Pak. J. Bot., 58(7): 1583-1599, July 2026.

ARTICLE | DOI: http://dx.doi.org/10.30848/PJB2026-7(16)

TINY-GRU BENCHMARKING FOR EDGE-BASED REFERENCE EVAPOTRANSPIRATION PREDICTION IN SMART PADDY IRRIGATION

KAI LUO[1,2], CHENG SIONG LIM[2]*, MOHAMAD HAFIS IZRAN BIN ISHAK[2], MOHD SAIFUL AZIMI MAHMUD[2] AND XUE YANG[1]

[1]School of Engineering, Chengdu College of University of Electronic Science and Technology of China, Chengdu 611731, Sichuan, China
[2]Faculty of Electrical Engineering, Universiti Teknologi Malaysia, Skudai 81300, Johor, Malaysia

*Corresponding author's email: lcsiong@utm.my

Abstract

Global food security and increasing water scarcity demand more efficient paddy irrigation scheduling, while the limited computational power and memory of edge devices pose challenges for accurate reference crop evapotranspiration (ET_0) prediction. To address this issue, we propose a lightweight GRU variant, termed Tiny-GRU, which integrates temporal attention, unstructured pruning, and post-training dynamic range quantization to enable efficient deployment under resource-constrained edge computing. Nine model configurations were benchmarked on a Raspberry Pi 5 and a resource-constrained Intel i5 platform, with performance evaluated in terms of predictive accuracy, inference latency, memory usage, and thermal stability. Results indicate that the optimized Tiny-GRU configurations achieve high forecasting accuracy (RMSE ≈ 0.02 mm·h^{-1}) while maintaining fast on-device inference (≈ 0.21 ms per sample on Raspberry Pi 5) and stable runtime behavior suitable for continuous edge operation. Statistical analysis confirmed that the observed performance differences among model configurations were significant ($p < 0.001$). Furthermore, a cross-platform visualization framework was developed to analyze trade-offs between predictive performance and resource consumption, thereby supporting decision-making for edge-oriented benchmarking in smart paddy irrigation. Overall, the proposed Tiny-GRU framework provides a practical and scalable solution for efficient ET_0 forecasting on resource-limited edge devices.

Key words: Reference evapotranspiration (*ET_0*); Lightweight GRU; Temporal attention pooling; Unstructured pruning; Post-training quantization; Edge computing; Smart paddy irrigation; Multi-metric benchmarking framework

Introduction

Global food security and freshwater resources are encountering challenges at a scale rarely seen before. Factors such as climate change, population growth, and inefficient irrigation practices continue to exacerbate water scarcity (Liu *et al.*, 2022; He & Rosa, 2023). Against this background, precision agriculture and smart irrigation have increasingly been viewed as practical means to enhance agricultural productivity while promoting sustainable water management (Sarkar *et al.*, 2025). The accurate prediction of reference crop evapotranspiration (ET_0) is especially important for irrigation scheduling and water allocation (Zhang *et al.*, 2023). It provides the scientific basis for estimating crop water demand, defining irrigation quotas, and maintaining field water balance. Recent studies have reported that inefficient irrigation practices in paddy fields can result in approximately 20–40% water loss, underscoring the urgent need for more precise irrigation scheduling strategies (e.g., Parsinejad *et al.*, 2008).

Traditional ET_0 estimation methods—including the FAO Penman–Monteith equation, regression-based models, and multilayer perceptrons—remain widely used due to their interpretability, yet they often struggle to capture nonlinear dynamics or long-term dependencies. Consequently, their adaptability under complex or changing climatic conditions can be limited (He & Rosa, 2023; Sarkar *et al.*, 2025).

In contrast, cloud-based ET_0 forecasting approaches have emerged in recent years by leveraging centralized computational resources to improve prediction accuracy. Despite their effectiveness, such systems rely on stable communication links and sufficient bandwidth, making them vulnerable to latency and connectivity issues. Continuous data transmission from agricultural IoT networks increases energy consumption and raises privacy concerns. These factors limit the sustainability of cloud deployment, particularly in regions with poor connectivity or complex terrain.

Edge computing, by contrast, enables local data processing with low latency and reduced energy cost, which makes it an appealing alternative for ET_0 prediction—especially in benchmarking lightweight models for smart paddy irrigation (Zou *et al.*, 2024). However, the limited computational capacity and memory of edge devices introduce fundamental trade-offs. Achieving low latency, minimal memory use, and thermal stability without sacrificing predictive accuracy remains a core challenge in this context (Zou *et al.*, 2024).

Deep time-series models have shown strong predictive capabilities in meteorology and agriculture. Recurrent neural networks (RNNs), long short-term memory (LSTM) networks, and gated recurrent units (GRUs) have been widely applied to agricultural and meteorological time-series prediction (Selva Jeba & Chitra, 2024; Farman *et al.*, 2025), while temporal

Received: 25-11-2025 | Revised: 19-02-2026 | Accepted: 24-02-2026 | Online: 20-03-2026

convolutional networks (TCNs) and lightweight Transformer-based models have been reported to effectively capture nonlinear dynamics and long-term dependencies (Villegas-Vega *et al.*, 2025). Our previous work demonstrated that GRUs outperform other models in predictive accuracy on low-power platforms (Yassen *et al.*, 2025). However, their inference latency and memory consumption remain relatively high. Existing lightweight approaches, such as TinyRNN, EDL-GRU, and MCUNet (Alandjani, 2024), reduce model complexity through pruning and quantization. However, their application in ET_0 forecasting has so far focused primarily on individual optimization strategies. Systematic cross-platform evaluation across multiple performance dimensions remains relatively limited.

The introduction of attention mechanisms has also attracted growing interest (Cai *et al.*, 2025; Guo *et al.*, 2025). While multi-head Transformer-style attention can enhance feature representation, its computational and memory requirements may pose challenges for deployment in resource-constrained embedded environments (Guo *et al.*, 2025). To address this issue, lightweight temporal attention pooling mechanisms have been proposed as a more efficient alternative, offering improved long-term dependency modeling with substantially lower computational overhead (Cai *et al.*, 2025). Despite these advances, achieving a balanced trade-off among predictive accuracy, inference latency, memory efficiency, and thermal stability in lightweight architectures remains an open challenge. This challenge is particularly evident for ET_0 prediction in smart paddy irrigation under edge constraints.

To address these challenges, this study proposes and validates a lightweight Tiny-GRU framework for edge computing, with a particular focus on the joint optimization of temporal attention pooling (TAP), unstructured pruning, and post-training dynamic range quantization (DRQ). Nine configuration paths (P1–P9) were designed for this study. P1 represents the baseline GRU, while P2 defines a lightweight Tiny-GRU that serves as the initial configuration. The remaining paths, P3–P9, gradually combine pruning, quantization, and temporal attention, forming multiple routes toward multi-strategy optimization.

Model training and baseline inference were conducted on a general-purpose computer equipped with an Intel® Core™ i5-12450H (12th Gen, Intel Corporation, USA), hereafter referred to as the Intel i5 platform. To approximate the computational limitations typical of embedded edge environments, the Intel i5 setup was restricted to single-threaded CPU execution, with GPU acceleration disabled, so as to emulate a constrained processing scenario. In parallel, a Raspberry Pi 5 (Raspberry Pi Foundation, UK) was used as a representative embedded edge device for benchmarking. This configuration enabled systematic measurements of inference latency, memory footprint, and thermal stability.

The experimental findings indicated that the proposed framework preserved high predictive accuracy while notably improving computational efficiency and resource adaptability. These results suggest its suitability for embedded and edge applications under resource-constrained conditions (Bazargani *et al.*, 2025; Chiranjeevi & Chavan, 2025). Within this setting, the study explored four main research questions.

Collectively, these questions can be grouped into two overarching themes: (i) multi-strategy optimization of lightweight GRU models to balance predictive accuracy and computational efficiency on edge devices, and (ii) cross-platform benchmarking and decision support for ET_0 prediction under resource-constrained edge environments.

(a) How can pruning and quantization methods—specifically unstructured pruning and post-training DRQ—be applied to optimize GRU architectures so as to balance predictive accuracy and benchmarking efficiency on edge devices?
(b) What are the separate influences of pruning, quantization, and temporal attention on model accuracy, latency, memory usage, and thermal stability?
(c) Can integrated multi-strategy optimization yield synergistic benefits and generate Pareto-optimal trade-offs that outperform single-strategy configurations?
(d) In what ways can a cross-platform, multi-metric evaluation framework, supported by visualization techniques, assist in optimizing and guiding ET_0 prediction for smart paddy irrigation?

To address these questions, a cross-platform and multi-metric benchmarking system was constructed. It systematically evaluates optimization paths with respect to predictive accuracy, inference latency, model size, memory demand, and thermal behavior, while revealing the trade-offs among these factors. The framework also helps identify both individual and combined effects of optimization strategies, thereby offering practical guidance for edge benchmarking in smart irrigation contexts. The proposed Tiny-GRU optimization framework, together with its evaluation tools, establishes a reproducible foundation for large-scale ET_0 prediction in resource-limited environments. It further shows promising generalization capacity, potentially extending to other IoT-based time-series domains such as environmental monitoring and energy forecasting.

The contributions of this study are threefold: (a) it presents a Tiny-GRU framework that integrates temporal attention, pruning, and quantization through multi-strategy optimization; (b) it proposes a cross-platform benchmarking protocol on Intel i5 and Raspberry Pi 5 devices, using thermal behavior as an indicator of energy efficiency; and (c) it introduces decision-support tools that employ Pareto fronts and multi-metric visualization to assist model selection under edge constraints.

The remainder of this paper is organized as follows. Section 2 reviews related studies on ET_0 prediction, deep time-series modeling, and optimization for edge applications, and highlights the novelty of this work. Section 3 details the benchmarking methodology, including data preprocessing, model structure, optimization schemes, platforms, and performance metrics. Section 4 presents and discusses the experimental outcomes with a focus on accuracy, efficiency, and thermal stability, as well as visualization-based decision tools. Section 5 concludes the paper with a summary of key findings, implications, and potential directions for further study.

Related Work

ET_0 Forecasting with deep time-series models under edge constraints: Accurate forecasting of reference crop evapotranspiration (ET_0) is fundamental to precision

irrigation and agricultural water management (Liu *et al.*, 2022). Traditional approaches, including the FAO Penman–Monteith equation, regression-based models, and multilayer perceptrons, remain widely used due to their physical interpretability and low computational cost (He & Rosa, 2023). However, these methods generally struggle to capture nonlinear interactions and long-term temporal dependencies, limiting their robustness under highly variable climatic conditions (Sarkar *et al.*, 2025).

Recent advances in deep time-series learning—such as recurrent neural networks (RNNs), long short-term memory (LSTM), gated recurrent units (GRUs), temporal convolutional networks (TCNs), and lightweight Transformer variants—have significantly improved ET_0 prediction accuracy compared with conventional models (Liu *et al.*, 2022; He & Rosa, 2023; Tausif *et al.*, 2023; Zhang *et al.*, 2023; Sarkar *et al.*, 2025). Among these architectures, GRUs are often preferred in resource-constrained settings because of their relatively compact structure, reduced parameter count, and stable performance across different hardware platforms (Zou *et al.*, 2024; Bazargani *et al.*, 2025). Nevertheless, even GRU-based models can exhibit non-negligible inference latency and memory demand when deployed on embedded devices, such as the Raspberry Pi, particularly for multivariate agricultural time series (Bazargani *et al.*, 2025; Chiranjeevi & Chavan, 2025).

With the increasing adoption of agricultural Internet of Things (IoT) systems, real-time ET_0 forecasting on edge devices has become increasingly important. However, the limited computational capacity, memory, and energy budget of edge platforms impose strict constraints on model complexity and runtime efficiency (Zhang *et al.*, 2023; Tausif *et al.*, 2023). To address these challenges, recent studies have explored lightweight design strategies, including pruning, quantization, and structural simplification, as means to reduce model size and inference cost (Wen *et al.*, 2020; Valerio *et al.*, 2021; Kim *et al.*, 2025). Frameworks such as MCUNet and EDL-GRU have demonstrated the feasibility of deploying compressed neural networks on embedded hardware (Wen *et al.*, 2020; Valerio et al., 2021; Kim *et al.*, 2025). Despite these advances, most existing studies focus on single optimization techniques and are often evaluated on a single platform, providing limited insight into cross-platform robustness and multi-objective trade-offs (Kim *et al.*, 2025). In addition, thermal stability and sustained runtime behavior—critical for long-term deployment on embedded edge devices—are rarely evaluated explicitly in these lightweight frameworks, including representative systems such as MCUNet.

Attention mechanisms have also been introduced to enhance the representational capacity of time-series models by emphasizing critical temporal segments (Kang *et al.*, 2024; Yin *et al.*, 2025). In contrast to multi-head Transformer attention, lightweight temporal attention pooling can improve long-term dependency modeling with substantially lower computational overhead, making it more suitable for real-time and edge-oriented applications (Kang *et al.*, 2024; Yin *et al.*, 2025). These characteristics suggest that combining lightweight GRU architectures with efficient temporal attention may provide a promising foundation for ET_0 forecasting under edge-computing constraints.

Model compression and multi-strategy optimization for edge deployment: Model compression techniques, particularly pruning and quantization, have become central to improving the deployability of deep learning models on edge devices. Pruning removes redundant parameters to reduce memory usage and computational cost, while quantization lowers numerical precision to achieve more compact model representations (Wen *et al.*, 2020; Jacob *et al.*, 2018). In time-series forecasting tasks, unstructured pruning is frequently adopted because of its flexibility and compatibility with existing model architectures, although its effectiveness on resource-limited hardware can depend on the execution framework and fine-tuning strategy (Wen *et al.*, 2020; Valerio *et al.*, 2021).

Quantization methods, including post-training quantization (PTQ), quantization-aware training (QAT), and post-training dynamic range quantization (DRQ), offer different trade-offs between accuracy preservation and deployment complexity (Jacob *et al.*, 2018; Kang *et al.*, 2024). Among these approaches, DRQ has attracted attention as a practical compromise, as it reduces model size and computational demand without requiring retraining, while maintaining good portability across heterogeneous hardware platforms (Jacob *et al.*, 2018; Alandjani, 2024). However, systematic evaluations of DRQ for agricultural ET_0 forecasting—especially under varying edge hardware constraints—remain scarce.

Recent work has begun to explore the joint use of lightweight architectures, pruning, quantization, and attention mechanisms to form multi-strategy optimization pipelines (Kang *et al.*, 2024; Kim *et al.*, 2025). While these approaches show promise, most studies emphasize predictive accuracy or inference latency in isolation, with limited consideration of memory usage, thermal stability, and cross-platform consistency. Unlike multi-head self-attention mechanisms commonly adopted in Transformer-based or attention-augmented models, which introduce additional projection matrices and increased computational overhead, the proposed temporal attention pooling (TAP) employs a lightweight temporal weighting scheme without increasing model depth. As a result, TAP incurs only marginal computational cost and memory footprint, making it well suited for deployment on resource-constrained edge devices. Moreover, comprehensive benchmarking on real-world edge devices is still lacking, particularly for agricultural time-series applications that involve multivariate inputs and strong diurnal variability (Tanabe & Ishibuchi, 2020; Chen & Xiao, 2025; Yin *et al.*, 2025).

These limitations highlight the need for a unified framework that systematically integrates lightweight GRU design, efficient temporal attention, and complementary compression strategies, and evaluates their combined effects across multiple edge platforms. Addressing these gaps is essential for establishing practical guidelines to balance predictive accuracy, latency, memory efficiency, and thermal behavior in intelligent irrigation systems.

Material and Method

Data and preprocessing: The meteorological dataset used in this study was obtained from Station 56289 of the China Meteorological Administration (CMA), located in Pengshan District, Sichuan Province, China (30.20° N, 103.87° E;

elevation = 437 m). The dataset spaned from January 1, 2021, to December 31, 2023, and consisted of near-surface meteorological observations relevant to evapotranspiration processes. This dataset is publicly accessible via the China Meteorological Administration's Open Data Portal (https://data.cma.cn), ensuring transparency and reproducibility of meteorological research.

The data acquisition, temporal reconstruction, preprocessing, and quality control procedures follow our previously published benchmarking study on edge-based ET_0 forecasting (Luo *et al.*, 2025). Only a brief summary is provided here for completeness. In that work, raw observations were reindexed and resampled to an hourly resolution, missing values were handled using variable-specific interpolation strategies, and outliers were detected and corrected using standard statistical criteria to ensure physical consistency and temporal continuity.

To further reduce high-frequency measurement noise introduced by sensor fluctuations and transmission artifacts, a Savitzky–Golay (SG) filter followed by a short moving-average smoothing was applied during preprocessing. These smoothing operations were designed solely for noise suppression and not intended to modify the underlying physical trends of the time series. Importantly, the applied filters preserve the mean level, diurnal pattern, and long-term variability of ET_0, and therefore do not introduce systematic bias into the evapotranspiration signals. Similar preprocessing strategies have been shown to effectively improve numerical stability while maintaining physical consistency in hourly ET_0 analysis (Luo *et al.*, 2025).

Hourly reference evapotranspiration (ET_0) was computed using the FAO-56 Penman–Monteith formulation (Allen *et al.*, 1998), which is widely adopted in irrigation scheduling and agro-hydrological applications. To support short-term forecasting, the FAO-56 equation—originally defined at a daily scale—was extended to hourly resolution following the ASCE standardized methodology (López-Urrea *et al.*, 2006). Previous studies have demonstrated that the ASCE-standardized hourly FAO-56 formulation provides reliable ET_0 estimates when compared with lysimeter measurements and high-resolution observational data (Trajković, 2010). The resulting hourly ET_0 series served as the prediction target for all experiments.

Model inputs comprise air temperature, relative humidity, wind speed, solar radiation, and soil moisture at 10 cm depth, together with a first-order autoregressive term of ET_0, which has been shown to improve short-term predictive performance in time-series forecasting (Luo *et al.*, 2025). Input sequences were constructed using a sliding-window approach for one-step-ahead prediction. Based on prior analysis, a 12-hour lookback window was adopted as the default configuration, providing an effective balance between temporal context and computational efficiency for edge deployment. In our previous study (Luo *et al.*, 2025), comparative experiments using 6 h, 12 h, and 24 h sliding windows showed that the 12 h configuration consistently achieved lower RMSE than shorter windows while avoiding the accuracy saturation and increased computational overhead observed with longer windows. Therefore, the 12-hour window was selected as a well-justified compromise between predictive performance and edge-computing efficiency.

To prevent information leakage, the dataset was split chronologically, with data from 2021–2022 was used for training and data from 2023 was reserved for validation. It should be noted that the dataset used in this study was obtained from a single meteorological station, and therefore the spatial generalizability of the reported results was inherently limited. The primary objective of this work is not to evaluate cross-regional generalization performance, but to benchmark the computational efficiency, runtime stability, and deployment feasibility of lightweight GRU-based models under resource-constrained edge-computing conditions. In this context, the single-station setting provides a controlled and reproducible testbed for cross-platform performance evaluation. Future work will extend the proposed framework to multi-station and cross-regional datasets to further investigate spatial transferability.

Details of the full preprocessing pipeline, including missing-value imputation, outlier correction, feature normalization, and sequence construction, are reported in Luo *et al*., (2025) and are not repeated here.

Model architecture and optimization strategies: To improve the efficiency of ET_0 forecasting on embedded edge devices, this study introduces a lightweight framework built upon the GRU architecture and augmented with several complementary optimization strategies. The design aims to reduce inference latency and memory demand while also enhancing thermal stability, which is treated here as an indirect measure of energy efficiency. At the same time, the framework maintains the level of predictive accuracy necessary for dependable operation under constrained computational resources. In essence, it seeks to strike a practical balance between precision and deployability for real-world edge applications.

Figure 1 provides an overview of the proposed Tiny-GRU framework, which is organized into five major components:

(a) Problem context: outlines the challenges of water scarcity, network limitations, and restricted computing power in smart paddy irrigation.
(b) Data preprocessing: includes feature selection, interpolation, normalization, and sliding-window generation.
(c) Model optimization: defines nine configuration paths (P1–P9) based on a Tiny-GRU backbone and integrates temporal attention, pruning, and dynamic range quantization (DRQ).
(d) Cross-platform training and benchmarking: conducted on an Intel i5-12450H platform (training + benchmark) and a Raspberry Pi 5 device (benchmark).
(e) Evaluation and decision analysis: combines accuracy, efficiency, and thermal indicators through Pareto-front and heatmap visualization to guide deployment decisions.

Architecture design and enhancement

(a) Tiny-GRU: Building upon the baseline model P1—a two-layer unidirectional GRU with 128 and 64 hidden units and roughly 118 k trainable parameters—a compact variant termed *Tiny-GRU* (P2) was developed to reduce computational load and parameter size without sacrificing

predictive capability. The model follows a deep-but-narrow design that consists of three unidirectional GRU layers with progressively decreasing hidden units (64 → 32 → 16). The first two layers output complete temporal sequences to capture dynamic dependencies, whereas the final layer extracts only the last hidden state, which is then passed through a linear layer to produce the forecast.

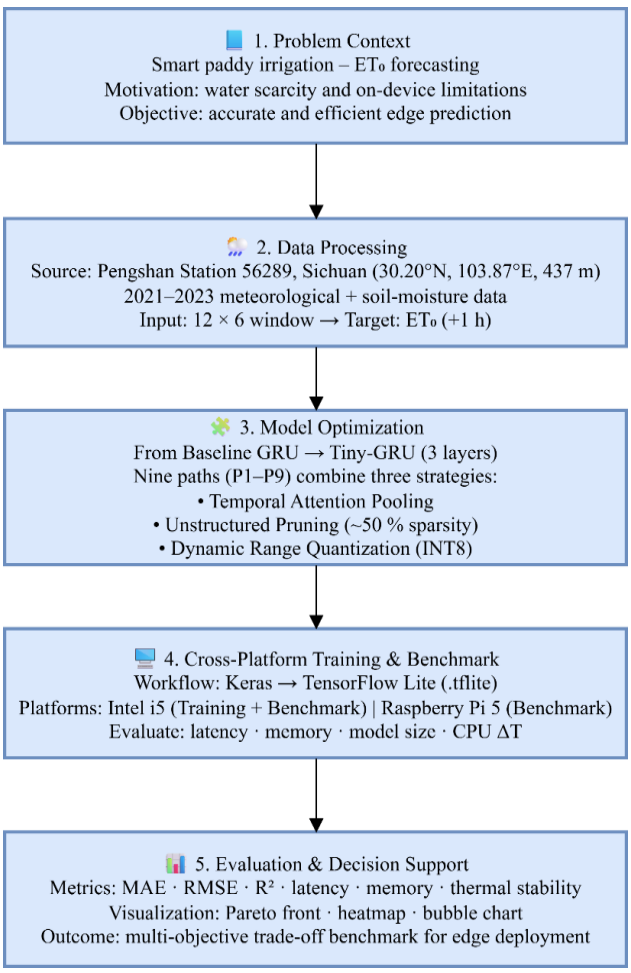

Fig. 1. Overall architecture of the proposed Tiny-GRU framework, illustrating five integrated components: problem context, data preprocessing, model optimization, cross-platform benchmarking, and evaluation/decision analysis.

Relative to P1, the Tiny-GRU model contains about 25 k trainable parameters and performs roughly 0.3 M multiply–accumulate operations (MACs) per sample when trained on a 12-hour input window with six features. These values represent approximately a 79% reduction in parameter count and a 75% reduction in MACs compared with the baseline model's 118 k parameters and 1.2 M MACs per sample. The architecture draws inspiration from progressive channel-reduction strategies used in lightweight CNNs but has been adapted here for sequential forecasting to balance representational efficiency and predictive accuracy (Liang *et al.*, 2021; Ullah *et al.*, 2021; Zhang *et al.*, 2023).

The Tiny-GRU offers several practical benefits:

- **Reduced complexity:** The gradual decrease in hidden units prevents quadratic parameter growth, preserving model capacity while keeping the overall size compact.
- **Edge adaptability:** Smaller hidden states lower memory bandwidth and cache requirements, which in turn shorten inference latency.
- **Stable convergence:** The training process employs an early-stopping mechanism to stabilize convergence and avoid overfitting (details in Section 3.4).

For fair comparison, all models were trained and evaluated under identical preprocessing (12-hour sliding window, Min–Max normalization), optimization settings (same patience for early stopping), and inference conditions.

Note: Parameter count and multiply–accumulate operations (MACs) were computed under a 12-hour input window with six input features. Values are theoretical estimates obtained from TensorFlow model summaries.

As shown in Table 1, Tiny-GRU substantially decreases computational burden while maintaining forecasting accuracy. In summary, P2 represents a lightweight refinement of the baseline GRU and serves as the foundation for subsequent compression and enhancement strategies. Its effects on runtime memory footprint and thermal stability are examined in Section 4.4 under the standardized experimental settings described in Section 3.4.1.

(b) Temporal attention pooling: Building on the hidden state sequence as shown in (2), we introduced a single-head linear self-attentive pooling mechanism to dynamically capture the relative contributions of different time steps.

$$X = [x_1, x_2, \ldots, x_T] \in R^{T \times d} \quad (1)$$

Here, X represents the sequence of hidden state vectors output by the Tiny-GRU (P2), where T is the sequence length and d denotes the dimensionality of each hidden state.

This approach is conceptually related to prior work on attention-based temporal modeling in time-series forecasting (Cheng *et al.*, 2020; Chen *et al.*, 2023), as shown in (3)–(5), but it is adapted here for resource-constrained edge applications with small hidden dimensions.

The attention scores were first obtained through a linear projection:

$$e_t = w_a^{\top} x_t + b_a, (w_a \in R^d, b_a \in R) \quad (2)$$

Table 1. Architectural and computational comparison between the baseline GRU (P1) and the lightweight Tiny-GRU (P2).

Model	Network configuration	Parameters (k)	MACs per sample (M)	Suitability for ET₀ prediction
P1 (Baseline GRU)	2 × (128, 64)	118	≈ 1.2	High accuracy, but limited suitability for edge-based ET₀ prediction due to higher computational and memory demand
P2 (Tiny-GRU)	3 × (64, 32, 16)	25	≈ 0.3	High suitability for edge-based ET₀ prediction, offering a favorable balance between accuracy and computational efficiency

The scores were then normalized using the Softmax function to derive attention weights:

$$\alpha_t = \frac{\exp(e_t)}{\sum_{j=1}^{T} exp(e_j)}, \sum_{t=1}^{T} \alpha_t = 1 \quad (3)$$

Finally, a weighted aggregation was performed to form the context vector:

$$context = \sum_{t=1}^{T} \alpha_t x_t \in R^d \quad (4)$$

This mechanism requires only $d + 1$ trainable parameters with a computational complexity of $\mathcal{O}(T\ d)$, which is lower than additive attention (requiring a feed-forward network) or scaled dot-product attention (requiring Q/K/V matrix operations). By avoiding multiple matrix multiplications, linear attention pooling reduces memory usage and inference latency on embedded CPUs (e.g., Raspberry Pi), making it more suitable for real-time applications on edge devices.

Compared with conventional mean pooling and last-step pooling, temporal attention pooling can dynamically assign weights α_t, emphasizing the most influential time steps for prediction. Moreover, the interpretability of the model is enhanced, as the visualization of attention weights provides interpretable insights into the meteorological conditions that predominantly influence forecasting outcomes. To ensure fairness, all pooling methods were compared under identical preprocessing, sequence length (12 h), and optimization settings.

Compression and lightweighting

(a) Magnitude-based pruning and weight sparsification: To benchmark reductions in storage requirements and inference latency, a magnitude-based pruning strategy was applied to the backbone of Tiny-GRU (P2, lightweight baseline), introducing weight sparsification for redundant parameters (Tang *et al.*, 2022). This method was originally proposed by Han *et al.* and later implemented in the TensorFlow Model Optimization Toolkit by Zhu and Gupta, where it has been widely adopted. Specifically, all three GRU layers and the output dense layer were wrapped with pruning operations. During training, a polynomial schedule was employed to gradually set low-magnitude weights to zero, with the sparsity ratio increasing from 0.0 to 0.5 as defined in (5):

$$\text{sparsity}(t) = 0.0 \rightarrow 0.5, t \in [0,\text{end\_step}] \quad (5)$$

The upper bound of training steps was computed according to (6):

$$end\_\text{step} = \left\lceil \frac{N_{train}}{32} \right\rceil \times 20 \quad (6)$$

This training configuration corresponds to a batch size of 32 and a maximum of 20 epochs, which fully covers the model's convergence process. To preserve both interpretability and numerical stability within the attention mechanism, the scoring layer, Softmax normalization, and weighted aggregation modules were excluded from pruning. After training, the strip_pruning operation was applied to finalize sparse weights and remove pruning wrappers, followed by evaluation under TensorFlow Lite dynamic quantization to assess its effect on storage demand and inference latency.

Because of early stopping and the limited frequency of mask updates, the final sparsity ratio did not reach exactly 0.5. In practice, the average sparsity across layers ranged between 47% and 49%, reflecting the stability of the pruning schedule. Retaining the attention layers slightly reduced the achievable compression ratio but ensured interpretability of the temporal weighting process. Overall, this pruning–quantization pipeline achieved approximately 50% structural sparsity while keeping the total parameter count nearly constant, effectively lowering memory bandwidth and inference latency without sacrificing predictive accuracy or transparency (Tang *et al.*, 2022; Paula *et al.*, 2025).

It is worth noting that the static model size changed only marginally (0.207 MB → 0.204 MB) (Table 2). This occurs because TensorFlow Lite stores sparse weights in a dense tensor representation under dynamic quantization. Consequently, the main performance gains from pruning are reflected in inference speed and memory efficiency rather than in file-size reduction (Paula *et al.*, 2025).

For consistency, all models—**P1**, **P2**, and **P6**—were trained using the same dataset partition, identical early-stopping strategy, and uniform hyperparameter configuration, ensuring comparability of results. The structural characteristics of these models provide the basis for evaluating how pruning and quantization influence sparsity and computational efficiency (Table 2).

As shown in Table 2, pruning introduced roughly 50% unstructured sparsity while maintaining nearly the same parameter scale as the Tiny-GRU baseline (P2), resulting in only a minimal change in file size. The contribution of this study lies in applying a pruning–quantization workflow to ET_0 forecasting under edge-computing constraints, while intentionally keeping the attention layers unpruned to preserve model interpretability and temporal stability. A detailed evaluation of predictive accuracy, inference latency, and thermal performance is presented in Section 4.

(b) Dynamic range quantization (DRQ): After training the Tiny-GRU model (**P2**), dynamic range quantization (DRQ) was applied through the TensorFlow Lite framework to evaluate improvements in storage efficiency and inference latency on resource-limited edge platforms. This method follows the integer quantization approach introduced by Jacob *et al.*, (2018) and integrated into the TensorFlow Lite (TFLite) toolchain, which has since been extended in several lightweight RNN quantization studies targeting edge deployment (Wang & Kang, 2023). The underlying mechanism compresses FP32 weights into INT8 format while keeping activations and input–output operations in FP32, a design choice that maintains numerical stability and simplifies data handling during inference.

In this study, the trained Tiny-GRU model was exported using the TensorFlow Lite Converter to generate a .tflite executable that runs directly on embedded CPUs without additional modifications. Unlike standard post-training quantization (PTQ), DRQ does not rely on calibration data,

which helps reduce benchmarking complexity. The conversion process used the TensorFlow Lite Converter with the optimization flag tf.lite.Optimize.DEFAULT, and inference tests were conducted with a batch size of one to approximate realistic edge-inference scenarios. Because of its simplicity and robustness, DRQ was adopted as the default setting for evaluating inference performance on embedded and edge devices.

DRQ does not alter the network architecture; instead, it achieves efficiency purely through compressed weight representation. Pruning and DRQ are complementary: pruning reduces computational load by removing redundant weights, while DRQ decreases storage and memory bandwidth demands through precision compression. Applied together, they enable further efficiency gains for ET_0 forecasting on edge devices, representing one of the first studies to systematically benchmark this combined pipeline. The two techniques thereby provide efficiency improvements in benchmarking tasks while maintaining comparable predictive accuracy.

Optimization pathways and strategy combinations: To systematically evaluate the independent contributions and joint effects of different optimization strategies, nine configuration pathways (P1–P9) were designed, as summarized in Table 3. These pathways cover the progression from the baseline model (P1) and the lightweight starting point (P2), through individual compression strategies, to combinations of compression with enhancement, and finally to the full multi-strategy integration (P9). This design supports controlled benchmarking and facilitates engineering-oriented evaluation.

- Baseline reference: P1 (two-layer GRU, 128→64 hidden units).
- Lightweight starting point: P2 (Tiny-GRU, 64→32→16 hidden units).

The layer configuration (64→32→16) was determined empirically to achieve a balanced trade-off between prediction accuracy and computational efficiency.

Preliminary experiments with shallower or wider configurations resulted in either noticeable accuracy degradation or increased inference latency, while this structure provided a favorable compromise for edge deployment.

- Single compression: P3 (Tiny-GRU + DRQ), P6 (Tiny-GRU + pruning).
- Compression with enhancement: P4 (Tiny-GRU + temporal attention), P5 (Tiny-GRU + attention + DRQ), P7 (Tiny-GRU + attention + pruning), P8 (Tiny-GRU + pruning + DRQ).
- Multi-strategy integration: P9 (Tiny-GRU + attention + pruning + DRQ).

For example, pruning and quantization exhibit synergistic improvements in model compression and inference speed (Argüello Ron *et al.*, 2022). When applied within the lightweight Tiny-GRU backbone, temporal attention pooling adds only a small amount of computation but helps recover much of the accuracy that would otherwise be lost due to compression. This outcome highlights the practical importance of combining compression and enhancement strategies within a unified framework.

As shown in Table 3, the pathway design establishes a stepwise framework that enables controlled benchmarking across multiple optimization stages. The final configuration (**P9**) brings together structural reduction, temporal modeling, sparsification, and quantization, forming the basis for the cross-platform evaluation described in **Section 3.4**. In doing so, the framework directly addresses the challenges of ET_0 forecasting under edge-computing limitations and demonstrates both methodological soundness and practical engineering value. All models were trained using the Adam optimizer with a learning rate of 0.001, batch size of 32, and dropout rate of 0.2 to prevent overfitting. Early stopping with a patience of 10 epochs was employed to ensure convergence stability.

Table 2. Structural comparison of baseline GRU, Tiny-GRU, and pruned GRU models,

Model	Params (k)	Sparsity (%)	Model size (MB, theoretical)
Baseline (P1, standard GRU)	118	0	0.970
Tiny-GRU (P2, lightweight baseline)	25.3	0	0.207
Pruned (P6, Tiny-GRU + pruning/quant.)	25.6	≈50	0.204

Table 3. Definition of baseline and optimization pathways (P1–P9).

Path	Base architecture	Attention	Pruning	Quantization (DRQ)	Remarks
P1	GRU (128→64)	✗	✗	✗	Baseline GRU model
P2	Tiny-GRU (64→32→16)	✗	✗	✗	Lightweight starting point
P3	Tiny-GRU	✗	✗	✓	Single quantization
P4	Tiny-GRU	✓	✗	✗	Temporal attention pooling
P5	Tiny-GRU	✓	✗	✓	Attention with quantization
P6	Tiny-GRU	✗	✓	✗	Pruning (≈50% sparsity)
P7	Tiny-GRU	✓	✓	✗	Attention with pruning
P8	Tiny-GRU	✗	✓	✓	Pruning with quantization
P9	Tiny-GRU	✓	✓	✓	Full multi-strategy integration

Note: DRQ = Dynamic range quantization

Table 4. Hardware and runtime configurations of the experimental platforms.

Platform type	CPU (Frequency)	Memory	Operating system	Runtime configuration
Raspberry Pi 5	ARM Cortex-A76 (4 × 2.4 GHz)	8 GB LPDDR4X	Raspberry Pi OS (Bookworm, 64-bit, Linux Kernel 6.x.y)	Room temperature ≈ 26°C; no external cooling; official power supply
Intel i5-12450H (12th Gen)	8 cores (4P + 4E), 2.0–4.4 GHz	16 GB DDR4	Windows 11 (64-bit)	Room temperature ≈ 26°C; single-threaded CPU execution; GPU disabled

Experimental platforms and configurations

Hardware and software environment: To ensure reproducibility and fair comparison across platforms (Shan *et al.*, 2025), two hardware environments were used in this study. The Raspberry Pi 5 (Raspberry Pi Foundation, UK) served as the embedded edge-benchmarking platform, while an Intel® Core™ i5-12450H processor (12th Gen, Intel Corporation, USA) was used as a general-purpose reference system. The Raspberry Pi family is widely adopted for embedded AI benchmarking due to its balance between computational capability and accessibility, and is consistent with the EdgeMark evaluation framework (Hasanpour *et al.*, 2025; Pandey & Asati, 2023).

All model training was conducted on the Intel i5 platform, which was restricted to single-threaded CPU execution with GPU acceleration disabled in order to emulate embedded edge constraints (Lv *et al.*, 2025). The same configuration was applied during inference for consistency, while the Raspberry Pi 5 served as the physical testbed for edge deployment. This setup enabled direct measurement of inference latency, memory utilization, and thermal behavior under realistic operating conditions. The detailed hardware specifications and runtime configurations of both platforms are summarized in Table 4.

All experiments were executed in a standardized Python environment using TensorFlow for model training and TensorFlow Lite for deployment and inference. Identical software settings were applied across all configurations to ensure fair comparison. Detailed information on software versions, libraries, and runtime dependencies is provided in the Supplementary Materials.

Dataset and partitioning: Input sequences were generated using a sliding window of 12 time steps applied to six input variables, yielding a total of 26,267 valid samples. To preserve temporal integrity and prevent information leakage, the dataset was divided chronologically, with 80% (21,012 samples) used for training and the remaining 20% (5,254 samples) reserved for validation. The validation set was used both for early stopping during training and as the common benchmark set for cross-platform inference evaluation. As the primary objective of this study is benchmarking feasibility rather than generalization assessment, no independent test set was constructed, consistent with prior engineering-oriented benchmarking studies (Hasanpour *et al.*, 2025).

Model configurations and inference monitoring: Nine configuration pathways (P1–P9) were implemented to evaluate the independent and combined effects of lightweight architecture design, pruning, quantization, and temporal attention. All models were trained using the Adam optimizer with a fixed learning rate and early stopping, and subsequently exported to TensorFlow Lite (.tflite) format for deployment and inference on both platforms (Vindas *et al.*, 2025). Additional sensitivity tests with slight variations in learning rate (±10%) and dropout rate (0.1–0.3) resulted in negligible changes in validation performance (<1% RMSE variation), indicating that the Tiny-GRU configuration is relatively robust to minor hyperparameter perturbations.

All model variants were evaluated using sample-wise inference (batch size = 1) on the full validation set (n = 5,254). Inference latency was reported as the mean ± standard deviation. On the Raspberry Pi 5, CPU temperature was monitored as an indirect indicator of energy consumption using the *vcgencmd* utility, following established practices in embedded AI benchmarking (Mohamed *et al.*, 2022; Vindas *et al.*, 2025). Recorded thermal indicators included the initial temperature, rolling maximum temperature, temperature increase (ΔT), and steady-state statistics.

To promote transparency and reproducibility, all core code and deployment scripts will be made publicly available with a registered DOI, together with full dependency specifications and platform-related tools. Each reported result represents the average of two independent experimental runs, ensuring stability of the observed performance trends in accordance with FAIR data principles (Wilkinson *et al.*, 2016).

Evaluation metrics and decision tools

Accuracy metrics: Model performance was evaluated on the validation dataset (n = 5,254) using three widely adopted accuracy indicators: mean absolute error (MAE, $mm \cdot h^{-1}$), root mean square error (RMSE, $mm \cdot h^{-1}$), and the coefficient of determination (R^2). These metrics respectively characterize average prediction deviation, sensitivity to large errors, and goodness-of-fit, and were computed following standard definitions in the literature (Willmott & Matsuura, 2005). Together, they provide a comprehensive assessment of predictive accuracy and model reliability.

Efficiency metrics: Model efficiency was assessed from four complementary perspectives (Lv *et al.*, 2025). Inference latency ($ms \cdot sample^{-1}$) was measured on both the Raspberry Pi 5 and Intel i5 platforms. On the Raspberry Pi 5, latency was obtained under sample-wise inference across 100 repeated runs, with cold-start effects and outliers exceeding 3σ excluded. On the Intel i5 platform, latency was measured under full-batch inference, and results are reported as mean ± standard deviation (SD).

Model size (MB) was determined from the exported .tflite files to quantify storage requirements, while peak memory usage (MB) was monitored during execution to characterize runtime resource demand. In addition, total training time and the number of epochs required for convergence were recorded as indicators of optimization efficiency.

Thermal performance metrics: As the Raspberry Pi 5 lacks onboard power monitoring, CPU temperature was adopted as a proxy indicator of energy consumption, following established edge benchmarking practices (Mohamed *et al.*, 2022). Although not a direct measure of power draw, CPU temperature provides a stable and repeatable indicator when ambient conditions are controlled.

Four thermal indicators were defined: the initial temperature prior to inference, the rolling maximum temperature computed using a five-sample moving average, the temperature rise ($\Delta T = T\_peak - T\_initial$), and the steady-state mean and standard deviation, which together characterize thermal dynamics and stability during sustained inference.

Statistical and visualization tools: Because inference latency and MAE did not satisfy normality assumptions, the Kruskal–Wallis test was applied to assess statistical significance, whereas memory usage, which satisfied both normality and homoscedasticity, was analyzed using one-way ANOVA. A significance level of $p < 0.001$ was adopted throughout. Statistical analysis focused on global significance, and post-hoc pairwise comparisons were not conducted to maintain interpretive consistency.

To support multi-objective benchmarking, several visualization tools were employed. Pareto front plots were used to illustrate trade-offs among RMSE, inference latency, and memory usage (Deb *et al.*, 2002). Heatmaps were constructed using normalized metrics, with lower-is-better indicators (RMSE, latency, memory usage, and ΔT) inverted and higher-is-better indicators (R^2) retained to ensure consistent interpretation (Zhen *et al.*, 2020). In addition, bubble plots were generated using RMSE and latency as coordinate axes, bubble size representing model size, and color encoding the P1–P9 optimization paths. Collectively, these statistical and visualization tools form a reproducible, multi-dimensional evaluation framework for benchmarking ET_0 forecasting models under edge-computing constraints.

Summary and methodological contributions: This chapter presented a comprehensive multi-metric benchmarking framework designed for ET_0 forecasting under edge-computing constraints. The methodology was organized into four major components. The first focused on data preprocessing, where a high-resolution, single-station dataset was carefully cleaned, normalized, and prepared to ensure stable model training. The second addressed model architectures and optimization pathways, moving from a baseline GRU toward a lightweight Tiny-GRU and further integrating temporal attention, pruning, and dynamic quantization to examine both independent and combined optimization effects. The third covered experimental platforms and configurations, in which benchmarking was performed on Intel i5 and Raspberry Pi 5 devices under consistent training and inference conditions to enable fair cross-platform comparison. The final component introduced evaluation metrics and decision-support tools, incorporating measures of predictive accuracy, computational efficiency, thermal behavior, and statistical testing, together with multi-objective visualization methods such as Pareto fronts, heatmaps, and bubble plots (Deb *et al.*, 2002; Zhen *et al.*, 2020).

By jointly considering accuracy, efficiency, and thermal characteristics, the proposed framework extends beyond traditional evaluation protocols and establishes a reproducible foundation for benchmarking edge-AI models. Unlike most previous ET_0 forecasting studies that focused mainly on predictive accuracy, this design explicitly incorporates the computational limitations of edge environments, emphasizing both its engineering practicality and methodological relevance. The multi-metric benchmarking framework developed here thus provides the basis for the cross-platform analysis presented in Section 4.

Results and Discussion

Accuracy and latency performance: The predictive accuracy and inference latency of the nine model configurations (P1–P9) were evaluated on two hardware platforms: the Raspberry Pi 5 as a representative embedded edge device, and an Intel i5 system configured in single-threaded CPU mode to emulate constrained execution conditions (Hasanpour *et al.*, 2025; Vindas *et al.*, 2025). As accuracy differences between platforms were negligible (RMSE discrepancy < 0.0005), the accuracy results reported in Table 5 correspond to those obtained on the Raspberry Pi 5. Inference latency was analyzed separately for each platform to examine the influence of hardware characteristics on real-time performance.

Table 5. Comparative prediction accuracy of nine GRU-based model paths (P1–P9) evaluated on the Raspberry Pi 5.

Model	MAE	RMSE	R^2
P1	0.0153	0.0226	0.9792
P2	0.0223	0.0278	0.9685
P3	0.0134	0.0208	0.9824
P4	0.0130	0.0203	0.9832
P5	0.0151	0.0228	0.9788
P6	0.0133	0.0210	0.9820
P7	0.0145	0.0218	0.9806
P8	0.0133	0.0211	0.9819
P9	0.0156	0.0230	0.9785

Overall, the optimized models consistently outperformed the baseline GRU (P1) in terms of predictive accuracy. Among all configurations, the attention-augmented Tiny-GRU (P4) achieved the best overall accuracy (RMSE = 0.0203, R^2 = 0.9832), while the aggressively compressed Tiny-GRU (P2) showed the lowest accuracy (RMSE = 0.0278, R^2 = 0.9685). Notably, P3 preserved high predictive accuracy (RMSE = 0.0208, R^2 = 0.9824), exhibiting only marginal degradation relative to P4. These results indicate that architectural design and optimization strategy, rather than computational hardware, primarily determine forecasting accuracy.

Inference latency exhibited clear and systematic trends across configurations and platforms, as illustrated in Fig. 2. Absolute latency values on the Intel i5 were lower owing to its higher clock frequency; however, the relative ranking of model performance closely matched that observed on the Raspberry Pi 5 (Mohamed *et al.*, 2022). As model compression increased, inference latency generally decreased, confirming the effectiveness of lightweight design and compression strategies for real-time execution.

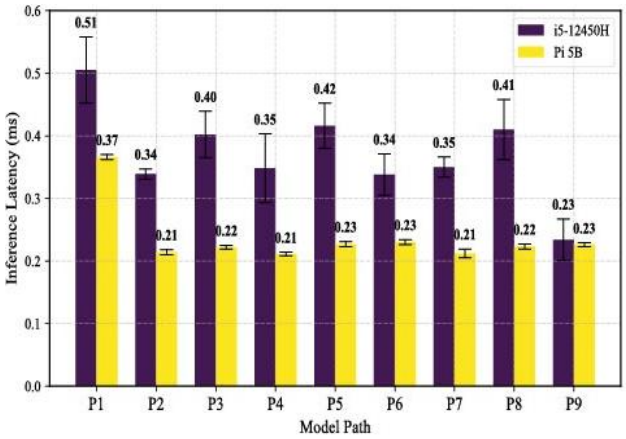

Fig. 2. Comparison of inference latency for nine GRU-based model paths on Intel i5 and Raspberry Pi 5 platforms.

From a trade-off perspective, three representative performance orientations can be identified. The efficiency-oriented configuration (P2) achieved the lowest latency but incurred a noticeable loss in accuracy. The accuracy-oriented configuration (P3) retained near-optimal predictive performance while exhibiting moderately higher latency. The balanced configuration (P4) provided the most favorable compromise, combining the highest accuracy with low inference latency (0.211 ms on the Raspberry Pi 5). Statistical testing confirmed that differences among these performance groups were significant ($p<0.001$; see Section 4.6).

The Raspberry Pi 5 exhibited a narrower latency range (0.211–0.366 ms) and greater temporal consistency. By contrast, the Intel i5 showed larger variability, which can be attributed to operating-system scheduling overhead under emulated edge conditions (Shahab *et al.*, 2025). When considered together with the thermal behavior discussed in Section 4.5, these findings suggest that the Raspberry Pi 5 provides a more stable benchmarking environment for evaluating edge-oriented ET_0 forecasting models.

In summary, integrating lightweight GRU architectures with temporal attention mechanisms enables models to achieve both high accuracy and low latency under resource constraints. This balance is particularly important for real-time ET_0 estimation in smart paddy irrigation, where timely and reliable predictions directly support irrigation scheduling and water-use efficiency.

Memory usage and model file size analysis: Memory usage and model file size were analyzed for the nine configurations (P1–P9) to assess their feasibility for deployment on resource-limited edge devices. This aspect is especially critical for agricultural IoT systems, where available storage and runtime memory are often tightly constrained (Hasanpour *et al.*, 2025; Shahab *et al.*, 2025). The main results are summarized in Table 6 and visualized in Fig. 3(a–c).

Table 6. Memory usage and model file size of nine GRU-based model paths (P1–P9) on the Raspberry Pi 5.

Model	Model size (MB)	Load memory (MB)	Real-time memory (MB)	Peak memory (MB)
P1	0.413	2.05	509.41	509.94
P2	0.207	2.56	509.83	510.09
P3	0.135	2.55	508.95	509.17
P4	0.207	2.53	509.28	509.69
P5	0.138	2.55	508.58	508.86
P6	0.204	2.53	508.72	509.14
P7	0.207	2.56	508.62	509.28
P8	0.135	2.56	509.15	509.86
P9	0.138	2.56	509.07	510.02

Overall, the adoption of lightweight architectures, pruning, and quantization led to consistent reductions in both storage footprint and runtime memory demand. While model loading memory showed modest variation across configurations (approximately 2.05–2.56 MB), peak memory usage exhibited clearer differentiation. The baseline model (P1) consumed the largest peak memory (509.94 MB), whereas the most memory-efficient configuration (P5) reduced peak usage to 508.86 MB. The absolute reduction per model was limited. However, such savings become increasingly relevant in multi-node deployments typical of distributed agricultural monitoring systems.

Model file size demonstrated more pronounced improvements. The uncompressed baseline model (P1) occupied 0.413 MB, while the most compact configuration (P5) achieved a file size of 0.138 MB, corresponding to a 66.6% reduction. This substantial decrease highlights the dominant contribution of post-training dynamic range quantization to storage efficiency, exceeding the effect of pruning alone (Mohamed *et al.*, 2022).

Across all configurations, memory reductions were achieved without compromising predictive accuracy, underscoring the effectiveness of the proposed multi-strategy optimization framework. Statistical analysis (Section 4.6) confirmed that the observed differences in memory usage and model size were significant ($p<0.001$), reinforcing the robustness of the benchmarking results.

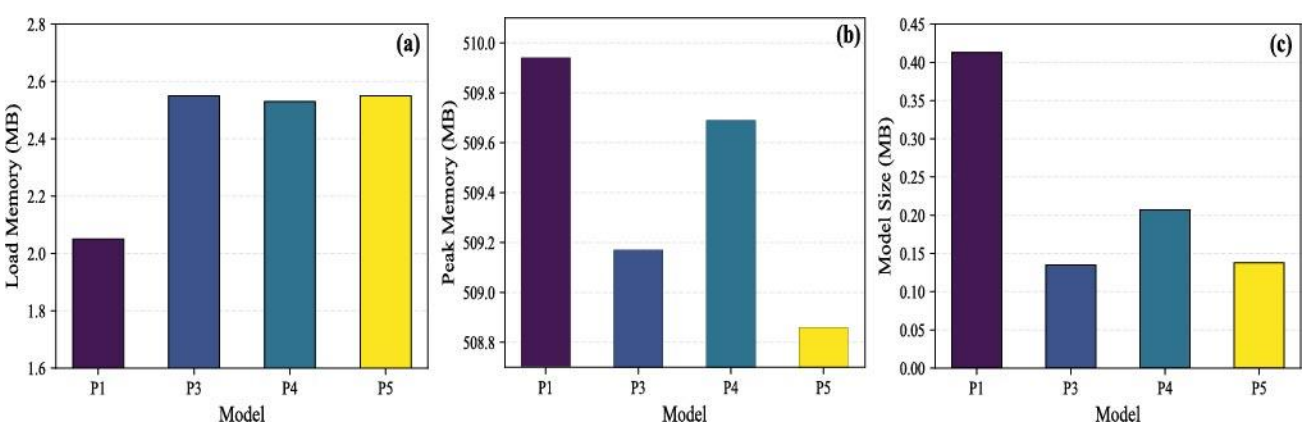

Fig. 3. Memory usage and model file size of nine model paths on the Raspberry Pi 5: (a) model loading memory, (b) peak runtime memory, and (c) model file size.

Taken together, these findings demonstrate that combining lightweight network design with quantization-driven compression substantially enhances the deployability of ET_0 forecasting models on edge devices. In particular, configuration P5 emerged as the most memory-efficient path, while other optimized configurations offered intermediate trade-offs between memory demand and predictive performance, providing flexibility for diverse edge deployment scenarios.

Multi-objective trade-offs and decision support: In edge-AI deployments, trade-offs between predictive accuracy, inference latency, and memory usage are almost inevitable—especially in agricultural IoT systems, where models must operate under strict resource limitations while still delivering near real-time performance (Hasanpour *et al.*, 2025; Shahab *et al.*, 2025). Unlike empirical models such as the FAO Penman–Monteith equation, which rely on simplified linear relationships among climatic variables, the GRU-based framework captures nonlinear dependencies and temporal interactions, thereby improving ET_0 forecasting under dynamic environmental conditions. To help interpret these trade-offs more intuitively, this study introduces a visualization-based decision-support framework that combines two-dimensional bubble plots, multi-metric heatmaps, and three-dimensional Pareto fronts (Zhen *et al.*, 2020). Together, these visualization tools reveal the interactions among performance metrics and offer practical guidance for selecting models according to specific application priorities.

Trade-off characteristics: The two-dimensional distribution of RMSE and inference latency shows a clear clustering trend (Fig. 4a). The baseline model (P1) recorded the highest latency (0.366 ms) with only a modest accuracy benefit (RMSE = 0.0226). By contrast, the optimized configurations (P2–P9) were concentrated in the lower-left region of the plot, with latency values between 0.211 and 0.230 ms and RMSE ranging from 0.0203 to 0.0230. This clustering suggests that lightweight optimization strategies were effective in reducing computational cost while maintaining predictive accuracy. Fig. 4(b) shows a magnified view that emphasizes the differences among individual strategies. P4 delivered the best overall performance, achieving the lowest latency (0.211 ms) together with the highest accuracy (RMSE = 0.0203). P3 and P8 also performed strongly, maintaining high accuracy (RMSE = 0.0208 and 0.0211, respectively) while producing the smallest model files (0.135 MB), which makes them particularly suitable for storage-limited hardware. P5, with a file size of 0.138 MB and moderate latency (0.227 ms), represented a memory-efficiency-oriented configuration. The pruning-based variants (P6–P7) exhibited more balanced trade-offs between latency and compression. When compared with the pathway definitions summarized in Table 3 (Section 3.3), both P4 and P9 also showed advantages in MAE, reinforcing the view that different optimization paths can complement each other across accuracy dimensions.

At a broader multi-metric scale, Fig. 5 highlights several consistent performance patterns. Relative to Tiny-GRU (P2), P4 achieved a 26.98% reduction in RMSE while maintaining nearly identical latency (a decrease of just 1.40%). P5 also improved accuracy over P2, reducing RMSE by 17.99%, though this came with a modest 6.07% increase in latency—illustrating the typical trade-off between precision and efficiency.

From a benchmarking perspective, P3 and P6 achieved file-size reductions of 67.31% and 50.61%, respectively, accompanied by lower mean CPU temperatures (2.0°C and 1.8°C decreases). These outcomes point enhanced thermal stability and reduced power stress, both of which mitigate overheating risks (Mohamed *et al.*, 2022)—a particularly important factor for sustained operation in field-based IoT devices without active cooling mechanisms.

Pareto front analysis: Figure 6 illustrates the Pareto-optimal set obtained under three simultaneous objectives. Among the nine model paths, only P3, P4, and P5 appeared on the Pareto front. Specifically, P3 achieved the lowest RMSE (0.0208), P4 delivered the lowest inference latency (0.211 ms), and P5 provided the lowest peak memory consumption (508.86 MB)—representing a 1.08 MB reduction compared with P1. Accordingly, P3, P4, and P5 can be regarded as accuracy-, latency-, and memory-oriented optimization pathways, respectively, representing distinct priorities for practical deployment under different edge-computing scenarios. For visualization clarity, a consistent color-coding scheme was applied across all Pareto-front and multi-metric figures, with each optimization path (P1–P9) represented by the same color throughout the manuscript. The remaining models were dominated in at least one performance dimension.

Taken together, these findings confirm that no single configuration excels across all metrics; rather, model selection must be aligned with the computational and operational constraints of the intended deployment environment. Statistical testing using one-way ANOVA further supported the significance of these trade-offs, with corresponding F-statistics and *p*-values summarized in Section 4.6.

Decision-support method and practical value: Building on the Pareto-front analysis, the proposed visualization framework provided an intuitive and systematic decision-support mechanism for selecting ET_0 forecasting models under edge-computing constraints. By jointly considering accuracy, inference latency, and memory usage, the framework enables users to identify configurations that best match specific deployment priorities without relying on a single performance metric. This multi-objective perspective enhances transparency and robustness in model selection, particularly in resource-limited agricultural IoT environments. The results demonstrated that reliable benchmarking and informed decision-making can be achieved directly on edge devices, supporting practical deployment in smart irrigation systems. More broadly, the framework establishes a transferable methodological basis for multi-metric evaluation and optimization in edge-based time-series forecasting, with potential applicability to other environmental monitoring and edge–cloud collaborative scenarios.

In practical deployment, the Tiny-GRU framework can be implemented on low-cost embedded devices (e.g., Raspberry Pi, ESP32) connected to field sensors measuring

air temperature, humidity, wind speed, and soil moisture. In the present study, benchmarking was conducted on Raspberry Pi 5 and a constrained Intel i5 platform. Ultra-low-power MCUs such as the ESP32 were not experimentally evaluated because the full TensorFlow Lite (.tflite) runtime could not be executed reliably under the available on-device memory constraints (i.e., insufficient RAM/flash for allocating model tensors and intermediate buffers). The quantized model performs near-real-time ET_0 prediction locally and transmits results via lightweight IoT protocols such as MQTT or LoRa for cloud–edge coordination. This configuration supports autonomous irrigation scheduling and enhances the applicability of the framework in real-world smart paddy irrigation systems.

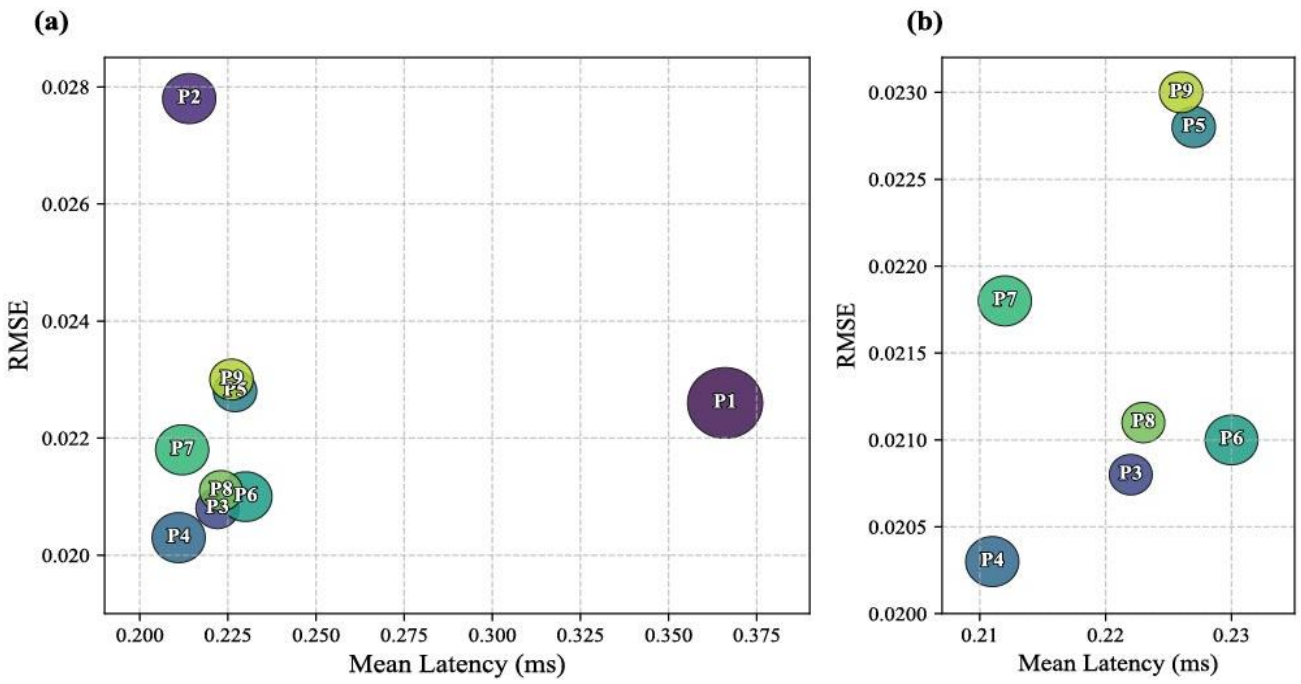

Fig. 4. Two-dimensional RMSE–latency bubble distribution of nine model paths on the Raspberry Pi 5. (a) Overall distribution of all models; (b) zoomed-in view excluding P1 for clarity. Bubble size represents model file size.

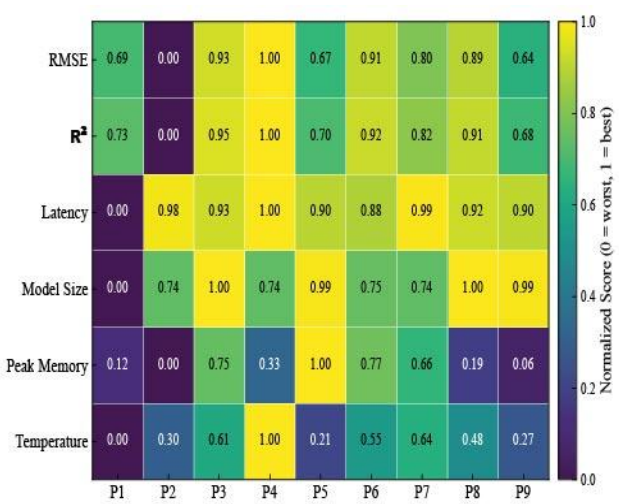

Fig. 5. Heatmap comparison of six normalized performance metrics, including RMSE, R^2, inference latency, model file size, peak memory usage, and average operating temperature, evaluated on the Raspberry Pi 5 platform. Values represent normalized means with error bars denoting ±1 standard deviation.

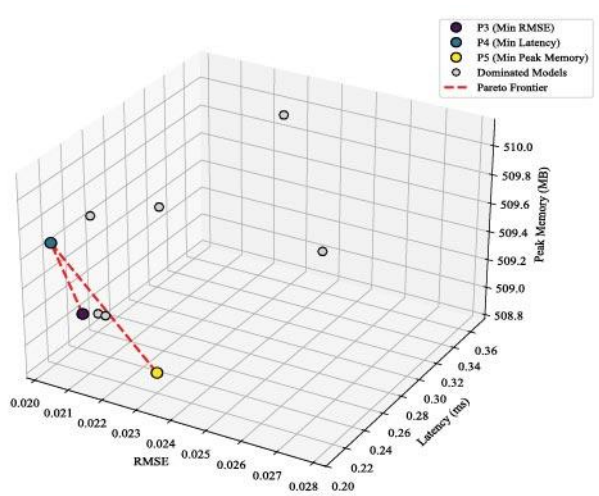

Fig. 6. Three-dimensional Pareto front analysis of RMSE, inference latency, and peak memory evaluated on the Raspberry Pi 5 platform. Points represent nine model paths (P1–P9), with a consistent color scheme used to denote each configuration across all figures, and Pareto-optimal solutions (P3, P4, P5) highlighted for clarity.

Continuous inference stability and thermal performance: Figure 7 compares the stability characteristics of continuous inference for the baseline model (P1), the intermediate path (P3), and the optimized path (P4) on the Raspberry Pi 5 platform. For visual clarity, the temperature curves in subfigure (a) were smoothed using a moving average, latency curves in subfigure (b) were down-sampled at ten-step intervals, and subfigure (c) displays peak memory usage. The corresponding quantitative data are summarized in Table 7.

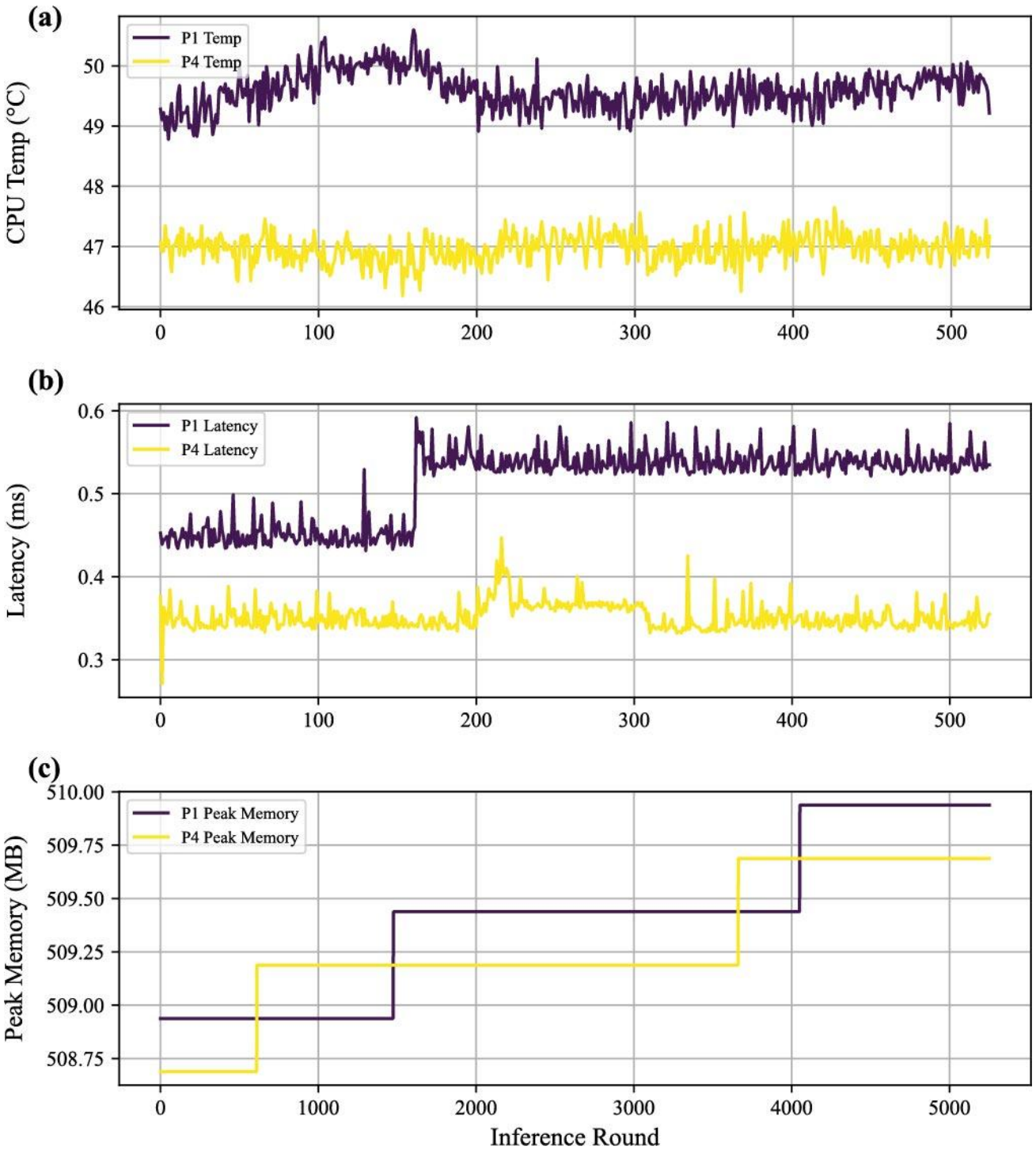

Fig. 7. Comparison of continuous inference stability for P1, P3, and P4 on the Raspberry Pi 5: (a) CPU temperature variation (smoothed using a moving average); (b) inference latency variation (down-sampled at ten-step intervals); (c) peak memory usage.

Overall, P4 consistently maintained the lowest CPU temperature (47.0°C) during inference and exhibited both low and stable latency (0.35 ms) with memory consumption comparable to P1 (509.3 MB). By contrast, P1 reached a higher temperature (49.6°C) and showed larger latency fluctuations (0.51 ms), whereas P3 fell between the two, recording 47.8°C, 0.33 ms, and 508.9 MB. These results imply that incremental optimization enhanced stability but did not match the robustness achieved by P4.

Differences in temperature, latency, and memory usage were statistically verified (see Section 4.6). In addition, recent studies have validated the use of CPU temperature as a practical proxy for power consumption in edge energy-efficiency assessments (Benoit-Cattin *et al.*, 2020; Jin *et al.*, 2022). This is because, under sustained inference workloads on embedded CPUs without on-board power sensors, increased computational activity leads to proportional rises in dynamic power dissipation and thermal accumulation. As a result, temperature variation serves as a reliable indirect indicator of relative energy efficiency across model configurations.

Taken together, P4 demonstrated superior thermal stability and consistent inference behavior, suggesting its suitability for long-duration benchmarking and deployment in thermally constrained edge environments.

Table 7. Mean (± standard deviation, SD) of CPU temperature, inference latency, and peak memory during continuous inference on the Raspberry Pi 5.

Model	Temperature (°C) mean ± SD	Latency (ms) mean ± SD	Peak memory (MB) mean ± SD
P1	49.6 ± 1.5	0.51 ± 0.04	509.4 ± 0.4
P3	47.8 ± 1.0	0.33 ± 0.04	508.9 ± 0.3
P4	47.0 ± 1.0	0.35 ± 0.02	509.3 ± 0.3

Note: Each value represents the mean ± SD across 100 continuous inference runs. CPU temperature was monitored at 1 Hz using the built-in system utilities of Raspberry Pi OS, while inference latency and memory usage were recorded via *psutil*

Table 8. Statistical significance analysis of model performance across optimization paths.

Model	MAE	Latency (ms) mean ± SD	Peak memory (MB) mean ± SD
P1	0.016	0.366 ± 0.004	509.382 ± 0.342
P2	0.022	0.214 ± 0.004	509.821 ± 0.314
P3	0.015	0.222 ± 0.003	508.965 ± 0.303
P4	0.012	0.211 ± 0.003	509.314 ± 0.322
P5	0.014	0.227 ± 0.004	508.642 ± 0.336
P6	0.014	0.230 ± 0.004	508.676 ± 0.329
P7	0.017	0.212 ± 0.007	508.564 ± 0.506
P8	0.014	0.223 ± 0.004	509.157 ± 0.434
P9	0.012	0.226 ± 0.003	509.082 ± 0.669

Note: MAE = Mean absolute error; latency and memory usage are reported as mean ± standard deviation over 99 inference samples. Global significance tests: Latency (Kruskal–Wallis, $p<0.001$); Memory (ANOVA, $p<0.001$); MAE (Kruskal–Wallis, $p<0.001$)

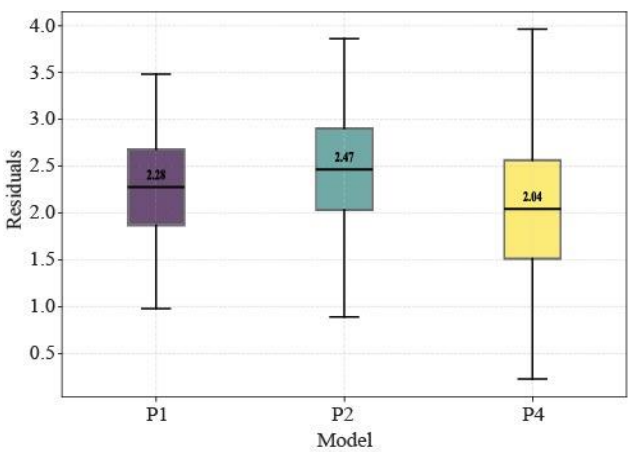

Fig. 8. Residual distributions of P1 (baseline GRU), P2 (lightweight Tiny-GRU), and P4 (attention-enhanced Tiny-GRU) under extreme weather conditions (T ≥35 °C, soil moisture ≤35%).

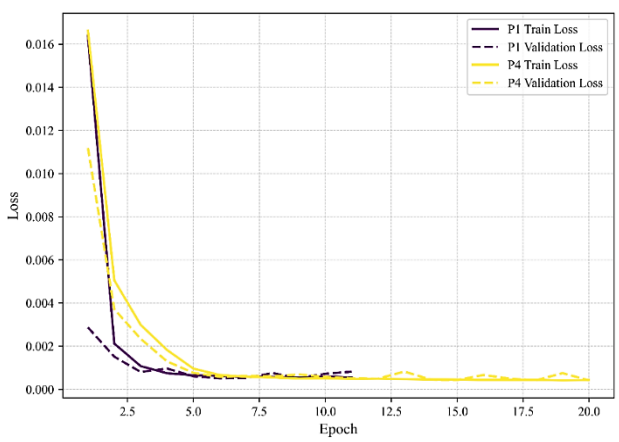

Fig. 9. Convergence curves of P1 (baseline) and P4 (optimized): (a) training-loss variation; (b) validation-loss progression.

Strategy impact analysis: attention mechanism example: Residual distribution analysis was conducted under extreme meteorological conditions (air temperature ≥ 35°C and soil moisture ≤ 35%) to evaluate model robustness and bias behavior. All experiments were conducted in a controlled laboratory environment (ambient temperature 25 ± 1°C) with passive cooling and no external airflow, ensuring that temperature variations across models reflected computational load rather than ambient fluctuations. As shown in Figure 8, the attention-enhanced Tiny-GRU (P4) exhibited the lowest median residual (2.04) and fewer outliers compared with the baseline GRU (P1, median = 2.28) and the lightweight Tiny-GRU (P2, median = 2.47). The mean residuals further indicate that P4 (−0.0015) and P1 (−0.0029) remained effectively unbiased, whereas P2 showed a tendency toward overestimation (0.0181). Although the interquartile range of P4 (1.05) was slightly larger than those of P1 (0.81) and P2 (0.87), its residuals were more symmetrically centered around zero, suggesting improved robustness rather than instability. Overall, these quantitative results demonstrate that temporal attention enhances model robustness and bias control under extreme climatic variability. Additional residual analyses for all configurations and platforms are provided in the Supplementary Materials.

Model convergence and statistical significance: Model convergence and statistical validation are essential for confirming the effectiveness of the proposed optimization strategies. This section examines training stability and generalization behavior, followed by hypothesis testing across the nine GRU configurations.

As illustrated in Fig. 9, both the baseline model (P1) and the optimized path (P4) displayed a steep decline in training loss during the first three epochs, after which the curves converged smoothly without pronounced oscillations. The validation loss reached a lower minimum for P4 (0.000412) than for P1 (0.000517), suggesting stronger generalization ability. Differences in convergence stability were minimal—peak-to-peak fluctuations were 3.0×10^{-4} for P1 and 3.4×10^{-4} for P4—indicating that optimization did not compromise training consistency.

To further substantiate performance differences among the models, statistical tests were conducted on three core indicators: MAE, inference latency, and peak memory usage (see Table 8). Since MAE and latency failed to meet the assumption of normality, they were evaluated using the Kruskal–Wallis test, whereas memory usage, which satisfied homoscedasticity, was analyzed via one-way ANOVA. All three metrics revealed statistically significant differences ($p<0.001$).

Statistical significance was evaluated across nine model configurations (P1–P9). Kruskal–Wallis tests were used for MAE and latency, while one-way ANOVA was used for memory usage. Only global significance was assessed ($p<0.001$) without post-hoc multiple comparison correction.

Beyond statistical significance, the observed differences also have practical implications. A reduction in RMSE of 0.003–0.005 mm·h^{-1} corresponds to an irrigation scheduling error of approximately 0.4–0.6 mm per hour, which can accumulate to 10–12 mm per day for water-sensitive crops. Similarly, lowering inference latency from 0.366 ms to 0.211 ms enhances real-time control responsiveness and overall water-use efficiency.

In terms of performance, P4 and P9 achieved the lowest MAE (0.012), while P2 performed the worst (0.022), indicating that excessive compression may hinder predictive accuracy. For efficiency, P4 again yielded the shortest latency (0.211 ± 0.003 ms). Although P7 recorded the lowest mean memory consumption (508.564 MB), its variability (±0.506 MB) was relatively high, suggesting reduced stability. P5, by contrast, showed a more consistent balance (508.642 ± 0.336 MB), making it the most memory-efficient configuration.

Taken together, the optimized paths demonstrated measurable and statistically validated improvements across key performance dimensions. Each optimization component—Tiny-GRU design, temporal attention, and pruning—contributed distinct advantages, offering practical flexibility for deployment in resource-limited environments. Among them, P4 achieved the most balanced combination of convergence stability and predictive accuracy, underscoring its robustness and practical relevance.

Conclusion and Future Work

This study developed a Tiny-GRU framework specifically designed for edge computing, combining temporal attention, pruning, and quantization within a cross-platform benchmarking system that integrates multi-metric evaluation and decision-support visualization tools. Cross-platform experiments conducted on the Raspberry Pi 5 and Intel i5 platforms showed that no single configuration achieved optimal performance across all criteria; rather, each pathway demonstrated distinct strengths. For example, P3 achieved an RMSE of 0.0208 with a 67% reduction in model size, P4 delivered the fastest inference speed (0.211 ms per sample) while lowering mean CPU temperature by 2.6°C, and P5 achieved the lowest memory footprint (508.86 MB) with a 66.6% file-size reduction. Statistical analyses (p<0.001) confirmed that these differences were highly significant, supporting the robustness of the proposed framework.

The framework effectively visualized trade-offs among accuracy, latency, model size, memory usage, and thermal stability through Pareto-front and heatmap-based analyses, offering agricultural IoT practitioners a transparent and practical reference for model selection. The results verified the feasibility, stability, and adaptability of optimized GRU models on embedded platforms, while also demonstrating that CPU temperature serves as a reliable and cost-effective proxy for energy consumption in devices without direct power sensors. From a practical irrigation-management perspective, the sub-millisecond inference latency achieved by the optimized Tiny-GRU configurations enables real-time ET_0 estimation on distributed edge devices, allowing large numbers of IoT nodes to perform local prediction and irrigation control simultaneously without centralized computation. This capability reduces communication overhead, improves system responsiveness, and supports timely irrigation scheduling in large-scale smart paddy irrigation networks**.** Beyond controlled experiments, the benchmarking methodology proved reproducible for ET_0 prediction in smart paddy irrigation, contributing to broader efforts toward sustainable and precision agriculture.

Nevertheless, a few limitations remain. Power consumption was inferred indirectly from temperature observations rather than measured directly using hardware-based sensors. The predictive horizon was restricted to single-step forecasting, and the study did not test cross-regional generalization under varying climatic and crop conditions. In addition, the experimental dataset was obtained from a single meteorological station, no independent field-scale or lysimeter-based measurements were collected for external validation, and full-scale field deployment in operational irrigation systems was beyond the scope of the present study.

Future research will focus on multi-step and multi-task forecasting, cross-regional transferability, and the integration of these models into edge–cloud digital twin systems. Specifically, future work will extend the proposed framework toward (i) multi-step ET_0 forecasting to support longer-term irrigation planning, (ii) federated learning strategies that enable collaborative model training across distributed farms while preserving data privacy, and (iii) adaptive quantization and runtime-aware compression schemes that dynamically adjust model precision according to device load and environmental conditions. In addition, integrating the Tiny-GRU framework into edge–cloud digital twin systems and conducting field-level deployment studies will be essential steps toward large-scale, real-world smart irrigation applications. These extensions will help move from isolated point-level optimization toward system-level agricultural intelligence, providing the technological foundation for next-generation smart irrigation management. Moreover, the Tiny-GRU framework can be readily adapted to other environmental prediction tasks such as soil-moisture estimation, evapotranspiration mapping, or on-farm energy-demand forecasting, demonstrating its scalability beyond paddy-irrigation systems.

List of abbreviations

The following abbreviations are used in this manuscript:

AI: Artificial Intelligence
ANOVA: Analysis of Variance
ARM: Advanced RISC Machine
CMA: China Meteorological Administration
DRQ: Dynamic Range Quantization
ET_0: Reference Crop Evapotranspiration
FAO-56: Food and Agriculture Organization Penman–Monteith Method
FP32: 32-bit Floating Point Precision
GPU: Graphics Processing Unit
GRU: Gated Recurrent Unit
IoT: Internet of Things
LSTM: Long Short-Term Memory
MAE: Mean Absolute Error
MACs: Multiply–Accumulate Operations
MCU: Microcontroller Unit
PTQ: Post-Training Quantization
QAT: Quantization-Aware Training
R^2: Coefficient of Determination
RNN: Recurrent Neural Network
RMSE: Root Mean Squared Error
SD: Standard Deviation
TCN: Temporal Convolutional Network
TFLite: TensorFlow Lite
Tiny-GRU: Lightweight Gated Recurrent Unit Model Proposed in This Study

Declarations

Availability of Data and Materials: The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

Competing Interests: The author(s) declared no potential conflicts of interest with respect to the research, author - ship, and/or publication of this article.

Funding: This work was supported by Universiti Teknologi Malaysia (UTM) through the UTM Encouragement Research Grant (Vot No. Q.J130000.3823.31J59), and by the **"Double First-Class" Gongga Discipline Construction Program of Sichuan Province** for disciplinary development funding support.

Authors' Contributions: Kai Luo designed and implemented the study, developed the models, conducted the experiments, analyzed the results, and drafted the manuscript. Lim Cheng Siong, Mohamad Hafis Izran Bin Ishak, and Mohd Saiful Azimi Mahmud provided supervision throughout the research process. They offered technical guidance, validated the methodology, and advised the first author during model development, deployment evaluation, and manuscript refinement. Xue Yang supported data collection and preprocessing, contributed to environmental sensing consultation, and assisted the first author in shaping the analysis framework. All authors read and approved the final manuscript.

Acknowledgements

The authors would like to acknowledge the support provided by Chengdu College of University of Electronic Science and Technology of China and the Faculty of Electrical Engineering, Universiti Teknologi Malaysia. The collaboration between Chinese and Malaysian institutions played a vital role in facilitating this research. The authors also express sincere appreciation to Lim Cheng Siong, Mohamad Hafis Izran Bin Ishak, and Mohd Saiful Azimi Mahmud for their valuable guidance and supervision provided to the first author throughout the research process. Finally, the authors would like to thank all team members for their dedicated efforts and contributions to the successful completion of this work.

References

Alandjani, G. 2024. Optimizing malware detection for IoT and edge environments with quantization awareness. *IEEE Access*, 12: 166776-166791.

Allen, R.G., L.S. Pereira, D. Raes and M. Smith. 1998. Crop evapotranspiration – Guidelines for computing crop water requirements (FAO Irrigation and Drainage Paper No. 56). Rome: Food and Agriculture Organization of the United Nations. https://www.fao.org/4/x0490e/x0490e00.htm

Argüello Ron, D., P.J. Freire, J.E. Prilepsky, M. Kamalian-Kopae, A. Napoli and S.K. Turitsyn. 2022. Experimental implementation of a neural network optical channel equalizer in restricted hardware using pruning and quantization. *Sci. Rep.*, 12(1): 8713.

Bazargani, M., S.T. Esfahani, B. Heidari, R. Hosseinzadeh Shabestary and M. Forghani. 2025. AFedSLL-LDL: A framework based on federated self-supervised learning and lightweight deep learning for attack detection in serverless edge computing. *Clust. Comput.*, 28: 975.

Benoit-Cattin, T., D. Velasco-Montero and J. Fernández-Berni. 2020. Impact of thermal throttling on long-term visual inference in a CPU-based edge device. *Electronics*, 9(12): 2106.

Cai, C., Q. Shi, X. Cheng, T. Yang and S. Hou. 2025. Short-term offshore wind power forecasting based on dynamic trend clustering of meteorological factors. *IEEE Trans. Sustain. Energy*, 74: 11178095.

Chen, K. and L. Xiao. 2025. Energy-efficient privacy-preserving AI models for real-time health monitoring in mobile IoT networks for professional sports applications. *Discov. Appl. Sci.*, 7: 1025. https://doi.org/10.1007/s42452-025-07626-6

Chen, L., G. Li, G. Huang and Q. Zhao. 2023. A lightweight model using frequency, trend and temporal attention for long sequence time-series prediction. *Neural Comp. Appl.*, 35: 21291-21307.

Cheng, Y., Z. Liu and Y. Morimoto. 2020. Attention-based SeriesNet: An attention-based hybrid neural network model for conditional time series forecasting. *Information*, 11(6): 305.

Chiranjeevi, M. and A. Chavan. 2025. Adversarial error mitigation technique for sensor fusion framework on dynamic edge computing platform. *Arab. J. Sci. Eng.*, https://doi.org/10.1007/s13369-025-10671-3

Deb, K., A. Pratap, S. Agarwal and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.*, 6(2): 182-197.

Farman, H., M.A. Hussain, S. Hassan, S. Shaikh and K. Ali. 2025. Activation function impact on rainfall prediction: comparative insights across ML and DL architectures. *Model. Earth Syst. Environ.*, 11: 451.

Guo, R., Z. Yue, Y. Wang, H. Li, T. Hu, Y. Wang and S. Yin. 2025. A 28-nm 28.8-TOPS/W attention-based NN processor with correlative CIM ring architecture and dataflow-reshaped digital-assisted CIM array. *IEEE J. Solid-State Circuits*, 60: 332-346. https://doi.org/10.1109/JSSC.2024.3419808

Hasanpour, M.A., M. Kirkegaard and X. Fafoutis. 2025. EdgeMark: An automation and benchmarking system for embedded artificial intelligence tools. *J. Syst. Archit.*, 167: 103488. https://doi.org/10.1016/j.sysarc.2025.103488

He, L. and L. Rosa. 2023. Solutions to agricultural green water scarcity under climate change. *PNAS Nexus*, 2(4): pgad117. https://doi.org/10.1093/pnasnexus/pgad117

Jacob, B., S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam and D. Kalenichenko. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 2704-2713. https://doi.org/10.1109/CVPR.2018.00286

Jin, C., X. Bai, X. Zhang, X. Xu, Y. Tang and C. Zeng. 2022. A measurement-based power consumption model of a server by considering inlet air temperature. *Energy*, 261: 125126.

Kang, J., J. Park, S. Choi and J. Sim. 2024. Q-LAtte: An efficient and versatile LSTM model for quantized attention-based time series forecasting in building energy applications. *IEEE Access*, 12: 3400588. https://doi.org/10.1109/ACCESS.2024.3400588

Kim, J., H. Kim, H.G. Kim, D. Lee and S. Yoon. 2025. A comprehensive survey of deep learning for time series forecasting: architectural diversity and open challenges. *Artif. Intell. Rev.*, 58: 216.

Liang, T., J. Glossner, L. Wang, S. Shi and X. Zhang. 2021. Pruning and quantization for deep neural network acceleration: A survey. *Neurocomputing*, 461: 370-403.

Liu, X., W. Liu, Q. Tang, B. Liu, Y. Wada and H. Yang. 2022. Global agricultural water scarcity assessment incorporating blue and green water availability under future climate change. *Earth's Future*, 10: e2021EF002567. https://doi.org/10.1029/2021EF002567

López-Urrea, R., F.M.D.S. Olalla, C. Fabeiro and A. Moratalla. 2006. *An evaluation of two hourly reference evapotranspiration equations for semiarid conditions. Agric. Water Manag.*, 86(3): 277-282.

Luo, K., C. S. Lim, M. H. I. B. Ishak, M. S. A. Mahmud and N. Ba. 2025. Benchmarking deep learning models for ET_0 forecasting on edge devices. *Nat. Hazards*, 56(12): 1269-1298. https://doi.org/10.2166/nh.2025.130

Lv, Z., S. Yang, S. Ma, Q. Wang, J. Sun, L. Du, J. Han, Y. Guo and H. Zhang. 2025. Efficient deployment of peanut leaf disease detection models on edge AI devices. *Agriculture*, 15(3): 332. https://doi.org/10.3390/agriculture15030332

Mohamed, H., E. Al-Masri, O. Kotevska and A. Souri. 2022. A multi-objective approach for optimizing edge-based resource allocation using TOPSIS. *Electronics*, 11(18): 2888.

Pandey, J. and A.R. Asati. 2023. Lightweight convolutional neural network architecture implementation using TensorFlow Lite. *Int. J. Inf. Technol.*, 15: 2489-2498.

Parsinejad, M., M.R. Yazdani and H. Ebrahimian. 2008. Field and regional scale evaluation of irrigation efficiency in paddy fields: A case study of Guilan, Iran. *Irrig. Drain.*, 57: 478-492.

Paula, E., J. Soni, H. Upadhyay and L. Lagos. 2025. Comparative analysis of model compression techniques for achieving carbon-efficient AI. *Sci. Rep.*, 15: 23461. https://doi.org/10.1038/s41598-025-07821-w

Sarkar, S.S., J. Bedi and S. Jain. 2025. A deep learning based framework for enhanced reference evapotranspiration estimation: evaluating accuracy and forecasting strategies. *Sci. Rep.*, 15: 15136. https://doi.org/10.1038/s41598-025-99713-2

Selva Jeba, G. and P. Chitra. 2024. River flood prediction through flow level modeling using multi-attention encoder–decoder-based TCN with filter-wrapper feature selection. *Earth Sci. Inform.*, 17: 5233-5249.

Shahab, H., M. Naeem, M. Iqbal, M. Aqeel and S.S. Ullah. 2025. IoT-driven smart agricultural technology for real-time soil and crop optimization. *Smart Agric. Technol.*, 10: 100847. https://doi.org/10.1016/j.atech.2025.100847

Shan, R., X. Jia, X. Su, Q. Xu, H. Ning and J. Zhang. 2025. AI-driven multi-objective optimization and decision-making for urban building energy retrofit: Advances, challenges, and systematic review. *Appl. Sci.*, 15(16): 8944. https://doi.org/10.3390/app15168944

Tanabe, R. and H. Ishibuchi. 2020. A review of evolutionary multimodal multiobjective optimization. *IEEE Trans. Evol. Comput.*, 24(1): 193-200.

Tang, H., X. Ling, L. Li, L. Xiong, Y. Yao and X. Huang. 2022. One-shot pruning of gated recurrent unit neural network by sensitivity for time-series prediction. *Neurocomputing*, 512: 15-24.

Tausif, M., S. Dilshad, Q. Umer, M.W. Iqbal, Z. Latif, C. Lee and R.N. Bashir. 2023. Ensemble learning-based estimation of reference evapotranspiration (ET). *Internet Things*, 24: 100973.

Trajković, S. 2010. Testing hourly reference evapotranspiration approaches using lysimeter measurements in a semiarid climate. *Hydrol. Res.*, 41(1): 38-49.

Ullah, A., K. Muhammad, W. Ding, V. Palade, I. Ul Haq and S.W. Baik. 2021. Efficient activity recognition using lightweight CNN and DS-GRU network for surveillance applications. *Appl. Soft Comput.*, 103: 107102. https://doi.org/10.1016/j.asoc.2021.107102

Valerio, L., F.M. Nardini, A. Passarella and R. Perego. 2021. Dynamic hard pruning of neural networks at the edge of the internet. *J. Netw. Comput. Appl.*, 190: 103330.

Villegas-Vega, R., A. Márquez-Grajales, E. Mezura-Montes, F. Salas-Martínez, M.A. Ojeda-Misses and C. Romo-Gómez. 2025. Optimization of LSTM networks through neuroevolution for drought forecasting in Mexico. *Theor. Appl. Climatol.*, 156: 562. https://doi.org/10.1007/s00704-025-05818-z

Vindas, Y., E. Roux, B.K. Guépié, M. Almar and P. Delachartre. 2025. An asymmetric heuristic for trained ternary quantization based on the statistics of the weights: An application to medical signal classification. *Pattern Recogn. Lett.*, 183: 64-72. https://doi.org/10.1016/j.patrec.2024.11.016

Wang, S. and Y. Kang. 2023. Gradient distribution-aware INT8 training for neural networks. *Neurocomputing*, 541: 126269. https://doi.org/10.1016/j.neucom.2023.126269

Wen, L., X. Zhang, H. Bai and Z. Xu. 2020. Structured pruning of recurrent neural networks through neuron selection. *Neural Netw.*, 123: 134-141.

Wilkinson, M.D., M. Dumontier, I.J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. Bonino da Silva Santos, P.E. Bourne, J. Bouwman, A.J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C.T. Evelo, R. Finkers and B. Mons. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Sci. Data*, 3: 160018.

Willmott, C.J. and K. Matsuura. 2005. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Clim. Res.*, 30: 79-82.

Yassen, M.A., E.S.M. El-Kenawy, M.G. Abdel-Fattah, I. Ismail and H.E.S. Mostafa. 2025. Explainable artificial intelligence for wind power forecasting model based on long short-term memory. *Neural Comput. Appl.*, 37: 14589-14611. https://doi.org/10.1007/s00521-025-11230-5

Yin, Y., Z. Gao, Y. Huang and Q. Liu. 2025. PLT-GRU: Physics-informed lightweight Transformer–GRU algorithm for few-shot battery state-of-health estimation. *Meas. Sci. Technol.*, 36(9): 096216.

Zhang, L., X. Zhao, G. Zhu, J. He, J. Chen, Z. Chen, S. Traore, J. Liu and V.P. Singh. 2023. Short-term daily reference evapotranspiration forecasting using temperature-based deep learning models in different climate zones in China. *Agric. Water Manag.*, 287: 108498.

Zhen, L., M. Li, D. Peng and X. Yao. 2020. Objective reduction for visualising many-objective solution sets. *Inf. Sci.*, 512: 278-294.

Zou, M., S. Kang, J. Niu, H. Huang, Y. Deng and H. Lu. 2024. Water-suitable reclamation can mitigate the amplified agricultural water stress driven by climate change in arid inland basin. *J. Hydrol.*, 640: 131736.