a 9% minimization in delay penalty, which proves the proposed system's ability to optimize the latency. In comparison of operational energy with baseline models, most of the models perform lightweight reasoning. In the case of LLM-AS, energy consumption is managed by applying adaptive task and resource-aware inference pathways. $E_{Tx}(t)$ with certain constraints, Equation 14 easily manages unnecessary packet exchanges and reduces energy consumption by applying an idle-state policy. The experimental results confirm a 12% reduction in transmission energy as it reduces communication overhead and latency. Other schemes do not apply packet loss, while LLM-AS integrates packet loss analysis. Using $D_v(t)$, it minimizes retransmissions and mitigates the impact of incomplete packets. Adaptive handling is not handled in most models, while in LLM-AS, it is mainly coupled in the main objective functions.

Context awareness factor Vs optimization coefficients

The factors of context awareness have an influence on the optimization coefficients $\alpha 1$ and $\alpha 2$. In all experiments, the values of δ_i increase from 0.1 to 1.0, LLM-AS acquired by increasing $\alpha 1$ (delay) and $\alpha 2$ (uncertainty). These statistical results clarify the quicker and more effective scheduling of sensor data. There are different levels of context awareness in LLM-AS. From 0.1 to 0.3 is a very low level, meaning too little understanding of LLM, while from 0.4 to 0.6 is a moderate level of context awareness with normal system states. The high level of context awareness in LLM-AS from 0.7 to 1.0 means that it understands critical data such as low latency, energy consumption factors, and high SNR values. The relationship between these two factors is shown in Fig. 10. This means that LLM-AS performs better in high context levels $\delta_i(t) \in \{0.1, 1.0\}$. The results clearly mention that context awareness is high, both coefficients have increased, and it adapts to faster scheduling with lower latency. It also reduces the uncertainty cost and seeks better reliability. All these values are reflected in Equation 14, and illustrate how higher levels of context-awareness enforce a faster scheduling mode with increasing values of $\alpha 1$ and $\alpha 2$. The summary of these experiments is mentioned in Table 6.

Conclusion and future directions

The extensive use of AI techniques in IoT systems, especially in remote sensing, makes these systems more realistic, intelligent, and responsive. In this article, we have proposed a novel mechanism for using LLM for contextual meaning and optimization at the sensor level. The system uses some optimization techniques to adjust the inference of LLM. Specifically, our goal was to minimize massive volumes of data in remote sensing. We have also handled some other functional challenges that affect sensing operations, including incomplete data, communication delay, lack of context awareness, and dynamically changing topology. The "LLM-Enabled Adaptive Scheduling in IoT Sensing for Optimized Network Performance (LLM-AS)" uses the LLM to adjust the system's sensing to avoid redundant and useless data sending and enhance decision-making for optimized network resources. It is trained with the CASAS dataset for different parameters such as packet loss trends, time-based fluctuations, event triggers, network failure patterns, and congestion signals with contextual decisions. The LLM-AS optimization function confirms an improvement of up to 60% in MTP, a decrease of 60% in median delay, and an optimized energy solution with a confidence interval of 95%, and a very small error margin. It also shows improvements in precision score, recall score, and RMSE.

In the future, we want to verify LLM-AS with multiple datasets and also test other features in these datasets. The system will be compared with other implemented systems and its applicability in a real-world scenario. The same idea can be implemented with reinforcement learning, while in a distributed system, it can be implemented in federated learning.

Data availability

All data generated during this study are included in this published article, and no such data were hidden to support this study.

Received: 12 August 2025; Accepted: 11 December 2025
Published online: 21 April 2026

References

1. Borgosz, L. & Dikicioglu, D. Industrial internet of things: what does it mean for the bioprocess industries?. *Biochem. Eng. J.* **201**, 109122 (2024).
2. Khan, M. N. et al. Self-adaptive and content-based scheduling for reducing idle listening and overhearing in securing quantum iot sensors. *Internet of Things* **27**, 101312 (2024).
3. Domínguez-Bolaño, T., Barral, V., Escudero, C. J. & García-Naya, J. A. An iot system for a smart campus: challenges and solutions illustrated over several real-world use cases. *Internet of Things* **25**, 101099 (2024).
4. Dui, H., Zhang, S., Liu, M., Dong, X. & Bai, G. Iot-enabled real-time traffic monitoring and control management for intelligent transportation systems. *IEEE Internet Things J.* **11**(9), 15842–15854 (2024).
5. Park, C., Lee, H., Lee, S. & Jeong, O. Synergistic joint model of knowledge graph and llm for enhancing xai-based clinical decision support systems. *Mathematics* **13**(6), 949 (2025).
6. Khan, M. N., Rahman, H. U., Hussain, T., Yang, B. & Qaisar, S. M. Enabling trust in automotive iot: lightweight mutual authentication scheme for electronic connected devices in internet of things. *IEEE Trans. Consumer Elect.* **7**(3), 5065–5078 (2024).
7. Kazi, K. *Ai-driven iot (aiiot) in healthcare monitoring* (IGI Global Scientific Publishing, New York, 2024).
8. Al-Nbhany, W. A., Zahary, A. T. & Al-Shargabi, A. A. Blockchain-iot healthcare applications and trends: a review. *IEEE Access* **12**, 4178–4212 (2024).
9. Amiri, Z., Heidari, A., Zavvar, M., Navimipour, N. J. & Esmaeilpour, M. The applications of nature-inspired algorithms in internet of things-based healthcare service: A systematic literature review. *Trans. Emerg. Telecommun. Technol.* **35**(6), 4969 (2024).
10. Mohan, K., Kannan, M.N., Mohan, M., Nathan, S.S.: Animo–animal monitoring system based on internet of things. In: 2024 IEEE 15th Control and System Graduate Research Colloquium (ICSGRC), pp. 337–341 (2024). IEEE

11. Rath, K.C., Khang, A., Roy, D.: The role of internet of things (iot) technology in industry 4.0 economy. In: Advanced IoT Technologies and Applications in the Industry 4.0 Digital Economy, pp. 1–28. CRC Press, ??? (2024)
12. Kumari, M., Pramanick, N., Agarwal, M. & Esenogho, E. An optimized ids framework for big data environments: Integrating gravitational search and smote-ipf data balancing for high-accuracy ids. *SN Computer Sci.* **6**(7), 786 (2025).
13. Myagmar-Ochir, Y. & Kim, W. A survey of video surveillance systems in smart city. *Electronics* **12**(17), 3567 (2023).
14. Zarpellon, B. O., Oro Arenas, L., Godoy, E. P., Marafão, F. P. & Paredes, H. K. M. Design and implementation of a smart campus flexible internet of things architecture on a brazilian university. *IEEE Access.* **12**, 113705–113725 (2024).
15. Ugli, D. B. R., Mohammed, A. F., Na, T. & Lee, J. Deep reinforcement learning-empowered cost-effective federated video surveillance management framework. *Sensors* **24**(7), 1–19 (2024).
16. Menon, A. *IoT in Everyday Life* (Educohack Press, Delhi, 2025).
17. Kim, J., Kim, H.: Koacd: The first korean adolescent dataset for cognitive distortion analysis. arXiv preprint arXiv:2505.00367 (2025)
18. Esenogho, E., Djouani, K. & Kurien, A. M. Integrating artificial intelligence internet of things and 5g for next-generation smartgrid: A survey of trends challenges and prospect. *IEEE Access* **10**, 4794–4831 (2022).
19. Zubaydi, H. D., Jagmagji, A. S., Molnár, S. & Alzubaidi, M. Alcs-pp Adaptive latency control scheme for packet pacing in self-clocked congestion control. *IEEE Access.* **13**, 77080–77123 (2025).
20. Chakravarty, S.K., Batra, A., Singh, N., Kumar, G.: Adaptive scheduling using the mqtt protocol in the iot based on environmental sensing and network load. In: 2025 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS), pp. 1–7 (2025). IEEE
21. Khan, M. N., Lee, S. & Shah, M. Adaptive scheduling in cognitive iot sensors for optimizing network performance using reinforcement learning. *Appl. Sci.* **15**(10), 5573 (2025).
22. Khan, M. N. et al. Energy-efficient dynamic and adaptive state-based scheduling (edass) scheme for wireless sensor networks. *IEEE Sensors J.* **22**(12), 12386–12403 (2022).
23. Mishra, S. D. & Verma, D. Energy-efficient and reliable clustering with optimized scheduling and routing for wireless sensor networks. *Multim. Tools Appl.* **83**(26), 68107–68133 (2024).
24. Ullah, I., Khan, I. U., Ouaissa, M., Ouaissa, M. & El Hajjami, S. *Future Communication Systems Using Artificial Intelligence Internet of Things and Data Science* (CRC Press, Boca Raton, 2024).
25. Kipongo, J., Swart, T. G. & Esenogho, E. Artificial intelligence-based intrusion detection and prevention in edge-assisted sdwsn with modified honeycomb structure. *IEEE access* **12**, 3140–3175 (2023).
26. Miah, M. S. U. et al. A multimodal approach to cross-lingual sentiment analysis with ensemble of transformer and llm. *Scientific Rep.* **14**(1), 9603 (2024).
27. Belhaouari, S. B. & Kraidia, I. Efficient self-attention with smart pruning for sustainable large language models. *Scientific Rep.* **15**(1), 10171 (2025).
28. Kim, S. J. & Lee, B. M. A novel approach to password strength evaluation using chatgpt-based prompt metrics. *IEEe Access.* **12**, 175071–175080 (2024).
29. Zhang, C., Qu, D., Du, L., Yang, K.: Unsupervised machine translation based on dynamic adaptive masking strategy and multi-task learning. In: Proceedings of the International Conference on Image Processing, Machine Learning and Pattern Recognition, pp. 634–639 (2024)
30. Kim, J., Lee, H.-Y., Kim, J.-H. & Kim, C.-E. Development of an llm-based cpx practicing chatbot for korean medicine education: implementation of automated scoring and feedback generation framework. *J. Korean Med.* **45**(4), 215–230 (2024).
31. Zhou, H. et al. Large language model (llm) for telecommunications: A comprehensive survey on principles, key techniques, and opportunities. *IEEE Commun. Surveys Tutorials.* **27**, 1955–2005 (2024).
32. Cui, H., Du, Y., Yang, Q., Shao, Y. & Liew, S. C. Llmind: Orchestrating ai and iot with llm for complex task execution. *IEEE Commun. Magazine.* **63**, 214–220 (2024).
33. Park, S.-Y. & Kim, C.-E. Enhancing korean medicine education with large language models: Focusing on the development of educational artificial intelligence. *J. Physiol. Pathol. Korean Med.* **37**(5), 134–138 (2023).
34. Zou, Z., Mubin, O., Alnajjar, F. & Ali, L. A pilot study of measuring emotional response and perception of llm-generated questionnaire and human-generated questionnaires. *Scientific Rep.* **14**(1), 2781 (2024).
35. Park, H. J., Kim, E. J. & Kim, J. Y. Exploring large language models and the metaverse for urologic applications: potential, challenges, and the path forward. *Int. Neurourol. J.* **28**(Suppl 2), 65 (2024).
36. Kim, D., Lee, K., Lee, Y. & Woo, H. Acmfed: Fair semi-supervised federated learning with additional compromise model. *IEEE Access.* **13**, 47734–47747 (2025).
37. Zhang, X., Sun, W., Chen, K. & Song, S. A novel method for intelligent operation and maintenance of transformers using deep visual large model detr+ x and digital twin. *Scientific Rep.* **15**(1), 98 (2025).
38. Majeed, A. & Hwang, S. O. Reliability issues of llms: Chatgpt a case study. *IEEE Reliabil Magazine.* **1**, 36–46 (2024).
39. Cho, T., Kim, R. & Choi, A. J. Research on enhancing model performance by merging with korean language models. *Eng. Appl. Artif. Intell.* **159**, 111686 (2025).
40. Song, I., Lee, K.: Optimizing communication and performance in federated learning for large language models. In: 2025 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), pp. 0964–0967 (2025). IEEE
41. Li, C.-Y. et al. Towards a holistic framework for multimodal llm in 3D brain CT radiology report generation. *Nature Commun.* **16**(1), 2258 (2025).
42. Hong, Y., Wu, J. & Morello, R. Llm-twin: Mini-giant model-driven beyond 5g digital twin networking framework with semantic secure communication and computation. *Scientific Rep.* **14**(1), 19065 (2024).
43. Kipongo, J., Swart, T.G., Esenogho, E.: Design and implementation of intrusion detection systems using rpl and aovd protocols-based wireless sensor networks. International Journal of Electronics and Telecommunications, 309–318 (2023)
44. Ouyang, X., Srivastava, M.: Llmsense: Harnessing llms for high-level reasoning over spatiotemporal sensor traces. In: 2024 IEEE 3rd Workshop on Machine Learning on Edge in Sensor Systems (SenSys-ML), pp. 9–14 (2024). IEEE
45. An, T., Zhou, Y., Zou, H., Yang, J.: Iot-llm: Enhancing real-world iot task reasoning with large language models. arXiv preprint arXiv:2410.02429 (2024)
46. Kannadasan, T.: Lightweight contextual llms for iot data interpretation in smart cities. In: 2025 International Conference on Visual Analytics and Data Visualization (ICVADV), pp. 909–914 (2025). IEEE
47. Sun, Y., Ortiz, J.: An ai-based system utilizing iot-enabled ambient sensors and llms for complex activity tracking. arXiv preprint arXiv:2407.02606 (2024)
48. Ray, P.P., Pradhan, M.P.: Llmedge: A novel framework for localized llm inferencing at resource constrained edge. In: 2024 International Conference on IoT Based Control Networks and Intelligent Systems (ICICNIS), pp. 1–8 (2024). IEEE
49. Zhang, M., Shen, X., Cao, J., Cui, Z. & Jiang, S. Edgeshard. Efficient llm inference via collaborative edge computing. *IEEE Internet Things J.* **12**, 13119–13131 (2024).
50. Li, J., Han, B., Li, S., Wang, X., Li, J.: Collm: A collaborative llm inference framework for resource-constrained devices. In: 2024 IEEE/CIC International Conference on Communications in China (ICCC), pp. 185–190 (2024). IEEE
51. Zhang, H., Ren, Y., Yuan, H., Zhang, J., Shen, Y.: Wi-chat: Large language model powered wi-fi sensing. arXiv preprint arXiv:2502.12421 (2025)

52. Xu, H., Han, L., Yang, Q., Li, M., Srivastava, M.: Penetrative ai: Making llms comprehend the physical world. In: Proceedings of the 25th International Workshop on Mobile Computing Systems and Applications, pp. 1–7 (2024)
53. Hu, J., Jia, H., Hassan, M., Yao, L., Kusy, B., Hu, W.: Lightllm: A versatile large language model for predictive light sensing. In: Proceedings of the 23rd ACM Conference on Embedded Networked Sensor Systems, pp. 158–171 (2025)
54. Shirali, M., Sani, M.F., Ahmadi, Z., Serral, E.: Llm-based event abstraction and integration for iot-sourced logs. In: International Conference on Business Process Management, pp. 138–149 (2024). Springer
55. Rehman, A. et al. Claf-iot: Context-aware llms-enhanced authentication framework for internet of things. *IEEE Internet Things J.* **12**, 28639–28646 (2025).
56. Zhong, N. et al. Casit: Collective intelligent agent system for internet of things. *IEEE Internet Things J.* **11**(11), 19646–19656 (2024).
57. Liu, K., Yang, B., Xu, L., Guo, Y., Xing, G., Shuai, X., Ren, X., Jiang, X., Yan, Z.: Tasksense: A translation-like approach for tasking heterogeneous sensor systems with llms. In: Proceedings of the 23rd ACM Conference on Embedded Networked Sensor Systems, pp. 213–225 (2025)
58. Yang, B. et al. Drhouse: An llm-empowered diagnostic reasoning system through harnessing outcomes from sensor data and expert knowledge. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* **8**(4), 1–29 (2024).
59. Mienye, I. D. et al. Large language models: An overview of foundational architectures, recent trends, and a new taxonomy. *Discover Appl. Sci.* **7**(9), 1027 (2025).
60. Karagiannidis, G. K., Sagias, N. C. & Mathiopoulos, P. T. Nakagami: A novel stochastic model for cascaded fading channels. *IEEE Trans. Commun.* **55**(8), 1453–1458 (2007).
61. Wang, B., Komatsuzaki, A.: GPT-J-6B: A 6 billion parameter autoregressive language model (2021)
62. Cook, D.J.: CASAS Datasets. https://casas.wsu.edu/publications/. Accessed: 2025-06-01 (n.d.)
63. Otoum, Y., Asad, A., Nayak, A.: Llms meet federated learning for scalable and secure iot management. arXiv preprint arXiv:2504.16032 (2025)

Author contributions

MN: Conceptualization, methodology, data curation, writing of original draft preparation. SL: Software, Validation, Visualization, Investigation. SSL: Formal analysis, writing & editing. MS: Funding acquisition, Project administration. SB: Software, Visualization, Investigation. IU: Supervision, Writing-review & editing, Conceptualization. AKB: Funding acquisition, Investigation, Resources.

Funding

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2025-02215590, Development of AI implementation obfuscation technology to prevent information leakage in On-Device AI).

This research is also supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R195), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Declarations

Ethical approval

This study does not involve human participants, animals, or sensitive personal data. All simulations and analyses were conducted using original or publicly available datasets, and all data sources have been properly cited. The research follows standard ethical guidelines for scientific integrity, transparency, and responsible data usage.

Competing interests

The authors declare no competing interests. The authors declare that the research was conducted without commercial or financial relationships that could be construed as a potential conflict of interest.

Additional information

Correspondence and requests for materials should be addressed to M.N.K., S.L. or M.S.