On-Device LLM for Context-Aware Wi-Fi Roaming

Ju-Hyung Lee[1] **Yanqing Lu**[1 2] **Klaus Doppler**[1]

Abstract

Roaming in Wireless LAN (Wi-Fi) is a critical yet challenging task for maintaining seamless connectivity in dynamic mobile environments. Conventional threshold-based or heuristic schemes often fail, leading to either *sticky* or *excessive* handovers. We introduce the first *cross-layer* use of an on-device large language model (LLM): high-level reasoning in the application layer that issues real-time actions executed in the PHY/MAC stack. The LLM addresses two tasks: (*i*) *context-aware AP selection*, where structured prompts fuse environmental cues (e.g., location, time) to choose the best BSSID; and (*ii*) *dynamic threshold adjustment*, where the model adaptively decides *when* to roam. To satisfy the tight latency and resource budgets of edge hardware, we apply a suite of optimizations—chain-of-thought prompting, parameter-efficient fine-tuning, and quantization. Experiments on indoor and outdoor datasets show that our approach surpasses legacy heuristics and DRL baselines, achieving a strong balance between roaming stability and signal quality. These findings underscore the promise of application-layer LLM reasoning for lower-layer wireless control in future edge systems.

1. Introduction

Wi-Fi roaming is a critical operation for maintaining seamless wireless connectivity as users move through physical environments. Traditionally, roaming logic has assumed relatively stable topologies—stationary access points (APs) and predictable client mobility. However, emerging scenarios are rapidly redefining this assumption. Automotive Wi-Fi and device-to-device (D2D) "mobile-AP" scenarios

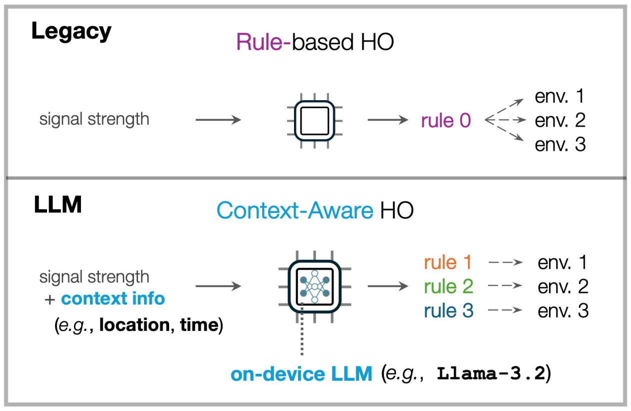


Figure 1. Cross-layer control via on-device LLM: Rule-based handover (legacy) vs. on-device LLM context-aware handover.

all feature mobility on *both* the client and AP sides, creating highly dynamic topologies that demand instantaneous, context-aware decisions at the wireless edge.

Despite decades of refinement, current roaming mechanisms still rely heavily on threshold-based triggers—such as scanning when RSSI drops below -70 dBm—and static handover logic that fails to adapt to varying conditions. Such strategies often result in either *sticky handovers* (where devices cling to weak links) or *excessive handovers* (leading to instability and overhead). These issues are exacerbated in modern use cases, where wireless conditions change rapidly across space and time.

Making intelligent roaming decisions under these constraints is inherently difficult. The system must reason over noisy, high-dimensional signals such as RSSI patterns, user location, time of day, and device state—all under tight latency budgets (typically within 10–100 ms). Prior approaches based on supervised learning or deep reinforcement learning (DRL) have shown promise but suffer from limited generalization, requiring task-specific retraining and extensive engineering to adapt to new environments.

In this work, we propose a new paradigm: using large language model (LLM) as adaptive roaming agents deployed directly at the wireless edge. While LLMs have recently been explored for cloud-level network management (Wu et al., 2024), ***this work is the first to deploy an LLM* on-device *for real-time control at the lower layers of a wire-***

[1]Nokia Technologies, Sunnyvale, CA, USA [2]Department of Computer Science, University of Southern California, Los Angeles, CA, USA. Correspondence to: Ju-Hyung Lee <juhyung.lee@nokia.com>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

less system, specifically at the PHY/MAC level. By leveraging the LLM's in-context reasoning capabilities, we enable it to interpret structured prompts that encode real-time situational context—such as RSSI, location, and time—and make intelligent roaming decisions locally, without external coordination or retraining.

Problem scope. We study two concrete tasks:

T1) *Context-aware AP selection* (§3): choose the optimal BSSID given current context;

T2) *Dynamic threshold adjustment* (§4): adaptively decide *when* to trigger roaming.

Contributions.

- **Cross-layer Wireless Control via On-Device LLM.** We demonstrate the first on-device LLM that reasons in the application layer while issuing real-time actions in the PHY/MAC stack.
- **Edge-efficient LLM pipeline.** Through prompt design, post-training, and quantization, we trim memory and compute cost, pushing inference toward real-time with only marginal accuracy loss.
- **Comprehensive evaluation.** Indoor and outdoor experiments show our approach outperforms legacy and DRL baselines, striking a strong trade-off between roaming stability and link quality.
- **Practical insights & Real-world demo.** We distill practical guidelines for deploying LLMs as edge-level wireless controllers (§5) and validate our approach through practical demonstrations (§A)[1].

We begin in Section 2 by reviewing the fundamentals of Wi-Fi roaming and describing how LLMs can act as real-time, context-aware decision-making agents in wireless systems.

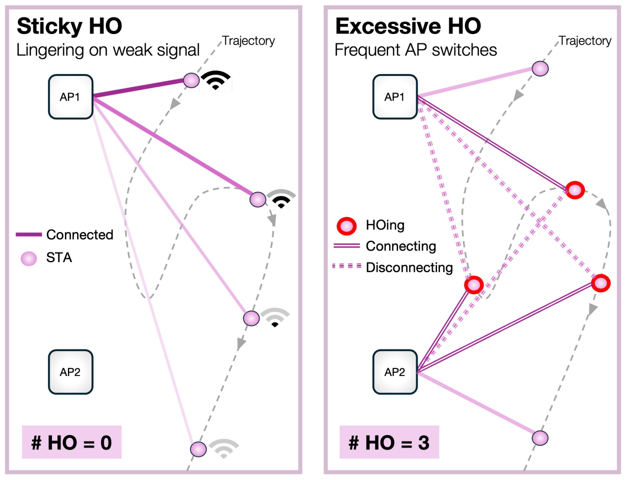
2. Background

2.1. Wi-Fi Roaming and Mobility Management

In Wi-Fi networks, *roaming* refers to the process by which a client device (*e.g.*, a smartphone or laptop) switches its connection from one AP to another as the user moves. These APs typically belong to the same extended service set (ESS), providing overlapping coverage areas. For instance, as a user walks through a building, the device must determine when it should disconnect from the current AP and connect to another AP to maintain a strong wireless connection and avoid dropped connections or degraded signal quality.

Roaming is a Layer 2 operation governed by the `IEEE 802.11` standard, playing a crucial role in ensuring seamless connectivity across dynamic environments such as office buildings, university campuses, and public transit stations.

Most client devices continuously monitor the received signal strength indicator (RSSI) of their currently associated AP. When the RSSI falls below a predefined threshold, known as `scanRSSI` (typically set around -70 dBm), the device initiates a roaming procedure. After crossing this threshold, the device actively scans for candidate APs, evaluating them based on factors such as RSSI, channel congestion, and PHY-layer capabilities. It then selects a new AP if this AP offers significantly better link quality, commonly defined by a relative RSSI improvement threshold (*e.g.*, at least $+8$ dB during active data transmission or $+12$ dB during idle periods).



(a) Sticky handover. (b) Excessive handover.

Figure 2. Common failure modes of rule-based Wi-Fi roaming.

Despite their simplicity, fixed RSSI thresholds and rule-based roaming logic frequently lead to suboptimal outcomes. As shown in Fig. 2, poorly tuned thresholds and static decision rules often produce two problematic scenarios:

- **Sticky handovers:** The device remains connected to a weakening AP, causing deteriorating throughput, increased latency, and poor user experience.
- **Excessive handovers:** The device frequently initiates scanning and switches between APs, causing disruptions in the connection, increased signaling overhead, lowered throughput, and degraded user experience.

Effectively managing roaming thus requires balancing two

[1]A full demonstration video and inference code are available at github.com/abman23/on-device-llm-wifi-roaming.

Aspect	Conventional ML / DRL	LLMs as Agents
Input Format	Fixed feature vectors	Flexible, multimodal prompts
Adaptability	Retraining required for new settings	Zero-/few-shot generalization
Reasoning Style	Implicit via model weights	Explicit, interpretable (*e.g.*, CoT)
Context Integration	Hand-crafted, limited scope	Natural support for rich context
Deployment	Efficient but narrow	Powerful, yet compute-heavy

Table 1. Comparison: Conventional AI vs. LLM as Decision-Making Agents

conflicting objectives: maintaining high signal quality while minimizing handover frequency. To quantitatively evaluate roaming strategies, we consider two key metrics:

- **AvgRSSI**: The average RSSI experienced by the client (Station, STA) throughout the test period, reflecting the overall connection quality (higher is better).
- **#HO**: The total number of handovers triggered during the evaluation period, indicating roaming stability or the overhead associated with frequent AP switching (lower is better).

A well-designed roaming policy must intelligently and dynamically balance these two metrics based on real-time environmental context and user requirements.

2.2. LLMs as Decision-Making Agents

LLMs have shown impressive generalization across diverse tasks in language, vision, reasoning, and planning. More recently, they have been explored as *decision-making agents* capable of operating in structured environments such as robotics and human-computer interaction (Huang et al., 2022; Irpan et al., 2022)—domains that demand not only prediction but also adaptive control.

In wireless systems, especially at the lower protocol layers (*e.g.*, PHY/MAC), such adaptive capabilities remain underexplored. Traditional methods in tasks like roaming, scheduling, and interference management typically rely on fixed heuristics or conventional machine learning models. Reinforcement learning approaches (*e.g.*, PPO) have been applied in wireless contexts; however, these methods often demand specialized model designs, dense reward signals, and substantial retraining efforts when deployed in new environments or tasks (Lacava et al., 2024; Lee et al., 2024; Wilhelmi et al., 2024).

LLMs offer a fundamentally different paradigm. They enable *in-context learning*, allowing the model to make structured decisions based on prompt-based inputs without task-specific retraining. This adaptability makes LLMs well-suited for dynamic wireless settings, where behavior must generalize across diverse users, locations, and temporal conditions. LLMs can naturally process heterogeneous input modalities—such as signal strength, location, time of day, and battery state—using structured prompts, and can output decisions accompanied by interpretable reasoning traces (*e.g.*, via chain-of-thought prompting).

These capabilities position LLMs not simply as prediction or classification models but as *context-aware agents* capable of integrating high-dimensional contextual information for adaptive wireless control. Although LLM inference occurs in the application layer, their outputs directly influence PHY/MAC layer parameters—such as selecting a target BSSID or adjusting roaming thresholds. This represents an emerging trend of deploying *application-layer AI for lower-layer wireless control*. Table 1 highlights key differences between conventional AI methods and LLM-based approaches in this scenario.

Despite their promise, deploying LLMs for real-time wireless control introduces practical challenges, particularly on edge devices. Strict latency constraints (typically within 100 ms) combined with limited computational and memory resources pose significant hurdles (Xu et al., 2024). In this work, we overcome these constraints through a combination of model-level optimizations (quantization and parameter-efficient fine-tuning) and task-level adaptations designed to minimize inference overhead. These strategies enable the deployment of adaptive, context-aware LLMs capable of efficient, real-time operation directly on-device.

3. Task (1): Context-Aware AP Choice by LLM

Our first objective is to determine whether a LLM can improve Wi-Fi roaming by selecting the optimal *basic service set identifier* (BSSID) using real-time context (*e.g.*, device location and time).

3.1. Problem Statement: Best BSSID Selection

When multiple APs are available, the device must select BSSID to roam to. The "best" node is typically the one offering the most favorable channel connection (*e.g.*, , strong RSSI). In dynamic environments, the optimal choice can depend not only on RSSI, but also other context information. For example, site-specific information might hint at which AP has better coverage, while time of day and user mobility might tells the network congestion patterns. Our task is to predict, at any given moment, which AP the client should associate with to maximize handover performance.

Challenging Scenarios. As shown in Figure 2, traditional roaming mechanisms often exhibit two critical shortcomings: *sticky handovers*, where the device remains connected to a suboptimal AP despite significant signal degradation (left panel, zero handovers), and *frequent handovers*, where the device constantly re-associates with different APs (center panel, multiple handovers), leading to instability and increased overhead. These scenarios degrade user experience via dropped connections, lower throughput, and higher latency.

3.2. Approach: LLM-based Decision with Contextual Information

We employ a LLM to evaluate available APs based on rich situational data. Specifically, the model ingests inputs such as the current and neighboring AP RSSI, location, time of day, and historical throughput. These contextual features may be encoded in a structured prompt or feature vector that the LLM can interpret. Unlike conventional AI solutions, which often require significant re-training or custom engineering to handle new context signals, an LLM can flexibly process additional inputs by extending the prompt format. The LLM then predicts which BSSID the STA should join next (or remain connected to, if already optimal). By going beyond a simple "highest RSSI wins" legacy, the model can learn nuanced rules (for instance, staying on a slightly weaker AP if it provides better backhaul, or deferring handover if a signal dip is only transient), thereby adapting more effectively to diverse and evolving environments.

Prompting Engineering. We adopt *chain-of-thought* (CoT) (Wei et al., 2022) prompting and examine how few-shot examples affect the LLM's decision-making. Specifically, we provide the model with situational context (current RSSI, neighboring AP RSSIs, user location, time of day, battery state) and a small number of labeled examples—ranging from zero to five "shots." In the CoT variant, each example includes a concise reasoning trace before the final decision, whereas the non-CoT variant supplies only the final label or action.

While CoT improves zero-/few-shot reasoning, we still need domain adaptation via supervised fine-tuning. Then, to align the model's decisions with user preferences, we apply *direct preference optimization* (DPO) (Rafailov et al., 2023).

Fine-Tuning. We fine-tune the pre-trained LLM on real Wi-Fi roaming log data. We adopt LoRA (Hu et al., 2022) for *parameter-efficient fine-tuning* (PEFT). Using LoRA, we inject small adaptation parameters and even fine-tune on a quantized model to save memory. This adaptation trains the LLM to understand Wi-Fi-specific cues (e.g., what RSSI patterns precede a disconnection) and to output the best AP choice accordingly. The lightweight nature of LoRA fine-tuning allows us to iterate and improve the model without needing enormous compute resources.

Preference-Based Learning. We further refine the model through preference optimization. For instance, logs or user feedback indicating which handover decisions yield better outcomes (e.g., higher throughput, fewer drops) are treated as preferred. The LLM is then optimized to prioritize these decisions via DPO, which directly tunes the model toward favored behaviors. During this process, the LLM explores candidate AP choices, scores them with a learned function, and iteratively updates its policy to favor those leading to better results. The end result is an LLM that not only produces valid AP selections but also aligns with real-world performance objectives.

3.3. Evaluation

We evaluate the Task (1): Context-Aware AP Choice by comparing our LLM-based approach against several baseline roaming strategies in a variety of Wi-Fi roaming scenarios. To evaluate roaming behavior, we use two key metrics:

$$\mathit{AvgRSSI} = \frac{1}{T}\sum_{t=1}^{T} \mathrm{RSSI}_t, \tag{1}$$

$$\#\,\mathit{HO} = \sum_{t=2}^{T} \mathbb{I}[\mathrm{BSSID}_t \neq \mathrm{BSSID}_{t-1}], \tag{2}$$

Here, T denotes the number of time steps, RSSI_t is the received signal strength at time t, and BSSID_t is the AP the device is associated with. ***AvgRSSI*** reflects connection quality, while ***# HO*** captures roaming stability. These two objectives often conflict: reducing handovers may hurt signal quality, and maximizing RSSI may cause unnecessary roaming. A well-designed roaming policy must intelligently balance these competing goals based on context.

The baselines for comparison are as follows:

- Heuristic: randomly selects an AP when RSSI falls below `scanRSSI`, triggering a roam.
- Legacy: chooses the AP with the highest RSSI when RSSI is below `scanRSSI`, thereby triggering a roam.
- opt-HO & opt-RSSI: opt-HO selects the AP that minimizes *# HO*, while opt-RSSI chooses the AP that maximizes *AvgRSSI*, over the test sequence. Both methods achieve global optimality via exhaustive search.

- PPO (Schulman et al., 2017): is a DRL agent trained for AP selection that takes Wi-Fi measurements and contextual information as input, then outputs the target AP for roaming.
- LLM: is our proposed LLM-based method, which leverages context-aware prompting to determine the optimal AP.

Detailed experimental setups and parameter settings are provided in Table 7 in Appendix C.

3.4. Experiments

Effect of Prompt Engineering. Figure 3 compares CoT prompting with a simpler, non-CoT approach across 0-, 1-, and 5-shot scenarios, highlighting three main findings: *i)* CoT consistently reduces handover count, indicating fewer unnecessary switches; *ii)* CoT maintains a stronger average signal (by about 1–2 dB); and *iii)* CoT begins with a lower error rate at 0-shot, although non-CoT catches or surpasses it at 5-shot. Overall, CoT proves especially beneficial in few-shot contexts, whereas non-CoT capitalizes more effectively on additional examples.

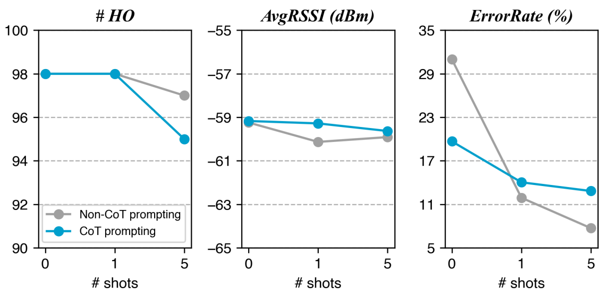


Figure 3. Impact of prompt engineering.

Effect of Post-Training. To reduce computational overhead without compromising accuracy, we adopt PEFT via LoRA. As shown in Table 2, with a suitable configuration (*e.g.*, a learning rate of 2×10^{-4}, batch size of 2, and LoRA rank of 128), the model achieves about 85% accuracy to test set for opt-HO-global optimal (exhaustive) results for # of roaming, while using only about 20% of VRAM and 30% of the GPU-hours required for full training. This efficiency–accuracy balance makes LoRA an attractive choice for computational heavy LLM fine-tuning, so that is used for our post-training.

Table 3 compares different post-training methods. While supervised fine-tuning (SFT) improves the average RSSI, it does not reduce handovers or the error rate. By contrast, combining SFT with DPO lowers the error rate significantly (to 12.66%), suggesting a more balanced outcome. Odds ratio preference optimization (ORPO) (Hong et al., 2024) further cuts handovers but drives up the error rate to 33%, underscoring a trade-off. Nevertheless, in scenarios prioritizing fewer roam events over higher error tolerance (*e.g.*, when a fallback mechanism is available), ORPO may still be advantageous.

Learning Rate	Batch Size	Rank (LoRA)	Quantization	*Accuracy* (%)
1e-5	1	32	✗	57.89
2e-4	1	32	✓	68.42
2e-4	1	32	✗	69.47
2e-4	2	32	✗	78.95
2e-4	1	128	✗	78.95
2e-4	2	128	✗	85.26

Table 2. Performance across various PEFT configurations. Accuracy is the percentage of LLM-selected APs matching test labels generated by opt-HO, which minimizes handovers over the time sequence (Base model: `Llama3.1`-8B (Grattafiori et al., 2024)).

Method	*# HO*	*AvgRSSI* (dBm)	*ErrorRate* (%)
No FT	35.5	-56.48	22.83
SFT	35.5	-55.91	22.83
SFT+DPO	34.5	-56.53	12.66
ORPO	33.0	-55.91	33.00

Table 3. Impact of post-training methods. *ErrorRate* denotes the fraction of invalid AP selections (*i.e.*, when the LLM chooses an AP that is unavailable or has an RSSI below `scanRSSI`). The base model used is `Llama3.1`-8B (Grattafiori et al., 2024).

Comparison: Legacy vs. DRL vs. LLM. We next evaluate our LLM approach against textsfLegacy, an offline DRL method based on proximal policy optimization (PPO), and two global optimal results that minimize handovers (opt-HO) or maximize signal strength (opt-RSSI). Figure 4 illustrates the trade-offs between handover counts (left) and received signal strength (right) for each strategy. The LLM (93 handovers) outperforms Heuristic (106), Legacy (100), and PPO (107) in reducing unnecessary roaming, though it does not reach the minimal handover level of opt-HO (57). Meanwhile, it preserves a higher average signal ($-58.58\,\mathrm{dBm}$) than Heuristic (-63.25), PPO (-59.67), or opt-HO (-63.81), and comes reasonably close to the Legacy method (-58.28) and opt-RSSI (-55.98).

Overall, these results demonstrate that each extreme heuristic sacrifices one metric to excel at the other: opt-HO drastically lowers handovers but offers weak signal quality, whereas opt-RSSI achieves excellent signal strength at the cost of excessive handovers. By contrast, the LLM balances both goals, achieving fewer handovers than conventional baselines and maintaining a robust connection, highlighting the benefits of a context-aware decision-making framework.

3.5. Limitations

For Task (1) BSSID Selection, the STA must make roaming decisions within 10–100 ms after a scan to meet near-real-

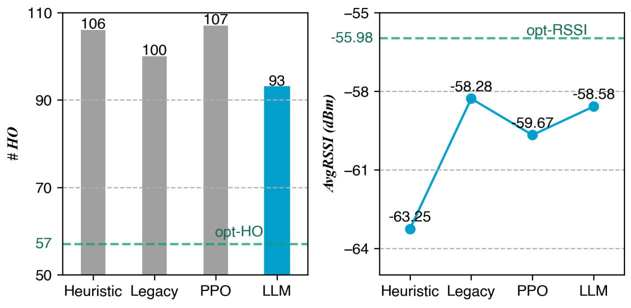


Figure 4. Comparison of ***# HO*** (left) and ***AvgRSSI*** (right) for each method for best BSSID selection (Task 1).

time (near-RT) requirements. However, as shown in Fig. 5, current LLMs exhibit inference times on the order of seconds, far exceeding the required latency. Furthermore, frequent inferences add to computational overhead and power consumption.

Thus, although LLMs provide powerful in-context learning, their current latency makes them more suitable for tasks that require less frequent, non-real-time inference in practical on-device deployments.

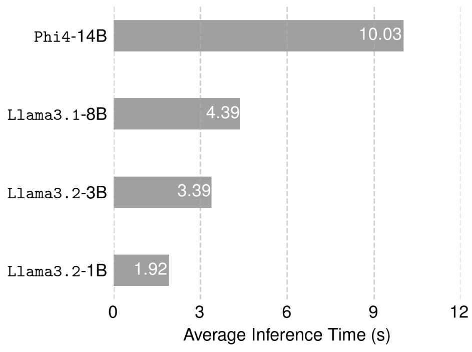


Figure 5. Average LLM inference time (A100 GPU).

4. Task (2): Online Roaming Optimization by On-Device LLM

Having established that an LLM can effectively choose APs, we now adapt it to the challenge of deciding *when* to roam in on-device setup. Our second objective is to evaluate whether an on-device LLM can enhance Wi-Fi roaming by dynamically adjusting the roaming threshold in on-device setup. This task requires less frequent and time-critical inference, making it well-suited for practical on-device deployment.

4.1. Problem Statement: Dynamic Threshold Selection

A major challenge in Wi-Fi roaming is determining *when* to initiate handover. This is typically controlled by a fixed RSSI threshold (`scanRSSI`), below which STA begins scanning for alternative APs. However, as previously discussed, a single static value can lead to two undesirable outcomes: setting it too high causes overly frequent scans and roaming (wasting energy and risking ping-pong effects), whereas setting it too low risks lingering on a deteriorating link until performance degrades or the connection drops. Building on the context-aware approach in Section 3, we now aim to dynamically select the threshold in real time, leveraging an *on-device LLM*.

Yet deploying a large LLM directly on mobile or edge platforms is challenging due to hardware constraints such as limited memory, battery capacity, and compute power. Accordingly, one needs to find a way to reduce the LLM's computation overhead while preserving strong performance in threshold-selection decisions.

4.2. Approach: Adaptive Threshold via On-Device LLM

Cross-Layer Operation. On-Device LLM operates within the application layer but directly controls MAC-layer parameters, specifically the roaming threshold. This illustrates a concrete example of *application-layer AI for lower-layer wireless control*, where high-level reasoning guides low-level network decisions.

Model Selection and Quantization. Deploying large models on resource-constrained edge devices requires careful consideration of model size and computational resources. To ensure practical lower-layer operation, we select a model size that optimally balances inference speed with decision accuracy. We further reduce computational overhead by employing the `Q2_K` quantization scheme. Specifically, `Q2_K` compresses weights into groups of 16, encoding each weight with 2-bit precision and using one shared 4-bit scale and offset per block. This compact encoding needs only 2.56 bits per parameter—about 8× fewer bits than `FP16`—while staying fully compatible with the `GGUF` model format supported by `llama.cpp`.

Task-Oriented Optimization. Since Task (2) is focused specifically on determining *when* to initiate roaming, rather than choosing a specific AP, our optimization objective is explicitly designed around dynamic threshold adjustment. We carefully tune the frequency at which the LLM updates roaming thresholds, enabling it to react quickly to significant RSSI fluctuations while minimizing computational and energy overhead.

4.3. Experiments

Unless otherwise noted, we evaluate this approach under the same general setup, baselines, and metrics used for Task (1) in Sec. 3.3. Detailed experimental setups and parameter settings are provided in Table 7 in Appendix C.

Effect of Quantization. Table 4 compares the impact of various quantization schemes (*e.g.*, `Q2_K`, `Q3_K_M`, `Q4_K_M`) on model size and roaming performance. Quantization significantly reduces the model's footprint from 8.5,GB (`Q8_0`) down to as low as 3.2,GB (`Q2_K`), while maintaining comparable performance metrics (approximately ≈ 138–145 handovers and around ≈ −58,dBm RSSI). Given these results, we select `Q2_K` quantization for subsequent experiments, as it provides substantial resource savings with negligible loss in decision accuracy.

Quant. Method	***# HO***	***AvgRSSI*** (dBm)	**Model Size** (GB)
✗	143	-57.98	16
`Q8_0`	140	-58.62	8.5
`Q6_K`	139	-58.63	6.6
`Q5_K_M`	144	-58.26	5.7
`Q4_K_M`	145	-58.80	4.9
`Q3_K_M`	138	-58.89	4.0
`Q2_K`	138	-58.85	3.2

Table 4. Comparison of quantization schemes for the `Llama3.1-8B` model in terms of roaming decision and memory footprint (Grattafiori et al., 2024).

The adoption of `Q2_K` reduces the model size by approximately 2.7×, from 8.5 GB with `Q8_0` quantization to 3.2 GB, enabling efficient operation within the 16,GB unified memory capacity of the Apple M-series MacBook used for our on-device experiments.

LLM Model Comparison. Table 5 further compares models ranging from 1B to 14B parameters. Larger variants (*e.g.*, `Phi4`-14B (Abdin et al., 2024)) provide slightly higher RSSI (around −57.9 dBm) but yield more handovers (156) and longer inference times (up to 46.9 s). Smaller models (*e.g.*, 1B) drastically reduce inference delays (4.4 s) and handovers (124) but at the cost of a weaker average signal (−60.07 dBm). Models in the 3–8B range often offer a balanced trade-off between speed and roaming performance.

How Often Should the LLM Adjust the Threshold?. We next vary the frequency of roaming threshold adjustments (from 10 s to 300 s). Decreasing this interval to 10 s increases the number of roaming (about 147) while boosting average RSSI (to −57.86 dBm). Increasing the interval to 300 s cuts roamings (down to 119) but weakens the average RSSI (to −59.94 dBm). A 30 s interval emerges as a practical middle ground, balancing roaming stability, signal quality, and computational overhead.

Model	***# HO***	***AvgRSSI*** (dBm)	**Model Size** (GB)
`Llama3.2-1B`	124	-60.07	0.6
`Llama3.2-3B`	150	-58.47	1.4
`Llama3.1-8B`	138	-58.85	3.2
`Phi4`-14B	156	-57.90	5.5

Table 5. Comparing LLMs for dynamic threshold selection (Task 2).

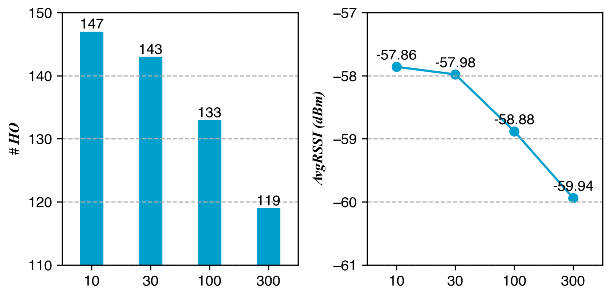


Figure 6. Frequency of threshold adjustments and its effect on performance.

Comparison on *On-Device*: Legacy vs. LLM. Finally, we compare our on-device LLM against four fixed thresholds (−50, −60, −70, −80 dBm) in on-device setup. As illustrated in Fig. 7, each static threshold performs best under specific conditions yet fails to generalize. For instance, −50 dBm consistently achieves a strong average RSSI but induces excessive handovers; −80 dBm cuts handover frequency but suffers from weaker connectivity. Intermediate thresholds like −60 or −70 dBm offer varied performance trade-offs, often performing well in outdoor environments but less consistently indoors.

In contrast, the proposed context-aware on-device LLM dynamically adjust its threshold to local context information, enabling it to maintain balanced RSSI performance and moderate handover frequency across diverse scenarios. While it does not always achieve the absolute best performance on every single metric, the LLM notably delivers robust, generalized performance without manual tuning. This underscores the practical value and effectiveness of cross-layer, adaptive threshold optimization using on-device LLMs for real-world Wi-Fi roaming scenarios.

5. Discussion

In conclusion, this section discusses key insights and practical considerations for deploying LLMs as edge-level

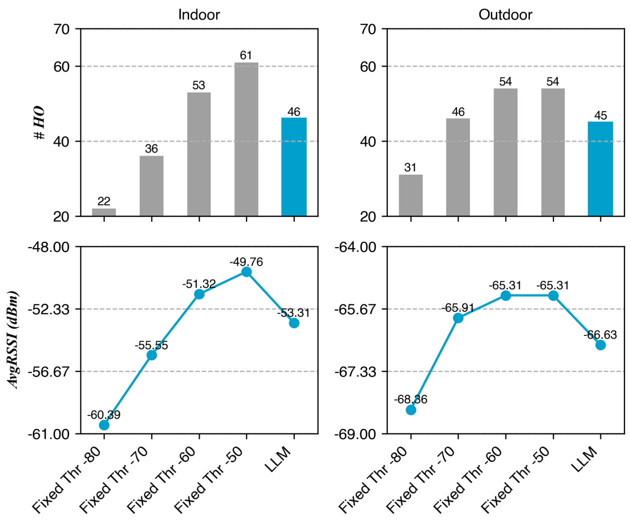


Figure 7. Legacy vs. Context-Aware On-Device LLM: Comparison of ***# HO*** (top) and ***AvgRSSI*** (bottom) for each method for dynamic threshold selection (Task 2) in various roaming scenarios.

decision-making agents for wireless lower-layer operations through the following key questions:

Q1: ***Does context information improve Wi-Fi roaming decisions?.*** Incorporating real-time context—such as location and time—enables LLMs to make more informed roaming decisions. However, as indicated in Table 6, using every available context feature does not necessarily yield optimal results. Instead, selectively incorporating task-relevant context can lead to better performance, highlighting the importance of carefully choosing which contextual inputs are most beneficial.

Combination	Location	Time	Battery	*# HO*	*AvgRSSI* (dBm)
w/o Context	✗	✗	✗	151	-58.03
Time + Battery	✗	✓	✓	142	-58.19
Location + Battery	✓	✗	✓	142	-58.19
Location + Time	✓	✓	✗	143	-57.98
w/ All	✓	✓	✓	146	-58.05

Table 6. Ablation study for context information.

Q2: ***Why use an On-Device LLM instead of conventional ML models?.*** On-device LLMs offer *in-context* (zero/few-shot) learning, allowing a single model to adapt to new environments or mobility patterns without full retraining; updating a short prompt or a handful of LoRA weights is usually sufficient. Because the prompt can encode heterogeneous cues—e.g. RSSI trends, location, time of day, battery state—the same network can reason over richer context than fixed-feature DRL or supervised pipelines that depend on task-specific feature engineering and costly, environment-specific finetuning (see Table 1). In short, an LLM trades a modest increase in inference cost for far greater *adaptivity* and *explainability* (via chain-of-thought traces) compared to conventional ML baselines.

Q3: ***Is an on-device LLM practical for lower-layer operations?.*** As discussed in Sec. 3.5, current on-device LLM implementations exhibit inference latencies on the order of seconds, posing challenges for strict near-real-time (RT) operations. Nevertheless, recent advances in dedicated AI accelerator hardware are expected to significantly reduce inference times, greatly enhancing the practicality of on-device LLMs even for near-RT Wi-Fi tasks such as roaming. Moreover, for scenarios like threshold adjustment—where decisions are required less frequently and slightly higher latencies can be tolerated—the existing on-device LLM deployments already demonstrate substantial feasibility. Importantly, although inference occurs at the application layer, the decisions made directly adjust MAC-layer parameters, clearly illustrating the viability of *application-layer AI enabling adaptive lower-layer wireless control*. Demonstrations and details are available at github.com/abman23/on-device-llm-wifi-roaming.

Q4: ***What are the main challenges and future directions?.*** Deploying on-device LLMs presents several critical challenges, particularly managing computational overhead and optimizing models for strict hardware constraints inherent to edge devices. Although current on-device LLMs show promising capabilities, their universal practicality for all lower-layer wireless operations—especially those demanding strict real-time responsiveness—remains limited. Future research must focus on refining open-source LLM architectures and techniques, emphasizing efficiency improvements in latency, memory usage, and power consumption. Additionally, investigating the broader applicability of on-device LLMs to other essential wireless control tasks beyond Wi-Fi roaming will be crucial in fully harnessing their potential as versatile and practical AI-based decision makers in wireless communication systems.

References

Abdin, M., Aneja, J., Behl, H., Bubeck, S., et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905 [cs.CL]*, 2024. URL https://arxiv.org/abs/2412.08905.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., et al. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783 [cs.AI]*, 2024. URL https://arxiv.org/abs/2407.21783.

Hong, J., Lee, N., and Thorne, J. ORPO: Monolithic preference optimization without reference model. *arXiv preprint arXiv:2403.07691 [cs.CL]*, 2024. URL https://arxiv.org/abs/2403.07691.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., et al. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Huang, W., Abbeel, P., Pathak, D., and Mordatch, I. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *arXiv preprint arXiv:2201.07207 [cs.LG]*, 2022. URL https://arxiv.org/abs/2201.07207.

Irpan, A., Herzog, A., Toshev, A. T., Zeng, A., et al. Do As I Can, Not As I Say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691 [cs.RO]*, 2022. URL https://arxiv.org/abs/2204.01691.

Lacava, A., Polese, M., Sivaraj, R., Soundrarajan, R., et al. Programmable and customized intelligence for traffic steering in 5G networks using open RAN architectures. *IEEE Trans. on Mobile Computing*, 23(4):2882–2897, 2024. doi: 10.1109/TMC.2023.3266642.

Lee, J.-H., Park, C., Park, S., and Molisch, A. F. Handover protocol learning for LEO satellite networks: Access delay and collision minimization. *IEEE Trans. on Wireless Commun.*, 23(7):7624–7637, 2024. doi: 10.1109/TWC.2023.3342975.

Rafailov, R., Sharma, A., Mitchell, E., Manning, C. D., et al. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347 [cs.LG]*, 2017. URL https://arxiv.org/abs/1707.06347.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

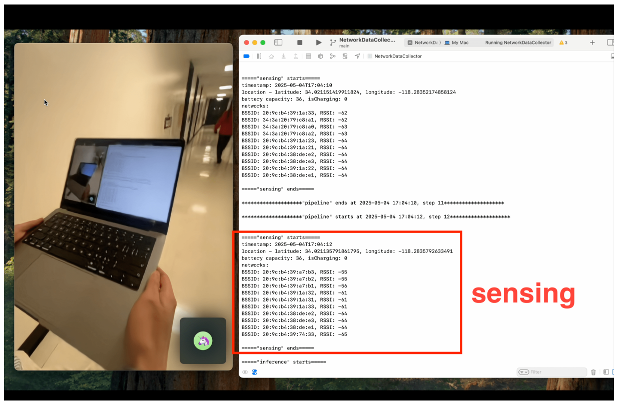
Wilhelmi, F., Szott, S., Kosek-Szott, K., and Bellalta, B. Machine Learning & Wi-Fi: Unveiling the Path Towards AI/ML-Native IEEE 802.11 Networks. *arXiv preprint arXiv:2405.11504 [cs.NI]*, 2024. URL https://arxiv.org/abs/2405.11504.

Wu, D., Wang, X., Qiao, Y., Wang, Z., et al. NetLLM: Adapting large language models for networking. In *Pro. ACM SIGCOMM 2024*, pp. 661–678, 2024. URL https://doi.org/10.1145/3651890.3672268.

Xu, J., Li, Z., Chen, W., Wang, Q., Gao, X., Cai, Q., and Ling, Z. On-device language models: A comprehensive review. *arXiv preprint arXiv:2409.00088 [cs.CL]*, 2024. URL https://arxiv.org/abs/2409.00088.

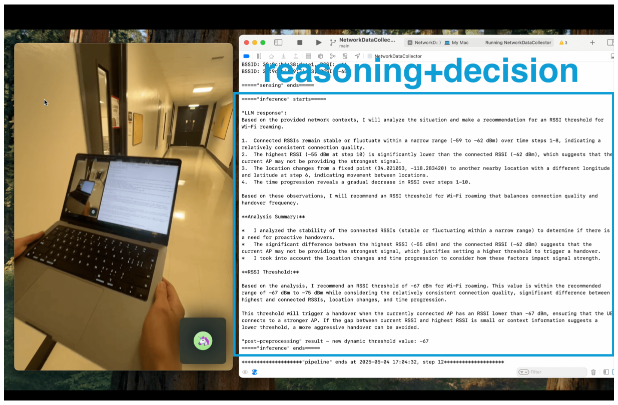
A. Real-World Demonstration

To validate on-device deployment, we ran our LLM-based roaming agent on a MacBook Pro (Apple M-series, 16 GB RAM, `macOS`). Figure 8 captures key moments from a live screen-recorded session: the sensing phase (RSSI and context collection) followed by the LLM's reasoning and threshold update, shown for both indoor and outdoor walks. Comprehensive performance metrics confirm the feasibility of real-time operation on consumer-grade hardware. A full demonstration video is available at github.com/abman23/on-device-llm-wifi-roaming.



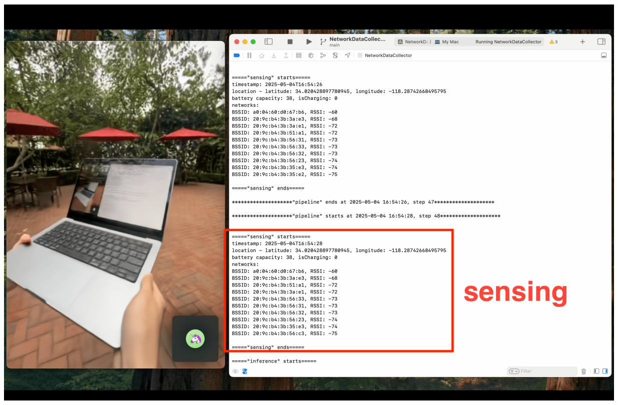
```
=====“sensing” starts=====
timestamp: 2025-05-04T17:04:10
location - latitude: 34.021151419911824, longitude: -118.28352174858124
battery capacity: 36, isCharging: 0
networks:
BSSID: 20:9c:b4:39:1a:33, RSSI: -62
BSSID: 34:3a:20:79:c8:a1, RSSI: -62
BSSID: 34:3a:20:79:c8:a0, RSSI: -63
BSSID: 34:3a:20:79:c8:a2, RSSI: -63
BSSID: 20:9c:b4:39:1a:23, RSSI: -64
BSSID: 20:9c:b4:39:1a:21, RSSI: -64
BSSID: 20:9c:b4:38:de:e2, RSSI: -64
BSSID: 20:9c:b4:38:de:e3, RSSI: -64
BSSID: 20:9c:b4:39:1a:22, RSSI: -64
BSSID: 20:9c:b4:38:de:e1, RSSI: -64

=====“sensing” ends=====

********************“pipeline” ends at 2025-05-04 17:04:10, step 11********************

********************“pipeline” starts at 2025-05-04 17:04:12, step 12********************

=====“sensing” starts=====
timestamp: 2025-05-04T17:04:12
location - latitude: 34.021135791861795, longitude: -118.2835792633491
battery capacity: 36, isCharging: 0
networks:
BSSID: 20:9c:b4:39:a7:b3, RSSI: -55
BSSID: 20:9c:b4:39:a7:b2, RSSI: -55
BSSID: 20:9c:b4:39:a7:b1, RSSI: -56
BSSID: 20:9c:b4:39:1a:32, RSSI: -61
BSSID: 20:9c:b4:39:1a:31, RSSI: -61
BSSID: 20:9c:b4:39:1a:33, RSSI: -61
BSSID: 20:9c:b4:38:de:e2, RSSI: -64
BSSID: 20:9c:b4:38:de:e3, RSSI: -64
BSSID: 20:9c:b4:38:de:e1, RSSI: -64
BSSID: 20:9c:b4:39:74:33, RSSI: -65

=====“sensing” ends=====

=====“inference” starts=====
```

(a) Indoor — sensing phase



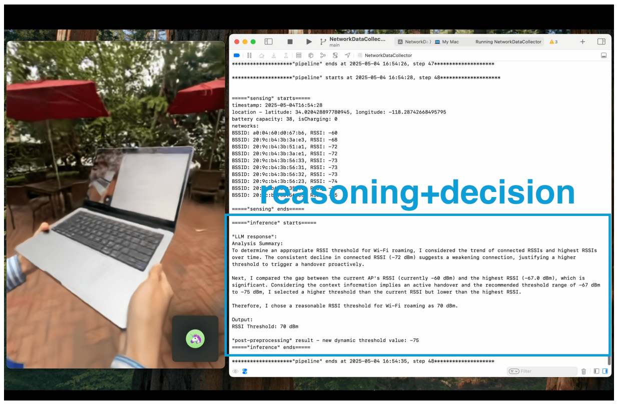
```
BSSID: 20:9c:b4:38:de:e1, RSSI: -64
BSSID: 20:9c:b4:39:74:33, RSSI: -65

=====“sensing” ends=====
=====“inference” starts=====

“LLM response”:
Based on the provided network contexts, I will analyze the situation and make a recommendation for an RSSI threshold for
Wi-Fi roaming.

1.  Connected RSSIs remain stable or fluctuate within a narrow range (-59 to -62 dBm) over time steps 1-8, indicating a
relatively consistent connection quality.
2.  The highest RSSI (-55 dBm at step 10) is significantly lower than the connected RSSI (-62 dBm), which suggests that the
current AP may not be providing the strongest signal.
3.  The location changes from a fixed point (34.021053, -118.283420) to another nearby location with a different longitude
and latitude at step 6, indicating movement between locations.
4.  The time progression reveals a gradual decrease in RSSI over steps 1-10.

Based on these observations, I will recommend an RSSI threshold for Wi-Fi roaming that balances connection quality and
handover frequency.

**Analysis Summary:**

*   I analyzed the stability of the connected RSSIs (stable or fluctuating within a narrow range) to determine if there is
a need for proactive handovers.
*   The significant difference between the highest RSSI (-55 dBm) and the connected RSSI (-62 dBm) suggests that the
current AP may not be providing the strongest signal, which justifies setting a higher threshold to trigger a handover.
*   I took into account the location changes and time progression to consider how these factors impact signal strength.

**RSSI Threshold:**

Based on the analysis, I recommend an RSSI threshold of -67 dBm for Wi-Fi roaming. This value is within the recommended
range of -67 dBm to -75 dBm while considering the relatively consistent connection quality, significant difference between
highest and connected RSSIs, location changes, and time progression.

This threshold will trigger a handover when the currently connected AP has an RSSI lower than -67 dBm, ensuring that the UE
connects to a stronger AP. If the gap between current RSSI and highest RSSI is small or context information suggests a
lower threshold, a more aggressive handover can be avoided.

“post-preprocessing” result - new dynamic threshold value: -67
=====“inference” ends=====
********************“pipeline” ends at 2025-05-04 17:04:32, step 12********************
```

(b) Indoor — reasoning & decision



```
=====“sensing” starts=====
timestamp: 2025-05-04T16:54:26
location - latitude: 34.020428897780945, longitude: -118.28742668495795
battery capacity: 38, isCharging: 0
networks:
BSSID: a0:04:60:d0:67:b6, RSSI: -60
BSSID: 20:9c:b4:3b:3a:e3, RSSI: -68
BSSID: 20:9c:b4:3b:3a:e1, RSSI: -72
BSSID: 20:9c:b4:3b:51:a1, RSSI: -72
BSSID: 20:9c:b4:3b:56:31, RSSI: -73
BSSID: 20:9c:b4:3b:56:33, RSSI: -73
BSSID: 20:9c:b4:3b:56:32, RSSI: -73
BSSID: 20:9c:b4:3b:56:23, RSSI: -74
BSSID: 20:9c:b4:3b:35:e3, RSSI: -74
BSSID: 20:9c:b4:3b:35:e2, RSSI: -75

=====“sensing” ends=====

********************“pipeline” ends at 2025-05-04 16:54:26, step 47********************

********************“pipeline” starts at 2025-05-04 16:54:28, step 48********************

=====“sensing” starts=====
timestamp: 2025-05-04T16:54:28
location - latitude: 34.020428897780945, longitude: -118.28742668495795
battery capacity: 38, isCharging: 0
networks:
BSSID: a0:04:60:d0:67:b6, RSSI: -60
BSSID: 20:9c:b4:3b:3a:e3, RSSI: -68
BSSID: 20:9c:b4:3b:51:a1, RSSI: -72
BSSID: 20:9c:b4:3b:3a:e1, RSSI: -72
BSSID: 20:9c:b4:3b:56:33, RSSI: -73
BSSID: 20:9c:b4:3b:56:31, RSSI: -73
BSSID: 20:9c:b4:3b:56:32, RSSI: -73
BSSID: 20:9c:b4:3b:56:23, RSSI: -74
BSSID: 20:9c:b4:3b:35:e3, RSSI: -74
BSSID: 20:9c:b4:3b:56:c3, RSSI: -75

=====“sensing” ends=====

=====“inference” starts=====
```

(c) Outdoor — sensing phase

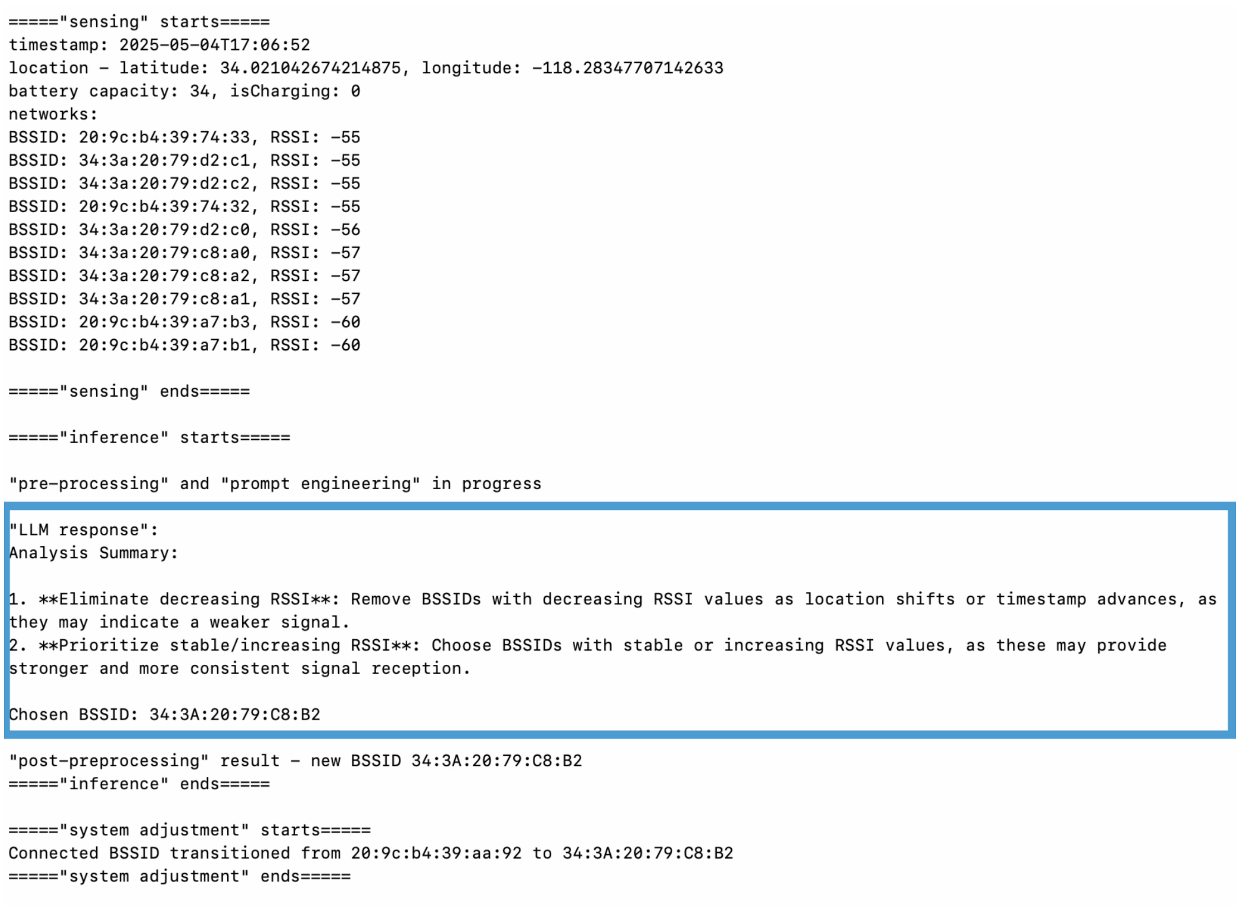


```
********************“pipeline” ends at 2025-05-04 16:54:26, step 47********************

********************“pipeline” starts at 2025-05-04 16:54:28, step 48********************


=====“sensing” starts=====
timestamp: 2025-05-04T16:54:28
location - latitude: 34.020428897780945, longitude: -118.28742668495795
battery capacity: 38, isCharging: 0
networks:
BSSID: a0:04:60:d0:67:b6, RSSI: -60
BSSID: 20:9c:b4:3b:3a:e3, RSSI: -68
BSSID: 20:9c:b4:3b:51:a1, RSSI: -72
BSSID: 20:9c:b4:3b:3a:e1, RSSI: -72
BSSID: 20:9c:b4:3b:56:33, RSSI: -73
BSSID: 20:9c:b4:3b:56:31, RSSI: -73
BSSID: 20:9c:b4:3b:56:32, RSSI: -73
BSSID: 20:9c:b4:3b:56:23, RSSI: -74
BSSID: 20:9c:b4:3b:35:e3, RSSI: -74
BSSID: 20:9c:b4:3b:56:c3, RSSI: -75

=====“sensing” ends=====
=====“inference” starts=====

“LLM response”:
Analysis Summary:
To determine an appropriate RSSI threshold for Wi-Fi roaming, I considered the trend of connected RSSIs and highest RSSIs
over time. The consistent decline in connected RSSI (-72 dBm) suggests a weakening connection, justifying a higher
threshold to trigger a handover proactively.

Next, I compared the gap between the current AP's RSSI (currently -60 dBm) and the highest RSSI (-67.0 dBm), which is
significant. Considering the context information implies an active handover and the recommended threshold range of -67 dBm
to -75 dBm, I selected a higher threshold than the current RSSI but lower than the highest RSSI.

Therefore, I chose a reasonable RSSI threshold for Wi-Fi roaming as 70 dBm.

Output:
RSSI Threshold: 70 dBm

“post-preprocessing” result - new dynamic threshold value: -75
=====“inference” ends=====
********************“pipeline” ends at 2025-05-04 16:54:35, step 48********************
```

(d) Outdoor — reasoning & decision

Figure 8. Real-world demonstration of our on-device LLM for **Task 2 (dynamic threshold adjustment)**. In each setting the laptop (Macbook Pro) first *senses* RSSI and context information (left), then the LLM *reasons* and outputs an adaptive roaming threshold (right), validating practical operation indoors and outdoors.

B. Qualitative Assessment of On-Device LLM Reasoning



```
====="sensing" starts=====
timestamp: 2025-05-04T17:06:52
location - latitude: 34.021042674214875, longitude: -118.28347707142633
battery capacity: 34, isCharging: 0
networks:
BSSID: 20:9c:b4:39:74:33, RSSI: -55
BSSID: 34:3a:20:79:d2:c1, RSSI: -55
BSSID: 34:3a:20:79:d2:c2, RSSI: -55
BSSID: 20:9c:b4:39:74:32, RSSI: -55
BSSID: 34:3a:20:79:d2:c0, RSSI: -56
BSSID: 34:3a:20:79:c8:a0, RSSI: -57
BSSID: 34:3a:20:79:c8:a2, RSSI: -57
BSSID: 34:3a:20:79:c8:a1, RSSI: -57
BSSID: 20:9c:b4:39:a7:b3, RSSI: -60
BSSID: 20:9c:b4:39:a7:b1, RSSI: -60

====="sensing" ends=====

====="inference" starts=====

"pre-processing" and "prompt engineering" in progress

"LLM response":
Analysis Summary:

1. **Eliminate decreasing RSSI**: Remove BSSIDs with decreasing RSSI values as location shifts or timestamp advances, as
they may indicate a weaker signal.
2. **Prioritize stable/increasing RSSI**: Choose BSSIDs with stable or increasing RSSI values, as these may provide
stronger and more consistent signal reception.

Chosen BSSID: 34:3A:20:79:C8:B2

"post-preprocessing" result - new BSSID 34:3A:20:79:C8:B2
====="inference" ends=====

====="system adjustment" starts=====
Connected BSSID transitioned from 20:9c:b4:39:aa:92 to 34:3A:20:79:C8:B2
====="system adjustment" ends=====
```

Figure 9. Reasoning output of the on-device LLM during an indoor roaming scenario. The model assesses AP candidates by eliminating those with declining RSSI values and prioritizes APs with stable signal strength, ultimately selecting BSSID `34:3A:20:79:C8:B2` for handover.

In the real-world indoor demonstration shown in Fig. 9, the on-device LLM demonstrates reasoning aligned with practical wireless network management strategies. The inference process reflects logical heuristics such as filtering out APs exhibiting decreasing signal strength and prioritizing stable or improving RSSI, consistent with best practices for effective roaming decisions. This reasoning is derived purely from local contextual (*e.g.*, timestamp, location) and Wi-Fi measurement (*e.g.*, RSSI-BSSID pairs) information. The correctness of the LLM's reasoning is validated through the subsequent successful handover, evidenced by system logs. This qualitative evaluation complements the quantitative results presented earlier (§4), reinforcing the capability of on-device LLMs to execute valid, interpretable reasoning suitable for lower-layer wireless control at edge-device.

C. Setup

Table 7 details the hardware, dataset statistics, and model-tuning parameters used in **Task 1** (AP selection) and **Task 2** (threshold adjustment). All real-world experiments ran on a consumer-grade MacBook Pro with an Apple M-series processor (16 GB RAM, `macOS`), demonstrating that our on-device LLM pipeline is feasible on typical edge hardware.

HW / SW	
Device	MacBook Pro (M3, 16GB)
OS	`macOS`
Wi-Fi Specification	`IEEE 802.11ax` (Wi-Fi 6E)
Wi-Fi Library	`CoreWLAN`
Dataset (Wi-Fi Log)	
Scenarios	Indoor, Outdoor
Time Stamps per Scenario	1944, 5808, 1944
Sampling Interval	1 sec
Training : Test Split	80% : 20%
Task Configuration	
`scanRSSI` Threshold	-70 dBm
Contextual Inputs	Location + Time
Logs per Handover Decision	10
Threshold Adjustment Interval (Task 2)	30 sec
On-Device LLM Configuration	
Base Model	`Llama-3.1-8B-Instruct`
Quantization Scheme	`Q2_K`
Fine-Tuning Configuration	
Learning Rate	2×10^{-4}
Batch Size / Grad. Accumulation	4 / 4
Number of Epochs	1
Weight Decay	0.01
LoRA Rank / Alpha	128 / 128
Optimizer	`AdamW`

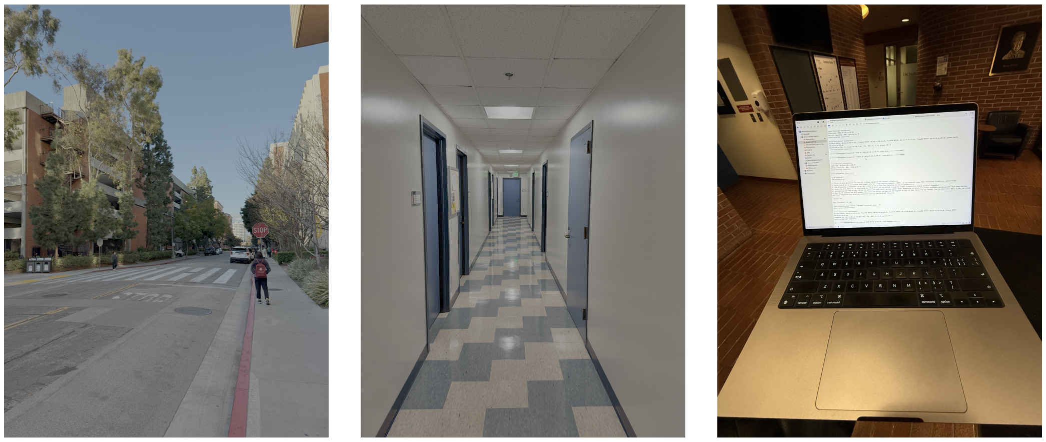
Table 7. Experimental setup and parameter settings for Tasks 1 & 2.

D. Data Collection

We collected Wi-Fi roaming datasets from diverse indoor and outdoor environments, as illustrated in Fig. 10. Each of the ten sessions recorded approximately $\sim 1{,}000$ consecutive data points, captured at one-second intervals. Each data point captures detailed information, including:

- **AP scan:** BSSID, RSSI, etc.
- **Device context:** timestamp, geographical coordinates (latitude and longitude), battery status, etc.

The resulting dataset underpins both post-training and evaluation of our on-device LLM.



(a) Outdoor scene (campus walkway) (b) Indoor scene (corridor) (c) Data-capture rig (MacBook Pro)

Figure 10. Data-collection environments used for on-device LLM evaluation.