Performance of Large Language Models Across Edge and Cloud Platforms in Smart Spaces

Aygün Varol*, Naser Hossein Motlagh†, Mirka Leino‡, and Johanna Virkki*
*Faculty of Information Technology and Communication Sciences, Tampere University, Finland
†Department of Computer Science, University of Helsinki, Finland
‡Faculty of Technology, Satakunta University of Applied Sciences, Finland
*aygun.varol@tuni.fi, †naser.motlagh@helsinki.fi, ‡mirka.leino@samk.fi, *johanna.virkki@tuni.fi

***Abstract*—Smart spaces integrate sensing and computing technologies with artificial intelligence (AI) to enhance adaptability and user convenience. Incorporating Large Language Models (LLMs) into these environments can further enhance their capabilities, transforming them into interactive spaces that provide user-centric and personalized services. In this paper, we present a hybrid edge-cloud approach for a generic smart space that integrates open-source LLMs with commonly available edge devices (laptops and Raspberry Pis), a commercial cloud platform, historical environmental data, real-time sensor data streams, and LLM prompts. This integration enables various types of analytics including *Descriptive*, *Diagnostic*, *Predictive*, and *Prescriptive*, ranging from basic to advanced analyses. Our proposed approach enables dynamic selection between edge and cloud resources to evaluate the performance of LLMs in terms of response latency and accuracy, leveraging these data analytics. Our evaluation results demonstrate the potential of LLMs to transform smart spaces into more intelligent and interactive environments that deliver enhanced user-centric services.**

***Keywords*—Smart Spaces, Internet of Things, Edge Computing, Environmental Monitoring, and Large Language Models.**

I. Introduction

Recent advancements in Artificial Intelligence (AI) have further transformed smart spaces into interactive, user-centric environments capable of learning from user behavior and environmental changes [1]. Among AI technologies, Large Language Models (LLMs) offer particularly promising capabilities including natural language understanding and contextual reasoning, which can unify diverse IoT functionalities through intuitive, human-centric interaction [2]. LLMs can learn user habits, infer context from multimodal sensor data, and enable seamless language-based control of smart environments [3], [4]. In real-world systems and applications, LLMs employ Retrieval-Augmented Generation (RAG) techniques, which enable the integration of additional data sources. In smart spaces, incorporating sensor data as RAG input allows LLMs to interpret sensor outputs into meaningful, human-readable text, enhancing interaction and usability [5].

However, deploying LLMs in smart spaces presents critical challenges. Cloud-based LLMs, such as GPT-4o [6], achieve state-of-the-art performance but rely heavily on continuous connectivity, raising concerns around latency, privacy, and reliability. In contrast, deploying LLMs on resource-constrained edge devices enhances local responsiveness and data locality, but often suffers from limited model capacity and degraded accuracy [7]. Hybrid approaches, including collaborative inference frameworks such as EdgeShard [8], attempt to partition model workloads across edge and cloud to balance these trade-offs. Yet existing hybrid solutions are typically arranged for specific deployment scenarios and lack generalizable frameworks applicable to heterogeneous smart space environments.

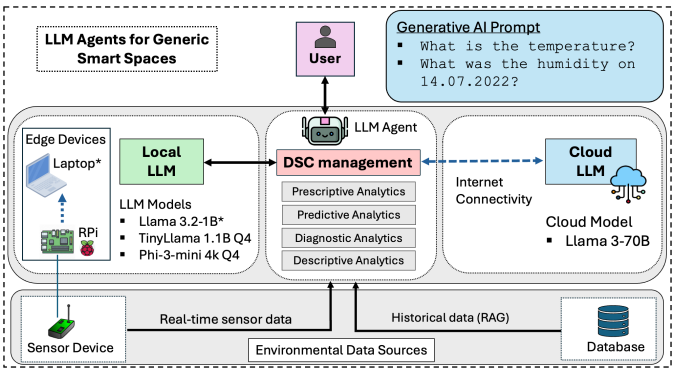


Fig. 1: Hybrid edge-cloud framework that integrates LLM agents for user-centric interactions in a generic smart space.

Unlike prior works that focus only on either compression-based edge deployment [7] or cloud-only architectures [6], we propose a hybrid edge-cloud framework that integrates LLM agents with sensor data and user prompts. Our framework that is shown in Fig. 1 facilitates real-time interaction and decision-making in smart spaces and allows flexible orchestration of LLM services across heterogeneous edge and cloud resources. Specifically, we consider a generic smart space consisting of two edge devices: a generic laptop and a Raspberry Pi 4 (RPi); a commercial cloud platform Groq[1]; a BOSCH BME680 sensor device for real-time measurements; and a two-year historical environmental dataset collected by the same BOSCH sensor family. We select four open-source LLMs for evaluation, comprising two quantized models (TinyLlama and Phi-3 Mini), a small model (Meta Llama 3.2 1B), and a large-scale model (Meta Llama 3.2 70B). While we deploy the quantized models on the RPi, both the small model and the quantized models run locally on the GPU-equipped laptop. We also employ the large-scale model on the Groqs cloud that is suitable for computationally intensive tasks.

[1]https://groq.com/

To evaluate the performance of LLMs, as shown in Fig. 1, we utilize the Device, Service, and Connectivity (DSC) management module (the pink rectangle). This module which is also presented in [9], collects data from both the database and sensors, hosts the LLM agent, interfaces with the LLM prompt for user interaction, and performs data analytics. These analytics include the *Descriptive*, *Diagnostic*, *Predictive*, and *Prescriptive* analytics, ranging from basic to more complex analyses [9]. The goal is to enable LLMs to interpret sensor data and convert it into human-readable language. Using these analytics approaches, we assess the latency and accuracy of responses generated by the LLMs. Our evaluation considers their performance on commonly available edge devices (laptops and RPis) as well as commercial cloud platforms. Performing analytics on edge devices benefits smart spaces by enabling real-time, local operation even during network disruptions. Our evaluations demonstrate the potential of LLMs in transforming generic smart spaces into interactive environments that deliver user-centric services.

II. Experimental Setup

A. Edge-Cloud Hybrid Framework

The experimental setup features an edge-cloud hybrid framework, as illustrated in Fig. 1. This setup includes a cloud server running the Meta Llama 3 70B model [10], hosted on the commercially available GroqCloud platform. GroqCloud is a high-performance AI inference platform that enables the deployment of large-scale models with real-time processing capabilities within our hybrid architecture. At the edge, devices are equipped with locally deployed, lightweight models optimized for real-time inference. These edge devices include a laptop with an Nvidia RTX 2060 GPU (6 GB VRAM) and an Intel i7-10750H CPU, as well as a Raspberry Pi 4, providing flexibility for model deployment and local interaction. In this framework, historical data and real-time measurements are stored locally on edge devices. The workflow of the framework is demonstrated in Fig. 2. In this workflow, upon receiving a user query, a RAG system retrieves relevant data—historical or real-time— to provide contextual information. For instance, it can extract specific parameters (e.g., temperature values), timestamps (e.g., October 21, 2020), or a range of measurement values. If Internet connectivity is available, the retrieved data context, along with the user query, can also be used to prompt the cloud model for inference on the GroqCloud. In scenarios where Internet connectivity is unavailable, the framework can operate offline using either the laptop, the RPi, or both. In addition, when real-time data streaming is enabled (e.g., using a sensor and a RPi), local LLMs can run directly on edge devices without cloud dependency. If an Internet connection is available, edge devices can stream real-time contextual data to GroqCloud for cloud-based LLM inference. These configurations form the foundation of our edge-cloud hybrid framework. The following section details our model deployments on edge devices.

Laptop-based LLM deployment: This setup employs the Meta Llama 3.2 1B model [11], a 4-bit quantized versions

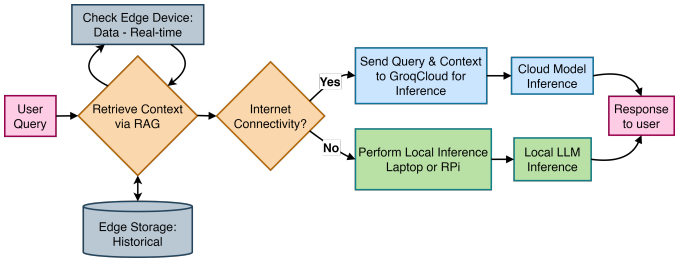


Fig. 2: The workflow of our edge-cloud framework.

(Q4_K_M) of the both TinyLlama model [12], and Microsoft Phi-3 Mini model [13]. Employing multiple lightweight models ensures diversity and evaluates the robustness of our proposed hybrid architecture across different computational constraints inherent to edge computing environments. These selected models prioritize a balance between performance and computational efficiency, enabling real-time inference on edge devices. Additionally, this approach provides flexibility in model selection according to available resources and the required inference accuracy. In this setup, based on the user's query, the laptop device either processes a our generated historical environmental context dataset or receives real-time data streamed from the RPi-equipped BOSCH environmental sensor. In the next subsection II-B, we explain our data strategy for generating the context dataset.

RPi 4-based LLM deployment: This configuration deploys 4-bit quantized versions of the TinyLlama and Microsoft Phi-3 Mini models onto a Raspberry Pi 4. The TinyLlama model consumes 3.17 GB of RAM and requires 0.67 GB of storage space. Similarly, the Phi-3 Mini model, featuring 3.8 billion parameters, occupies 2.2 GB of storage using the Q4_K_M quantization technique. This configuration is specifically designed to facilitate model deployment on resource-constrained hardware. In this deployment scenario, the RPi 4 simultaneously functions as an edge computing platform that uses the historical context dataset, as well as a sensor node that captures real-time environmental data using the BOSCH sensor while performing local inference tasks.

B. Environment Data

Real-time sensor data: To perform environmental monitoring in real time, we use a RPi 4 equipped with a Bosch BME680 sensor, measuring temperature, humidity, pressure, and gas levels at a one-second sampling interval. Sensor data is transmitted to the edge devices through HTTP requests within the local network, facilitating immediate incorporation into the RAG pipeline. The pipeline leverages the pretrained Sentence Transformer model all-MiniLM-L6-v2, enabling the edge-based LLMs to interpret real-time environmental conditions effectively. In the event of an Internet outage or intentionally restricted connectivity, the edge LLM agent remains fully operational, running the local LLM independently. The RAG pipeline continues to function, ensuring that essential services—such as real-time monitoring and user interaction—are maintained. This approach presents the practicality of the

edge-cloud hybrid architecture in delivering reliable, privacy-enabled AI services for smart spaces.

Historical dataset: To evaluate the performance of the hybrid edge-cloud approach, we utilize an open-source dataset of indoor air pollutants collected using a BOSCH low-cost sensor [14]. The dataset comprises 173,468 records gathered from November 17, 2020, to July 14, 2022 (a total of 214 days) from an indoor environment in Pune, India, enabling a comprehensive evaluation under various indoor conditions. NO_2, CO, $PM_{2.5}$, O_3, temperature, humidity, and air pressure, key indoor air quality variables defined by the United States Environmental Protection Agency (EPA) [15]. In addition, the dataset contains NH_3 measurements, which we exclude from our analysis as it is not covered by regulatory guidelines. A notable challenge in the dataset is the presence of infrequent or incomplete daily recordings, a common issue in sensor networks due to factors such as network outages or power failures [16]. To address this and ensure uniform coverage across the observation period, we adopted a daily averaging strategy. This approach aligns with the 24-hour reference periods commonly used in regulatory standards, mitigating the impact of missing data and transient outliers. The resulting dataset consists of 214 days, with each day representing the average value of each measured variable, providing a reliable basis for analyzing long-term indoor air quality trends.

C. Question design for prompt

As depicted in Fig. 1, users interact with the AI agent via natural language prompts, dynamically choosing between cloud and edge-based processing according to computational requirements. The users, based on their intent we ask different forms of questions with different complexity. While some questions may be only the current state of the indoor environment, other questions would include predictions and prescriptions about the future states of the environment. Therefore, to define the complexity levels of questions that require the interpretations of environmental data by LLMs, we apply the four levels of Gartner's analytics [17], which include:

- *Descriptive Analytics* that provides a foundational view of environmental conditions, offering both historical and real-time insights crucial for monitoring and understanding the current state of a smart space.
- *Diagnostic Analytics* that aims to explain the factors underlying changes in sensor readings.
- *Predictive Analytics* that focuses on generating specific numerical forecasts for upcoming periods.
- *Prescriptive Analytics* that aims to recommend specific interventions to improve indoor environment conditions.

Gartner's analytics approach enables us to formulate a wide range of questions, from simple to complex, each requiring varying levels of processing and accuracy from LLMs. Using this approach, we carefully design 15 distinct questions/-queries, tailored to the specific type of analytics the LLMs are expected to perform. For example, the "*Descriptive Analytics*" focuses on simple data retrieval, such as searching for specific values within a dataset. The '*Diagnostic Analytics*" involves comparative analysis of historical sensor data to identify factors contributing to observed changes. The "*Predictive Analytics*" aims to forecast future sensor values based on previously collected data, assessing the model's ability to detect trends and make accurate predictions. The "*Prescriptive Analytics*" evaluates whether the LLM can provide actionable recommendations to enhance indoor environmental quality (e.g., air quality and thermal comfort) based on sensor data. The following blue text box presents examples of the designed questions (Q) and the best model answers (A) (blue text) corresponding to each type of analytics.

Descriptive Analytics

Q: What is the Temperature on 21.11.2020?
A: 29.1

Diagnostic Analytics

Q: From 26.11.2020 to 27.11.2020, list previous and current readings for each sensor parameter and specify if these changes likely impacted the $PM_{2.5}$ level.
A: NH3: 0.82 – 0.69 NO2: 0.12 – 0.14 (No) CO: 4.92 – 4.39 (Yes) PM25: 0.0 – 0.0 (No) Temp: 29.58 – 30.04 (No) Pressure: 943.59 – 945.51 (No) Humidity: 41.06 – 40.78 (No) O3: 7.69 – 5.12 (Yes)

Predictive Analytics

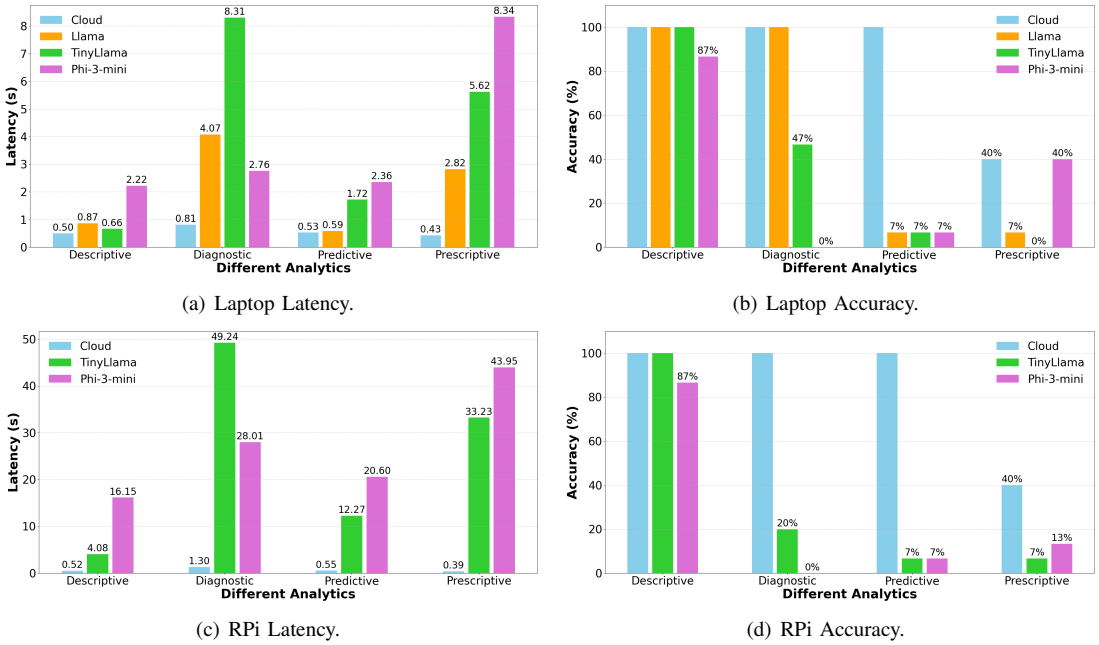
Q: Based on data from 28.11.2020 to 29.11.2020, predict the value of Humidity for the next day.
A: 41.06

Prescriptive Analytics

Q: Between 04.12.2020 and 05.12.2020 analyze the CO value. Based on this value, recommend action to optimize both air quality and thermal comfort.
A: Increase ventilation and consider using air purifiers to optimize air quality, and maintain a comfortable temperature between 22–25°C to balance thermal comfort.

III. Results

Herein, we evaluate the performance of the proposed hybrid edge–cloud system using our 15 designed queries to compare local inference of LLMs on edge devices against cloud-based processing. Our analysis focuses on two key performance metrics: latency and accuracy in handling various queries. Specifically, we assess performance in two scenarios—querying historical environmental data and querying during real-time measurements. Table I summarizes the numerical measurements and performance metrics—including minimum, maximum, mean, and median values for both latency and accuracy. In our analysis, we focus primarily on the mean values to support the visualizations and explanations. Fig. 3 shows a comparative view of edge device performance under local versus cloud deployments, and Fig. 4 illustrates the performance of LLMs when processing real-time queries.



(a) Laptop Latency.

(b) Laptop Accuracy.

(c) RPi Latency.

(d) RPi Accuracy.

Fig. 3: Performance of LLMs deployments on edge devices (laptop and RPi) vs cloud.

A. Performance analysis using historical data

To evaluate performance of LLMs inference on the edge devices and the cloud for the historical data, we use Fig. 3 and the white (upper) part of the Table I.

Descriptive Analytics: The performance of descriptive analytics indicates that the cloud-based model (Llama 3 70B) generally provided the best combination of low latency (mean: Laptop 496 ms, RPi 515 ms) and perfect accuracy (100%). Among the local edge models on the laptop, Llama and TinyLlama also attained 100% accuracy, with average latencies of 867 ms and 663 ms, respectively, while Phi-3-mini recorded a higher mean latency of 2221 ms and a lower accuracy of 86.7%. Even on the resource-constrained RPi, TinyLlama and Phi-3-mini maintained accuracies of 100% and 86.7%, respectively; however, their latencies were substantially longer (mean latency of 4076 ms for TinyLlama and 16152 ms for Phi-3-mini). Notably, TinyLlama demonstrated a median latency of 475 ms on the laptop, outperforming the cloud-based median of 516 ms, yet maintaining 100% accuracy. Therefore, for descriptive analytics within smart spaces, TinyLlama on edge devices such as laptops represents an effective local alternative when there are cloud connectivity or privacy concerns.

Diagnostic Analytics: The cloud-based model (Llama 3 70B) demonstrated the best overall performance (mean latency: 814 ms Laptop, 1301 ms RPi; accuracy: 100%). Edge-based Llama also achieved 100% accuracy with mean latencies of 4073 ms on Laptop. In contrast, TinyLlama showed poor performance, recording a mean latency of 8306 ms on Laptop and 49240 ms on RPi, with accuracy notably reduced to 46.7% and 20%, respectively. The Phi-3-mini, however, it depicted some moderate performance in terms of latency (i.e., 2.76 seconds on laptop), but in terms of accuracy, this model failed to accomplish diagnostic queries on both devices. This highlights the computational constraints faced by local edge devices, especially when executing more complex analytics tasks. In diagnostic analytics, the cloud-based model delivered optimal performance with 100% accuracy and the lowest median latency (Laptop: 808 ms, RPi: 858 ms). Local edge deployments faced substantial latency increases, with Llama providing acceptable accuracy (100%) but much higher median latency (4079 ms). TinyLlama's performance degraded severely in accuracy (46.7%) and exhibited extreme latency on both Laptop (median: 6522 ms) and RPi (median: 56714 ms). Phi-3-mini failed entirely, underscoring computational constraints of these edge devices. Therefore, diagnostic analytic tasks strongly require model deployments on the cloud, which effectively handle the complexity required to explain environmental changes.

Predictive Analytics: The cloud-based model (Llama 3 70B) maintained an optimum accuracy of 100%, presenting a minimum latency (mean latency: 528 ms Laptop, 547 ms RPi). The models including Llama, TinyLlama and Phi-3-mini, on laptop, present the mean latencies of 589 ms, 1720 ms, and 2356 ms, respectively; showing significantly low accuracy of 6.7% across all models. The model deployments on RPi further magnified these limitations, TinyLlama and Phi-3-mini incurring mean latencies of 12267 ms, and 20602 ms, yet with low accuracies of 7%. These results describe that edge-based models failed to understand the trend of the sensor data and make predictions. The predictive analytics, when inferring the models on the cloud environment, depicted a perfect accuracy of 100% with considerably lower latencies of 530 ms and 550 ms on laptop and RPi, respectively. These results clearly highlight the superiority of cloud deployments for

predictive analytics within smart spaces due to their capability to accurately predict trends under strict latency requirements.

Prescriptive Analytics: Prescriptive Analytics posed the greatest challenge for both cloud and local LLMs. This analytics focuses on recommending specific interventions that can improve indoor environmental conditions. In our implementation, to validate the prescriptive analytics, we combine model-based suggestions with the authors' review and a ChatGPT-4o model [6]. According to our measurements, the cloud-based model (Llama 3 70B) and Phi-3-mini on the laptop achieved 40% accuracy in suggesting reasonable interventions with mean latency of 426 ms, and 8336 ms, respectively. The cloud-based model also achieved the same accuracy with roughly the same 393 ms mean latency on RPi; however Phi-3-mini's accuracy dropped to 13% with a considerable high mean latency of 43946 ms. Llama on Laptop, and TinyLLama on both edge devices, with highest accuracy of 6.7%, failed at suggesting reasonable interventions based on sensor data. These results describe the complexity of translating sensor data trends into context-specific guidance without additional fine-tuning or domain-specific knowledge. As a results, the prescriptive analytics are best supported either by cloud-based solutions or local deployments of moderately efficient models (such as Phi-3-mini) on laptops; and by integrating domain-specific knowledge into these models, they can offer sensors data-based specific recommendations in smart spaces.

B. Performance analysis using real-time measurements

In addition to querying the historical dataset, we tested our system with real-time measurements. Earlier, in subsection II-B, we explained the sensor data collections. In our investigations, we also consider the cases that there is no Internet connection, and hence our edge-cloud system operates offline by inferring local models only on the edge devices. For real-time measurement analysis, we consider these scenarios:

- **Scenario 1:** a laptop that functions as the edge device to run LLMs, while a RPi equipped with a BME680 sensor that performs measurements.
- **Scenario 2:** an RPi device that operates as edge and the sensing device, performing both measurements and LLMs inference on the same hardware platform.

For real-time analysis of sensor measurements, as diagnostic, predictive, and prescriptive analytics require historical sensor datasets, therefore, we only evaluate the models while querying for descriptive analytics. While the numerical representation of the results from this analytics is presented in the gray part in Table I, the results are shown visually in Fig. 4.

Scenario 1: In this setup, the RPi device streams real-time data to the laptop via a Wi-Fi connection within a home local area network. The laptop runs Llama, TinyLlama, and Phi-3 Mini offline. When an internet connection is available, the system can also utilize the cloud model (Llama 3 70B) as well. In this scenario, the cloud model achieved the best performance with mean latency of 231 ms, with an accuracy of 100%. The Llama and Phi-3 Mini achieved a mean latency of 552 ms, and 4403 ms, respectively; while both models obtained an accuracy

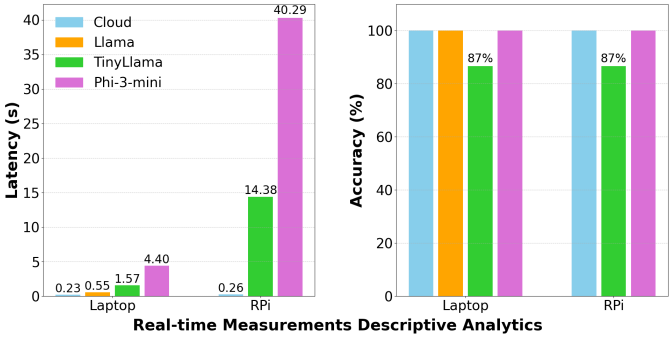


Fig. 4: Performance of the real-time measurements.

of 100%. On the other hand, the TinyLlama model achieved a mean latency equal to 1573 ms, and a slightly lower accuracy of 86.7%. From these results, we conclude that streaming real-time data to a more powerful edge device significantly improves accuracy at the cost of a slight increase in latency. Real-time smart space scenarios are effectively supported by cloud-based solutions. Additionally, for offline operations in smart spaces, this scenario can efficiently support local model deployments (e.g., Llama) on edge devices such as laptops.

Scenario 2: In this setup, the RPi device gathers sensor data and performs local LLM inference. The cloud-based model achieved the best performance, with a mean latency of 257 ms and an accuracy of 100%. However, when running TinyLlama, latencies increased significantly, reaching an average of 14379 ms, while maintaining a high accuracy of 86.7%. Similarly, Phi-3 Mini exhibited a high mean latency of 40287 ms but achieved an accuracy of 100%. These results indicate that employing cloud-based models on the RPi enables real-time retrieval of sensor measurements. In addition, local inference on highly constrained devices such as the RPi is feasible but also this approach presents substantial performance trade-offs.

C. Summary of our findings

With historical data, the cloud model hit 100% accuracy and lowest latency on descriptive queries, while TinyLlama on a laptop was the best offline substitute. In diagnostic tests, the cloud model and laptop-based Llama were top performers; other edge models faltered, especially on the RPi. Predictive analytics succeeded only in the cloud, indicating local models need further training. Prescriptive tasks reached 40% accuracy with cloud and laptop-based Phi-3-Mini, but edge-deployed TinyLlama and Llama failed to give useful actions. For real-time measurements on a laptop, the cloud model again led (high accuracy, low latency); Llama stayed competitive, Phi-3-Mini was accurate but slower, and TinyLlama showed moderate accuracy with higher delay. On the RPi, Phi-3-Mini remained accurate but slow, whereas TinyLlama was quicker yet less accurate. Therefore, smart-space analytics benefit from a task-aware split: complex or predictive workloads to the cloud; descriptive or latency-critical tasks to capable edge devices (e.g., laptops). This strategy boosts performance, privacy, and resilience, keeping essentials running even offline.

TABLE I: Performance of LLMs in conducting the analytics. The white part the table presents the results when using the historical context dataset, and the gray part presents the real-time measurements. The latency (in millisecond) of models are shown by the statistics on the left side of the table, and the accuracy of the models are presented on the right side of the table.

Analysis	Model	Min		Max		Mean		Median		Accuracy	
		Laptop	RPi	Laptop	RPi	Laptop	RPi	Laptop	RPi	Laptop	RPi
Descriptive	**Cloud**	285.45	483.66	572.60	562.24	496.36	515.80	516.34	509.86	100.0	100.0
	Llama	402.75	None	2724.50	None	867.14	None	572.03	None	100.0	None
	TinyLlama	401.89	1822.99	1625.05	12538.12	662.95	4076.03	474.54	2879.39	100.0	100.0
	Phi-3-mini	876.36	4434.06	6328.69	56982.78	2220.60	16152.28	1469.18	11415.80	86.7	86.7
Diagnostic	**Cloud**	543.01	785.17	1317.68	7128.16	814.24	1301.52	807.86	857.80	100.0	100.0
	Llama	3862.37	None	4285.09	None	4072.95	None	4078.58	None	100.0	None
	TinyLlama	3238.57	25384.14	28337.64	83174.97	8306.07	49240.82	6522.09	56713.60	46.7	20.0
	Phi-3-mini	582.35	4567.16	11494.56	121749.55	2758.25	28009.22	662.09	5200.10	0.0	0.0
Predictive	**Cloud**	484.66	513.97	746.61	597.91	528.07	547.47	507.18	544.56	100.0	100.0
	Llama	475.89	None	1143.06	None	588.89	None	553.58	None	6.7	None
	TinyLlama	639.94	2449.89	3749.93	32888.40	1720.30	12267.10	1343.29	8087.75	6.7	6.7
	Phi-3-mini	450.65	3176.29	10040.75	84822.49	2356.48	20601.72	501.34	3905.35	6.7	6.7
Prescriptive	**Cloud**	263.30	279.63	792.46	536.90	426.25	392.83	405.68	380.67	40.0	40.0
	Llama	1800.61	None	6158.80	None	2821.86	None	2285.21	None	6.7	None
	TinyLlama	940.64	5940.91	14925.00	58553.49	5623.23	33232.13	4210.51	33572.90	0.0	6.7
	Phi-3-mini	998.31	7542.18	25264.96	176681.28	8335.91	43945.80	6091.33	14904.78	40.0	13.3
Descriptive	**Cloud**	180.85	173.53	381.51	695.01	231.37	257.15	213.34	236.55	100.0	100.0
	Llama	424.77	None	817.15	None	551.99	None	521.68	None	100.0	None
	TinyLlama	936.47	8886.38	3296.58	52412.93	1572.70	14379.17	1202.00	9746.10	86.7	86.7
	Phi-3-mini	3890.75	33778.67	5571.97	49358.23	4402.72	40286.56	4216.22	40371.12	100.0	100.0

IV. Conclusion

We presented a hybrid edge-cloud framework for deploying LLMs in generic smart spaces, using widely available edge devices, alongside commercially available cloud platforms. We evaluated the strengths and limitations of both edge-based and cloud-based implementations. Our experimental results showed that cloud-based LLM deployments achieve superior accuracy and reduced latency in handling complex analytical tasks, such as predictive and prescriptive analytics, making them highly effective for processing large datasets. Meanwhile, our edge-based LLM deployments enabled local operation despite hardware constraints, making them well-suited for real-time data processing. While resource-constrained edge devices perform efficiently for simpler tasks, such as descriptive analytics, they also provide a viable solution for maintaining data privacy and ensuring functionality during offline operation. Our findings highlight the complementary roles of edge and cloud LLM deployments, suggesting implementing hybrid frameworks that optimize performance while addressing privacy in AI-driven smart spaces. In future, we will explore advanced hybrid architectures that integrates domain-specific knowledge for better predictive and prescriptive analytics.

Acknowledgment

This work is supported by the Ministry of National Education of the Republic of Türkiye, and by the Research Council of Finland with grant number 362594, and by the Jane and Aatos Erkko Foundation EVIL-AI project.

References

[1] A. Saleh, P. K. Donta, R. Morabito, N. H. Motlagh, and L. Lovén, "Follow-Me AI: Energy-efficient user interaction with smart environments," *IEEE Pervasive Computing*, 2025.

[2] A. Varol, N. H. Motlagh, M. Leino, S. Tarkoma, and J. Virkki, "Creation of ai-driven smart spaces for enhanced indoor environments–a survey," *arXiv preprint arXiv:2412.14708*, 2024.

[3] O. Friha, M. A. Ferrag, B. Kantarci, B. Cakmak, A. Ozgun, and N. Ghoualmi-Zine, "Llm-based edge intelligence: A comprehensive survey on architectures, applications, security and trustworthiness," *IEEE Open Journal of the Communications Society*, 2024.

[4] S. Gallo, F. Paternò, and A. Malizia, "A conversational agent for creating automations exploiting large language models," *Personal and Ubiquitous Computing*, vol. 28, no. 6, pp. 931–946, 2024.

[5] T. An, Y. Zhou, H. Zou, and J. Yang, "Iot-llm: Enhancing real-world iot task reasoning with large language models," *arXiv preprint arXiv:2410.02429*, 2024.

[6] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford *et al.*, "Gpt-4o system card," *arXiv preprint arXiv:2410.21276*, 2024.

[7] G. Qu, Q. Chen, W. Wei, Z. Lin, X. Chen, and K. Huang, "Mobile edge intelligence for large language models: A contemporary survey," *IEEE Communications Surveys & Tutorials*, 2025.

[8] M. Zhang, X. Shen, J. Cao, Z. Cui, and S. Jiang, "Edgeshard: Efficient llm inference via collaborative edge computing," *IEEE Internet of Things Journal*, 2024.

[9] N. H. Motlagh, M. A. Zaidan, L. Lovén, P. L. Fung, T. Hänninen, R. Morabito, P. Nurmi, and S. Tarkoma, "Digital twins for smart spaces—beyond iot analytics," *IEEE internet of things journal*, vol. 11, no. 1, pp. 573–583, 2023.

[10] AI@Meta, "Llama 3 model card," 2024. [Online]. Available: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md

[11] Meta, "Llama-3.2-1B-Instruct," 2024, accessed: 2025-03-06. [Online]. Available: https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct

[12] TheBloke, "TinyLlama-1.1B-Chat-v1.0-GGUF," 2024, accessed: 2025-03-06. [Online]. Available: https://huggingface.co/TheBloke/TinyLlama-1.1B-Chat-v1.0-GGUF

[13] Microsoft, "Phi-3 Mini 4K Instruct Model," 2024, accessed: 2025-03-06. [Online]. Available: https://huggingface.co/microsoft/Phi-3-mini-4k-instruct-gguf

[14] S. Sonawani and K. Patil, "Dataset of indoor air pollutants using low-cost sensors," *Mendeley Data*, vol. 1, 2022.

[15] United States Environmental Protection Agency, "Indoor pollutants and sources," 2024, accessed: 2025-03-06. [Online]. Available: https://www.epa.gov/indoor-air-quality-iaq/indoor-pollutants-and-sources

[16] M. A. Zaidan, N. H. Motlagh, B. E. Boor, D. Lu, P. Nurmi, T. Petäjä, A. Ding, M. Kulmala, S. Tarkoma, and T. Hussein, "Virtual sensors: toward high-resolution air pollution monitoring using ai and iot," *IEEE Internet of Things Magazine*, vol. 6, no. 1, pp. 76–81, 2023.

[17] M. Maoz, "How it should deepen big data analysis to support customer-centricity," *Gartner G00248980*, 2013.