computers

MDPI

Article

Self-Referencing Digital Twin for Thermal and Task Management in Package Stacked ESP32-S3 Microcontrollers with Mixture-of-Experts and Neural Networks

Yi Liu [1], Parth Sandeepbhai Shah [2], Tian Xia [3,*] and Dryver Huston [1,*]

[1] Department of Mechanical Engineering, University of Vermont, Burlington, VT 05405, USA; yliu1@uvm.edu
[2] Intel Corporation, Rio Rancho, NM 87124, USA; parth.sandeepbhai.shah@intel.com
[3] Department of Electrical and Biomedical Engineering, University of Vermont, Burlington, VT 05405, USA
* Correspondence: txia@uvm.edu (T.X.); dryver.huston@uvm.edu (D.H.)

Abstract

Thermal limitations restrict the performance of low-cost, vertically stacked embedded systems. This paper presents a self-referencing digital twin framework for thermal and task management in a multi-device ESP32-S3 stack. The system combines a Mixture-of-Experts (MoE) model for task allocation with a neural network for short-term temperature prediction. Acting as a lightweight digital replica of the physical stack, the digital twin continuously monitors device states, forecasts thermal behavior 30 s into the future, and adapts workload distribution accordingly. The MoE model evaluates each device individually and asynchronously, estimating the portion of workload it should receive based on current state features including SoC temperature, CPU frequency, stack position, and recent task history. A separate neural network predicts future temperatures using real-time data from local and neighboring devices, enabling proactive thermal-aware scheduling. Training data for both models is collected through controlled experiments involving fixed-frequency operation and structured frequency switching with idle phases. All predictions and control actions are driven by in-built sensor feedback from the ESP32-S3 microcontrollers. The resulting digital twin supports distributed task scheduling based on temperature and works well in simple, low-cost edge systems with heat constraints. In one-hour experiments on a 6 ESP32-S3 stack, the proposed scheduling method completes up to 572 computation rounds at a 50 °C temperature limit, compared with 493 and 542 rounds under logistic regression based control and 534 rounds at fixed 240 MHz operation, while keeping peak temperature at 51 °C.

Keywords: digital twin; 3D package stack; thermal management; task allocation; temperature prediction; mixture-of-experts; neural network; ESP32

Academic Editor: Paolo Bellavista

Received: 13 November 2025
Revised: 9 December 2025
Accepted: 18 December 2025
Published: 21 December 2025

1. Introduction

In recent years, the concept of the digital twin has gained attention as a method for linking physical systems with their virtual counterparts to enable real-time monitoring, prediction, and control. A digital twin is a dynamic virtual representation of a physical asset that is continuously updated with live data, allowing simulations and decisions to reflect actual system conditions [1,2]. This approach has been widely explored in domains such as manufacturing, industrial design, and aerospace, where synchronized models support predictive maintenance, resource optimization, and operational resilience. As the cost of sensing and embedded computing continues to fall, lightweight digital twin systems are

becoming increasingly feasible in edge environments and resource-constrained platforms. In this context, the present work explores how a real-time control framework based on ESP32-S3 microcontrollers and AI models can function as a self-referencing digital twin for thermal and task management in compact, stacked ESP32-S3 microcontrollers. The AR environment in this study serves only as a visual display of system states and predicted temperatures and does not take part in sensing, prediction, or control.

Three-dimensional (3D) chip stacking has become an increasingly practical solution for improving the performance and compactness of modern electronic systems. By arranging devices vertically, this design reduces interconnect distance and saves space, making it suitable for embedded and edge computing scenarios [3,4]. However, the dense layout also brings new challenges in thermal management. Inner devices in the stack tend to accumulate heat more quickly due to limited airflow and restricted cooling surfaces, leading to uneven temperature distribution and potential performance degradation [5,6].

To maintain stable operation in such systems, dynamic thermal control and task scheduling are essential. Basic strategies, such as lowering processor frequency or shifting workloads when temperatures exceed a fixed threshold, have been widely adopted in embedded and mobile devices [7,8]. These methods are simple to implement and work under stable conditions, but they lack flexibility and may fail to respond effectively under rapidly changing thermal states or non-uniform workloads.

Recent advances in machine learning offer new possibilities for thermal-aware control in real time. Neural networks, in particular, have been shown to accurately predict temperature trends by analyzing historical data, even under nonlinear thermal behavior [9–11]. In addition, Mixture-of-Experts (MoE) models can be used to divide computational tasks among multiple units based on real-time system conditions, including thermal status and operating capacity [12]. When combined, these models can support more precise and adaptive scheduling in thermally constrained systems.

To evaluate the feasibility of this approach, a test platform based on six vertically stacked ESP32-S3 microcontrollers was developed. While not designed for high-performance computing, the ESP32-S3 serves as a low-cost and controllable prototype to simulate the behavior of a 3D chiplet stack. This setup allows for real-time temperature monitoring, adjustable operation frequencies, and distributed task execution, making it suitable for verifying thermal prediction and scheduling strategies in a constrained environment.

The experimental results demonstrate that the AI-based system improves both thermal stability and task throughput compared to static or rule-based methods. With accurate short-term temperature forecasting and flexible load distribution, the system achieves better performance under defined thermal limits.

Earlier thermal control methods operate by monitoring temperature changes and adjusting workload when a temperature limit is reached. The controller developed with logistic regression in previous work is used as one baseline because it predicts short-term temperature changes and modifies workload after limits are exceeded. The approach examined in this study uses direct temperature prediction together with proportional workload assignment, which enables task distribution to account for expected thermal conditions. For consistency, the evaluation compares this approach with fixed frequency operation and the logistic regression controller under the same thermal limits. The comparison considers task throughput, temperature regulation, and the computation effort required for model inference.

The remainder of this paper is organized as follows, and the overall architecture of the proposed self-referencing digital twin system is illustrated in Figure 1. The system consists of a digital twin layer that monitors device states and makes scheduling decisions, and a physical layer where ESP32-S3 devices execute assigned tasks and provide real-

time feedback. Section 2 reviews related work on thermal-aware scheduling and digital twin frameworks. Section 3 details the design and training of the Mixture-of-Experts (MoE) model for task allocation. Section 4 describes the hardware configuration and the experimental protocol for data collection. Section 5 introduces the neural network used for short-term temperature prediction. Section 6 presents experimental results, followed by discussion in Section 7 and conclusions in Section 8.

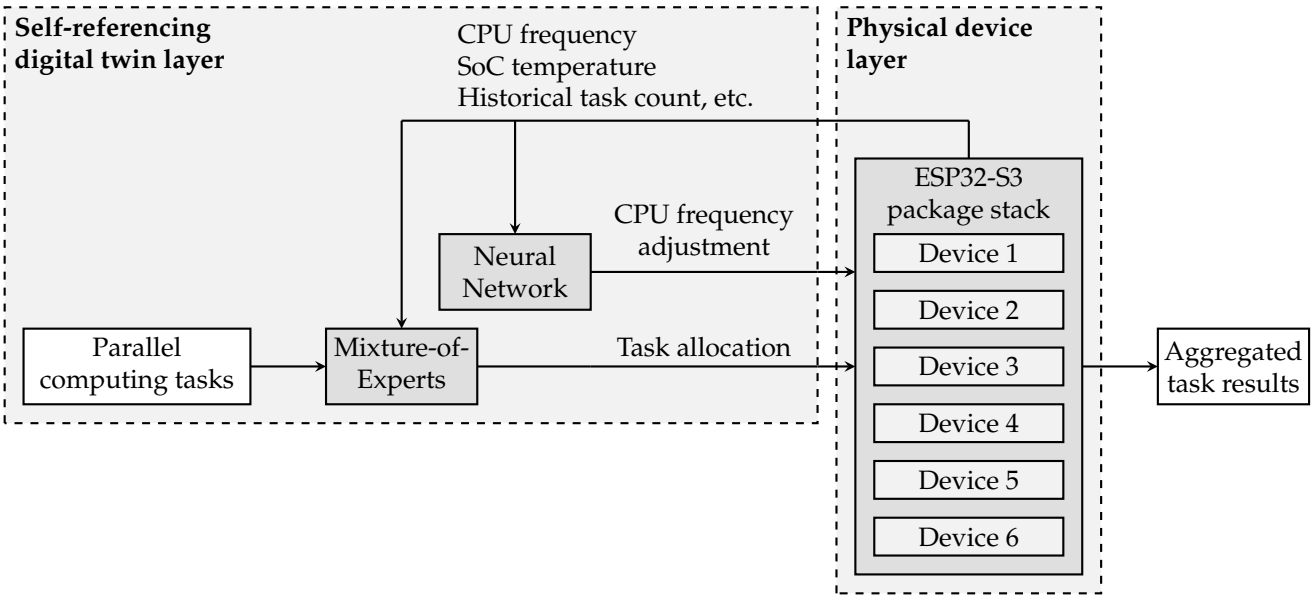


Figure 1. Overview of the proposed self-referencing digital twin framework for thermal-aware task scheduling. The digital twin layer uses real-time device states to predict future temperatures and dynamically adjust CPU frequency and task allocation across the ESP32-S3 stack to prevent overheating and maintain throughput.

2. Background and Related Work

Thermal management and task scheduling have long been studied in systems where power density and space constraints create thermal challenges. Traditional methods rely on static thresholds to trigger actions such as frequency throttling, core disabling, or workload migration. These rule-based approaches are computationally efficient and easy to implement but provide limited adaptability to real-time thermal variations and spatial temperature gradients in densely integrated systems [3,4].

To improve responsiveness, temperature-aware scheduling policies have been developed. These typically monitor on-chip or core temperatures and shift computation toward cooler units when imbalance is detected [8]. Techniques such as temperature-guided round-robin or thermal-aware bin packing have been proposed for multicore processors and system-on-chip (SoC) environments [6]. While these methods reduce thermal hotspots, they generally assume a fixed workload profile and do not account for future temperature trends.

More recently, predictive models based on machine learning have been introduced to estimate short-term temperature evolution. Regression-based models and decision trees have shown promise in embedded and mobile computing platforms [9,10]. Neural networks, including lightweight convolutional and recurrent structures, have also been used to anticipate thermal spikes and proactively manage workloads or frequency states [11,12]. In particular, these methods are effective in capturing the nonlinear relationships between frequency, power consumption, and thermal dynamics.

To explore thermal management in a compact and cost-effective hardware environment, a prior study [7] developed and evaluated a real-time control strategy using a stack

of six ESP32-S3 microcontrollers. The system was designed to mimic the geometric and thermal characteristics of a vertically integrated 3D chiplet stack, with each ESP32-S3 capable of independent operation and internal temperature monitoring.

Figure 2 illustrates the workflow architecture of the scheduling system. A large matrix is first constructed and divided into ten independent subtasks, each of which can be processed by any available ESP32-S3 device. The host continuously scans all connected serial ports to identify active devices. When one or more devices are available, subtasks are dynamically assigned based on the scheduling policy. Each device reports task completion back to the host, enabling the system to track overall progress. Once all subtasks are finished, the host aggregates the results and proceeds to the next iteration. The figure therefore represents the structural organization of task generation, device discovery, dynamic assignment, and result aggregation within the system.

In that prior system, each device was programmed to execute a computational task, specifically a matrix-based edge detection simulation, and periodically report its temperature and status to a host controller via serial communication. To reduce computational overload on overheated devices, a logistic regression model was trained to predict whether the temperature would increase in the next 10 s based on recent thermal history. The input features included the current temperature and the moving average over the past 10 s, sampled at 0.05 s intervals, as in Figure 3. A binary classifier was then used to determine if frequency scaling or task redistribution should be triggered.

The temperature prediction model was implemented using Scikit-Learn and trained on experimental datasets collected from ESP32-S3 devices running at different fixed frequencies (80 MHz, 160 MHz, and 240 MHz). For each frequency level, a separate logistic regression model was trained to fit the device's thermal behavior. The trained models achieved classification accuracies ranging from 96.9% to 97.7%, indicating that the model could correctly anticipate whether the temperature would rise within the near future.

When the predicted temperature rise exceeded a preset threshold, a rule-based controller adjusted the processor frequency downward or shifted part of the computation to a different ESP32 device with a lower thermal load. A dynamic load-sharing mechanism was implemented in the host computer to divide a large matrix into subtasks and allocate them to the available ESP32 units. The decision to assign tasks was based solely on each device's current temperature and frequency.

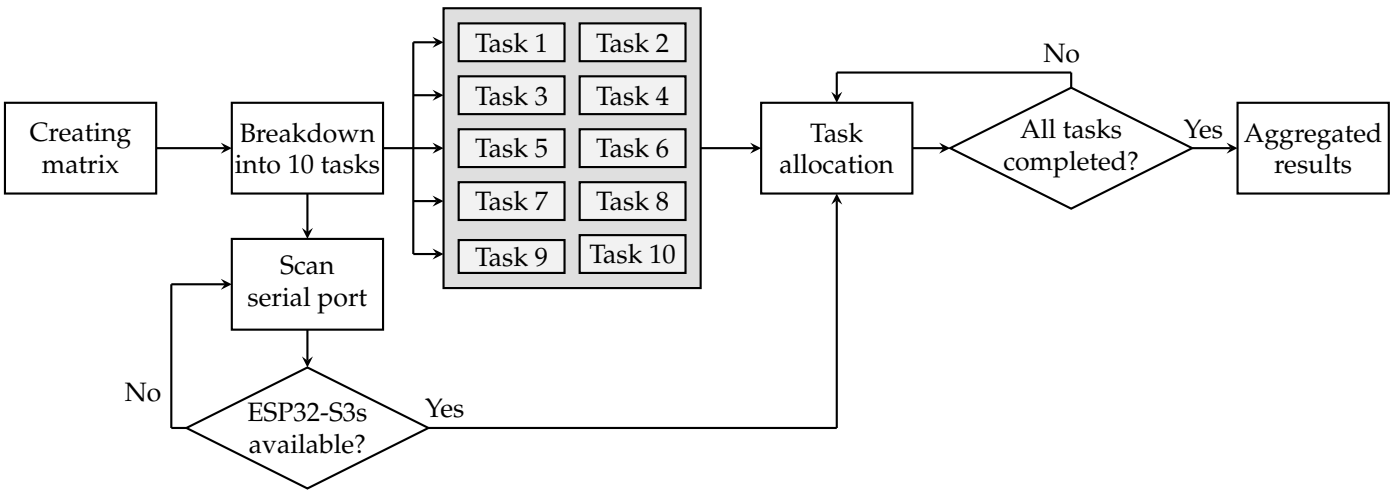


Figure 2. Task scheduling flow for the federated ESP32-S3 system. A matrix is first created and split into ten subtasks. The system scans for available ESP32-S3 devices via serial ports and dynamically allocates tasks accordingly. Once all tasks are completed, results are aggregated for output. Reproduced with permission from Liu et al. [2025] [7], published by MDPI, 2025.

Experiments were conducted using both fixed-frequency (baseline) and AI-assisted (logistic regression + control rule) setups. In the baseline configuration, devices running at 240 MHz completed the highest number of computation rounds (534 in one hour) but reached maximum temperatures of up to 56 °C. By contrast, the AI-driven approach capped the temperature at 50 °C or 45 °C while completing 542 and 493 rounds, respectively, indicating improved thermal safety with minimal performance loss.

However, this approach exhibited several limitations. First, the logistic regression model is inherently linear and lacks the capacity to model complex, nonlinear thermal interactions within the stack. This restricted its predictive accuracy, especially under rapidly changing workloads or uneven spatial heat distribution. Second, the task allocation logic was fixed and rule based, considering only current temperature and frequency without modeling inter-dependencies or optimizing for global system performance. The load-sharing model did not adapt to predicted future states or dynamically balance efficiency across the stack.

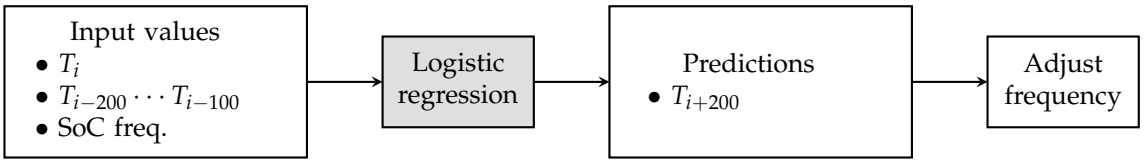


Figure 3. Thermal prediction and frequency control framework used in the previous study. A logistic regression model takes recent temperature history and SoC frequency as input, and predicts the future temperature at time step T_{i+200}. Based on the prediction, the system adjusts the processor frequency to avoid thermal violations. Caption reproduced with permission from Liu et al. [2025] [7]; published by MDPI, 2025.

Additionally, the previous work treated each ESP32 chiplet independently, without modeling spatial interactions such as thermal diffusion between vertically adjacent devices. This simplified the system design but may have limited the overall thermal control precision. Finally, the prediction window of 10 s, while effective for simple workloads, might not provide sufficient lead time for managing more complex computational tasks.

The present work builds upon this framework by addressing its key limitations. A neural network model is introduced to provide more accurate short-term temperature forecasting using multidimensional input features, including timestamp, frequency, current and past temperatures, chiplet position, and neighboring chip temperatures. Furthermore, the task allocation strategy is enhanced using a Mixture-of-Experts (MoE) model that adaptively selects target devices for computation based on both predicted thermal states and runtime performance metrics. These improvements enable a more flexible and scalable system suitable for complex or time-sensitive embedded applications operating under thermal constraints. Recent work has also applied neural-network-based methods to embedded optimization and system management tasks [13,14]. These works reflect a broader trend toward learning-driven resource management, which aligns with the direction of the present study.

3. Experimental Setup

3.1. Test Vehicle

The experimental platform is built using six ESP32-S3 microcontrollers arranged vertically inside a custom 3D printed shell. Each device includes a dual core processor, integrated temperature sensor, and supports selectable operating frequencies of 80 MHz, 160 MHz, or 240 MHz. The devices are spaced 1 mm apart and are connected to a host computer via USB. The vertical structure and tight enclosure lead to significant temperature gradients when the devices are active. The physical configuration is illustrated in Figure 4.

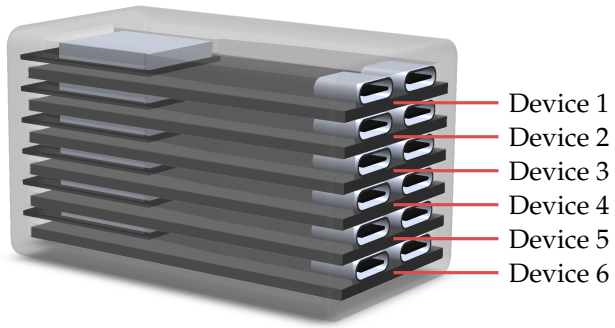


Figure 4. Vertical stack configuration of ESP32 S3 devices, showing layout and device numbering.

3.2. Data Collection

To collect training data for MoE model, the ESP32-S3 stack was operated for one hour under three fixed frequency settings (80 MHz, 160 MHz, and 240 MHz), shown in Figure 5. During this period, the SoC temperature of each device was recorded continuously. The temperature curves shown in Figure 6 illustrate the thermal behavior of the devices across different frequencies.

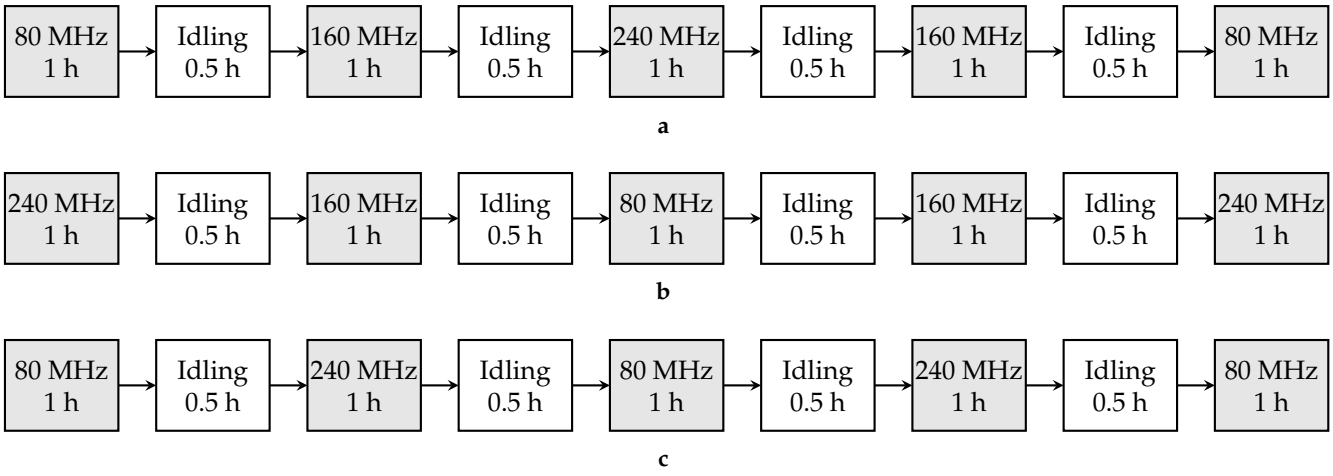


Figure 5. Experimental protocols for alternating frequency tests with idling periods. (**a**) Frequency sequence: 80 MHz → 160 MHz → 240 MHz → 160 MHz → 80 MHz, each active phase lasting 1 h, separated by 0.5 h idling. (**b**) Frequency sequence: 240 MHz → 160 MHz → 80 MHz → 160 MHz → 240 MHz. (**c**) Frequency sequence: 80 MHz → 240 MHz → 80 MHz → 240 MHz → 80 MHz. Each configuration explores the temperature response of the package stack under distinct transition paths between frequencies.

To train the neural network model for short-term temperature prediction, a dedicated dataset was collected by executing controlled, time

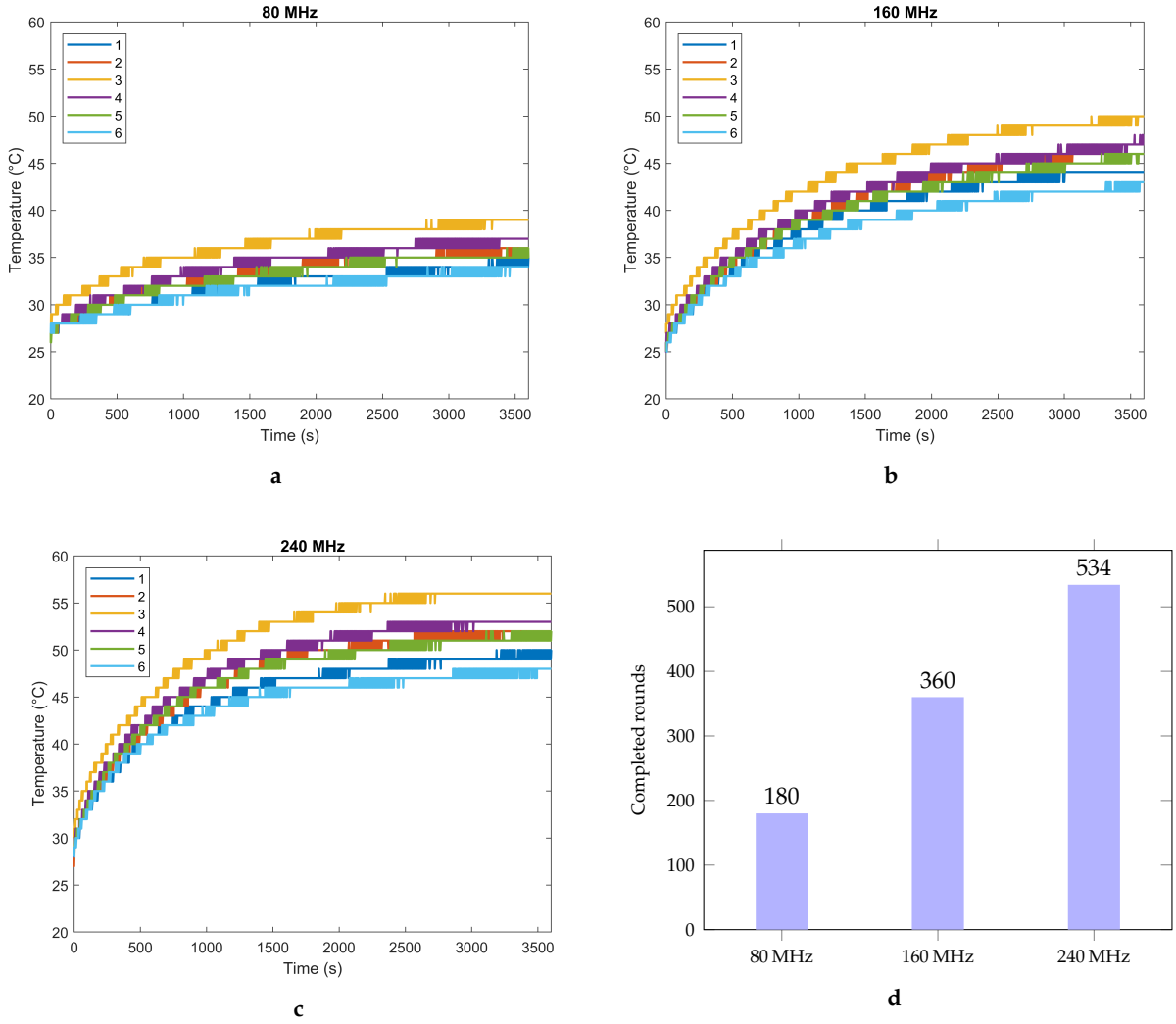


Figure 6. (**a**–**c**) Measured SoC temperature trajectories of six ESP32-S3 devices under constant workloads at (**a**) 80 MHz, (**b**) 160 MHz, and (**c**) 240 MHz. (**d**) Number of completed computation rounds under corresponding frequencies. Reproduced with permission from Liu et al. [2025] [7], published by MDPI, 2025.

4. Mixture-of-Experts Model Design

4.1. Input and Output Definition

The Mixture-of-Experts (MoE) model developed in this work is designed to support real-time task allocation under thermal constraints by evaluating one device at a time. Rather than comparing multiple devices simultaneously, the model operates asynchronously: it is invoked when a device completes its current subtask and reports its internal status. The model then estimates what proportion of the upcoming workload should be assigned to that device, based on its current condition.

To make this prediction, a fixed-length input vector is constructed to represent the device's operational state. This vector consist of the following features:

- SoC temperature: indicates the current processor thermal condition, which reflects immediate thermal stress and risk of overheating.
- Operating frequency: represents the current CPU clock speed and reflects the recent balance between power consumption and processing performance.
- Historical task count: defined as the number of subtasks the device has recently completed, serving as a cumulative load indicator.

- Positional index: encodes the device's vertical location within the stack, capturing spatial thermal effects due to heat accumulation and dissipation.

The output of the MoE model is a scalar value $y \in [0, 1]$, interpreted as the recommended proportion of the next task batch to assign to the querying device. For example, if the total batch consists of 10 subtasks and the model predicts $round(y, 1) = 0.3$, then three subtasks will be assigned to the device in the next scheduling cycle.

To construct supervision labels for training, task completion rates were measured under fixed frequency operation; the thermal behavior and statistics are shown in Figure 6. The ESP32-S3 stack was configured to run continuously at 80 MHz, 160 MHz, and 240 MHz, with each device executing a repeated edge detection workload for one hour. The number of completed task rounds per hour was recorded under each frequency setting, resulting in 180, 360, and 534 rounds, respectively. These values were used to estimate task-handling capacity under each configuration.

Rather than applying global normalization or soft allocation rules, task ratio labels were computed by dividing each measured by

$$\text{task ratio} = \frac{\text{measured throughput}}{534}. \tag{1}$$

This gives task ratio labels of 0.337, 0.674, and 1.0 for devices operating at 80 MHz, 160 MHz, and 240 MHz, respectively. These values were then uniformly assigned to all input samples corresponding to each frequency setting. Since the model performs predictions on one device at a time, labels do not need to be normalized across devices or task batches.

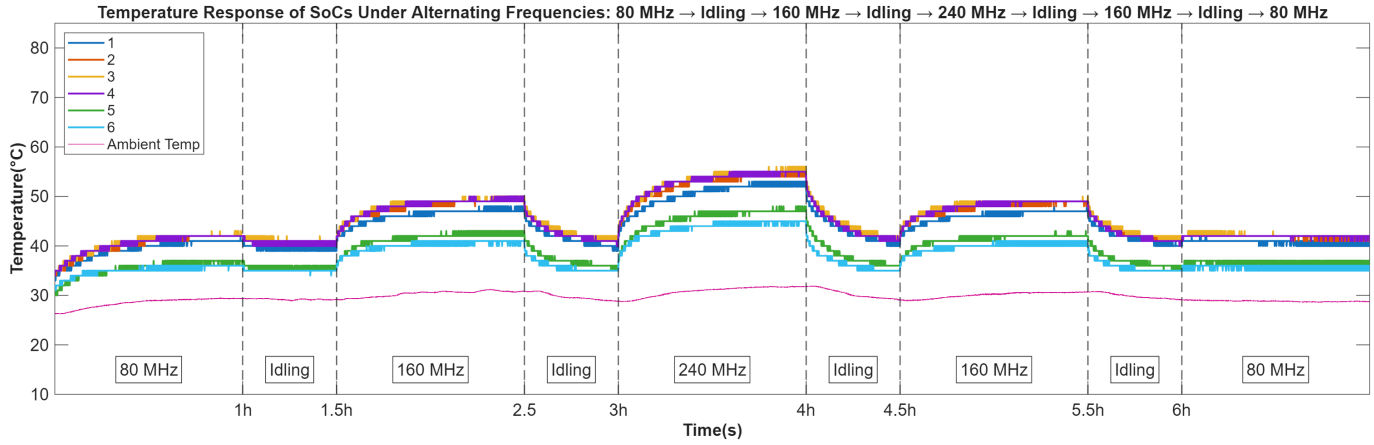
Devices at different vertical positions exhibit different heating rates, and a device operating at a higher temperature or with a rapidly increasing thermal slope must reduce its sustainable operating frequency. This increases the time required to complete each subtask and decreases the amount of workload the device can process before reaching its thermal limit. Conversely, cooler devices can sustain higher throughput. Therefore, the measured fixed-frequency throughput values (180, 360, and 534 rounds per hour) inherently capture the thermal limitations of each device and serve as appropriate supervision labels for the MoE model. The task ratio predicted by the MoE model thus reflects the device's workload suitability under its current thermal state.

Each output reflects an independent prediction of how much work the current device can handle, without requiring knowledge of the global system state. Additionally, using fixed-frequency data allows the model to observe a wide range of temperature and load combinations in a stable way. Had the dataset been collected while running under neural network thermal controller (thermal behavior shown in Figure 7), the resulting samples would likely be biased.

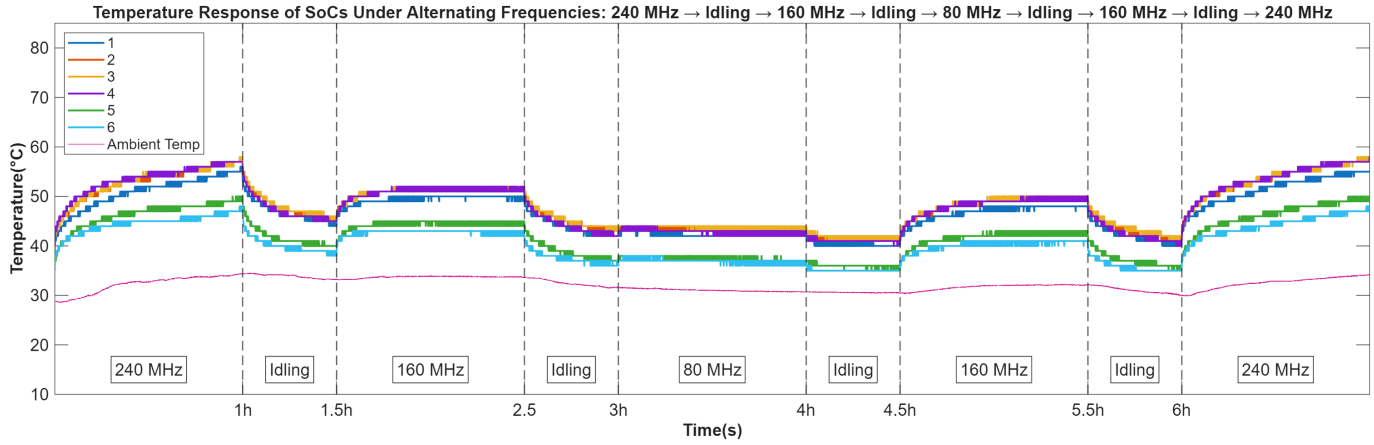
4.2. Model Architecture

The MoE model applied in this work is structured to evaluate a single device at a time and output a scalar representing the proportion of workload that the device should handle in the next task round. Unlike conventional MoE implementations that perform comparative inference across multiple candidates simultaneously, the proposed model operates asynchronously, responding only to the current status of the querying device.

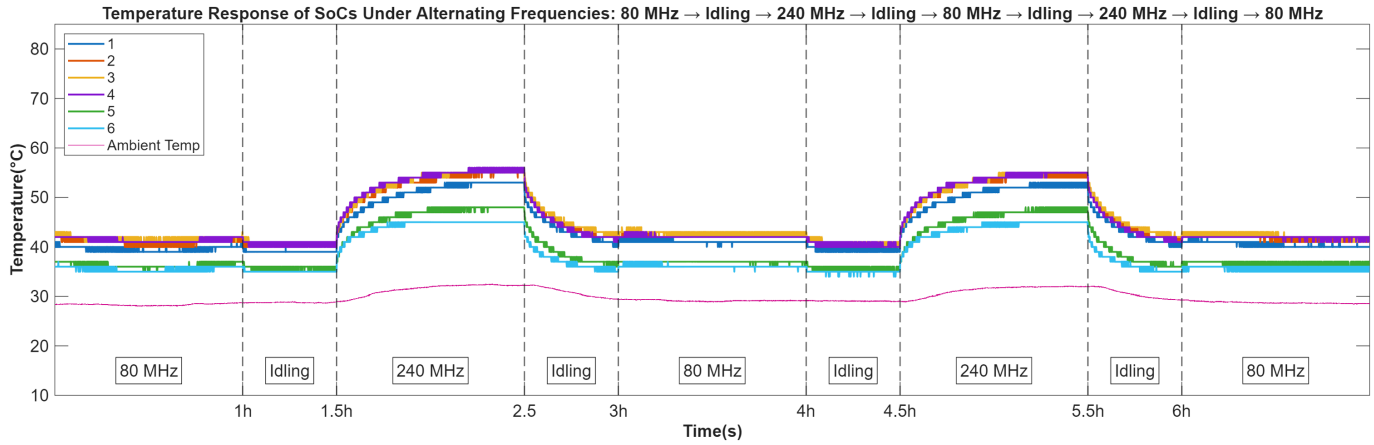
Figure 8 illustrates the structure of the model. The input state vector is processed in parallel by each expert and the gating network. Expert outputs are collected and combined via gating weights to produce the final task ratio output. Unlike classification MoE models, this design focuses on scalar estimation per device, allowing individualized and continuous load distribution over time.



a



b



c

Figure 7. Training data consisting of temperature response of SoCs under alternating frequency profiles. (**a**) Frequency sequence: 80 MHz → idling → 160 MHz → idling → 240 MHz → idling → 160 MHz → idling → 80 MHz. (**b**) Frequency sequence: 240 MHz → idling → 160 MHz → idling → 80 MHz → idling → 160 MHz → idling → 240 MHz. (**c**) Frequency sequence: 80 MHz → idling → 240 MHz → idling → 80 MHz → idling → 240 MHz → idling → 80 MHz. Each line corresponds to a different ESP32-S3 device in the stack, with ambient temperature shown as reference.

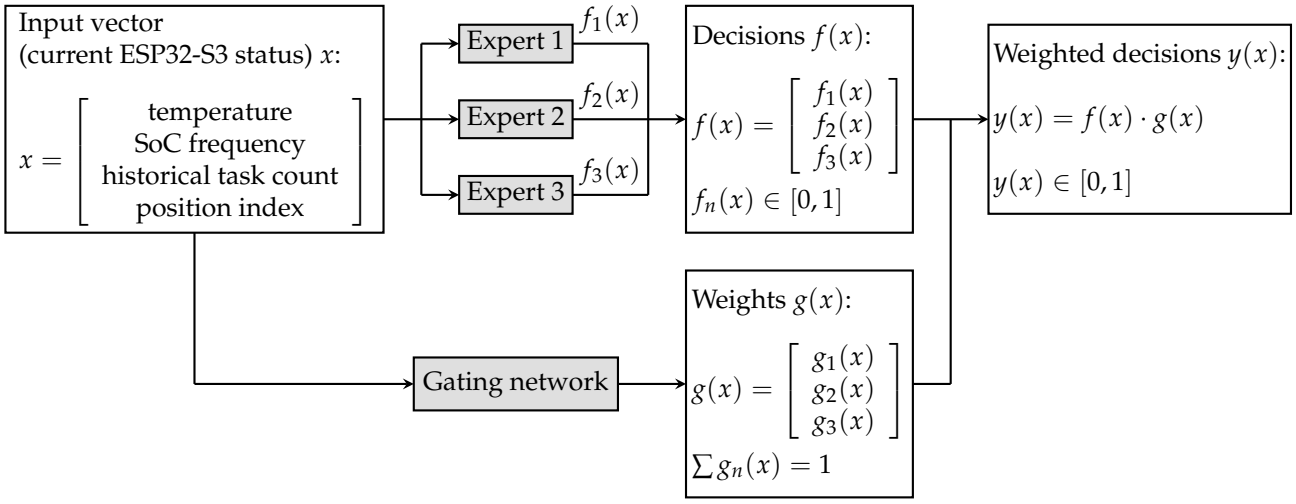


Figure 8. The MoE model evaluates one device at a time, producing a task ratio $y(x) = f(x) \cdot g(x)$ based on expert outputs and softmax weights.

As previously defined in Section 3.1, the input to the model consists of a compact state vector including thermal and operational characteristics of the device, and the output is a scalar task allocation ratio in the range of $[0, 1]$. Internally, the model is composed of two primary components, a set of expert networks and a gating network.

Each expert network independently processes the input and produces a scalar output representing its assessment of the device's suitability for additional workload. All expert networks share the same input dimensionality but are parameterized independently, following the classic structure proposed in the early MoE literature [15,16]. Prior work has shown that even without explicit role assignment, experts can develop functional specialization when trained with a gating network [17,18]; the gating mechanism in this model adaptively weighs the expert outputs to produce a final task ratio to the current device.

In parallel, the same input vector is passed to the gating network, which produces a weight vector via a softmax operation. These weights determine the contribution of each expert's output to the final decision. The final model output is calculated as the weighted sum of expert predictions:

$$y(x) = \sum_{i=1}^{n} g_i(x) \cdot f_i(x), \tag{2}$$

where $f_i(x) \in [0, 1]$ is the output of expert i, and $g_i(x)$ is the corresponding gating weight. The result $y(x) \in b\{0, 1\}$ is returned to the device that made the query and determines the fraction of the next task batch to be assigned to it. In implementation, this value is typically mapped to an integer number of subtasks based on the batch size.

To train the MoE model, real temperature and workload behavior were collected from the ESP32-S3 stack operating at three fixed frequencies: 80 MHz, 160 MHz, and 240 MHz. Each device continuously performed edge detection tasks for one hour, during which internal temperature, operating frequency, and task execution status were recorded. The resulting time–temperature profiles, shown in Figure 6, captured the thermal response patterns of each device under a sustained load.

Using this dataset, a simple MoE model was implemented in PyTorch (V 2.7.1) [19]. The input to the model consisted of the SoC temperature, CPU frequency, historical task count, and device position index. The output was trained to approximate the task ratio that each device should be assigned.

Since the MoE model outputs a continuous task-ratio value rather than a discrete class label, accuracy is not applicable to this regression-based allocator. Its behavior is

instead validated through the consistency of its output trends with device thermal states and through the resulting system-level throughput improvements.

5. Neural Network Model Design

Temperature prediction enables early thermal response in systems where workload intensity directly affects heat accumulation. A regression-based neural network is used to estimate the temperature of each ESP32 S3 device several seconds into the future, based on its current and neighboring thermal state. The forecasted value is later used in the task scheduling model to adjust workload before the temperature reaches a control threshold.

The model input includes ten features, listed in Table 1. These represent a snapshot of the system state at a given moment. The features include the device's internal temperature, operating frequency, execution status, stack position, ambient temperature, and SoC temperature of adjacent devices. This information is collected over serial communication at 20 samples per second and stored for offline training.

Table 1. Input features used for neural temperature forecasting.

Feature	Description
`Temp_SoC`	SoC temperature of the current ESP32 S3
`M_Rec`	Whether the device is ready to receive a task
`M_Proc`	Whether the device is processing a task
`M_End`	Whether the device has completed a task
`CPU_Freq`	Current CPU frequency (in megahertz)
`Stack_Pos`	Vertical index of the device within the stack
`Temp_a`	Ambient temperature near the device
`Hum`	Local humidity measurement
`Temp_0`	Temperature of the device below
`Temp_1`	Temperature of the device above

The model uses the collected data shown in Figure 7 and is trained to predict the SoC temperature 600 samples ahead, equivalent to approximately 30 s. Each training example includes the feature vector at time t and a ground truth temperature at time $t + 10$ s. The training dataset is generated by shifting and aligning the raw sensor data collected from real-time experiments involving frequency switching and idle transitions. The model is implemented using a feedforward neural network with two hidden layers, each containing 64 units. The model is trained using Scikit-Learn [11] multilayer perceptron regressor with ReLU activation and default optimizer settings. The training process uses 80 percent of the available data, while the remaining 20 percent is reserved for testing.

The training model achieves a mean absolute error of 0.18 degrees Celsius, a root mean square error of 0.29 degrees, and a coefficient of determination (R^2) of 0.996 on the test set. These results indicate that the model can track thermal trends with high accuracy using lightweight features and minimal historical data.

6. Results

6.1. Thermal Prediction Accuracy Comparison

To evaluate the temperature forecasting performance of the neural network model, raw temperature traces collected from real-time execution were compared against exponential curve fits. These experiments were conducted under both 45 °C and 50 °C target temperature constraints using two different control strategies: logistic regression and rule-based task allocator (from prior work [7]) and neural network prediction combined with the MoE-based task allocator.

Figure 9 presents the raw temperature trajectories of all six ESP32-S3 devices. Subfigures (a) and (c) show logistic regression control at 45 °C and 50 °C, respectively, while (b) and (d) illustrate neural network-based control under the same thermal targets. In all cases, the devices exhibit consistent heating behavior followed by saturation near the predefined thresholds. Compared to logistic regression, the neural network controller more tightly tracks the target temperature, particularly under the 45 °C constraint where overshoot is minimal.

To observe the behavior more easily, an exponential function was fitted to each device's temperature trajectory as shown in Figure 10. These fitted curves reveal the system's thermal convergence patterns and help isolate differences between control methods. At both temperature targets, the neural network-based controller exhibits faster convergence and smaller thermal gradients across devices. The temperature response is smoother, more uniform, and exhibits less cross-device variation compared to logistic regression.

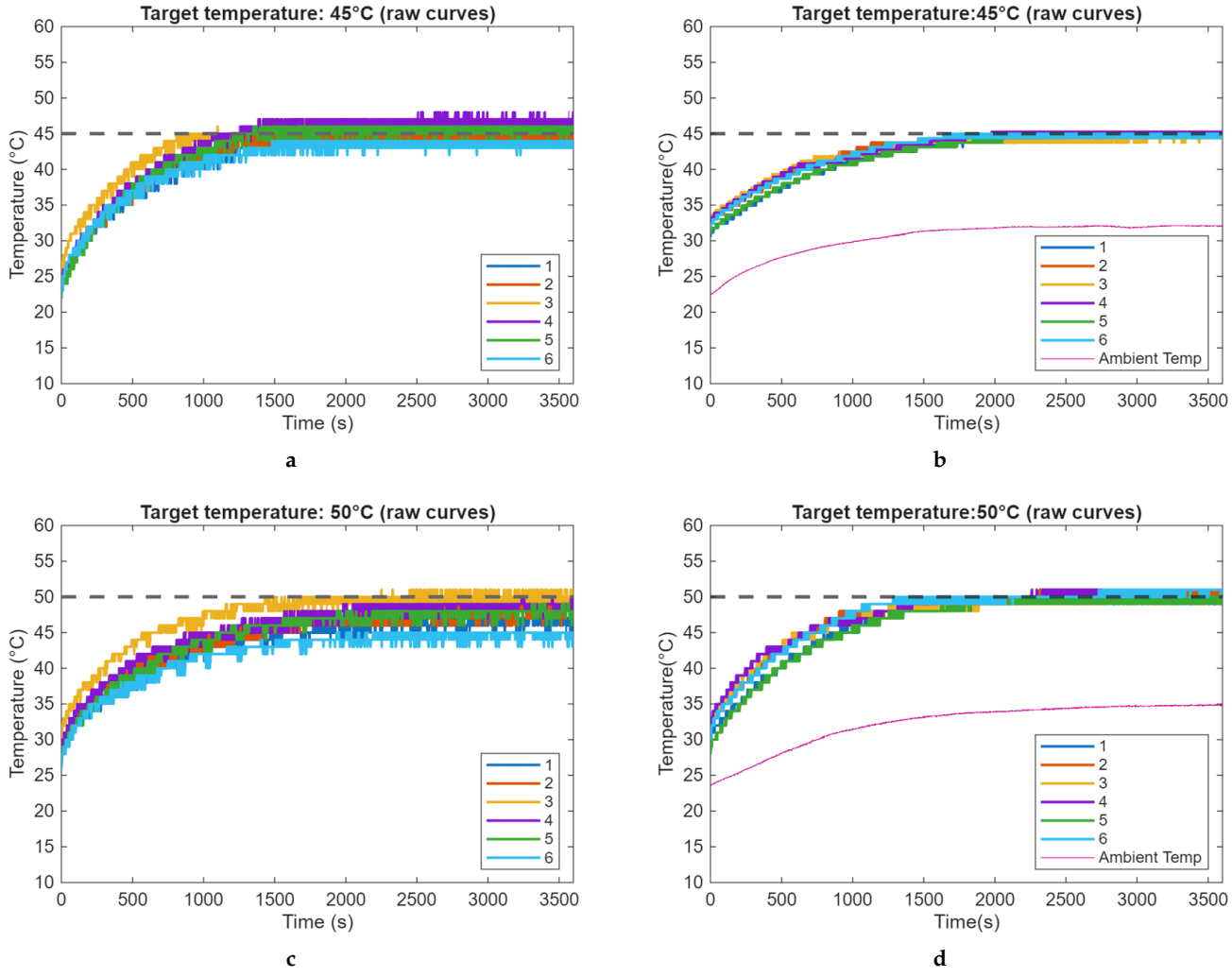


Figure 9. Raw temperature response curves of six ESP32-S3 devices under AI-based thermal management at different target temperatures and control strategies. (**a**) Logistic regression control with a 45 °C target. (**b**) Neural network control with a 45 °C target. (**c**) Logistic regression control with a 50 °C target. (**d**) Neural network control with a 50 °C target. The black dashed line denotes the temperature limit defined by the control algorithm. Subfigures (**a**,**c**) are adapted from our previous work [7], published by MDPI, 2025.

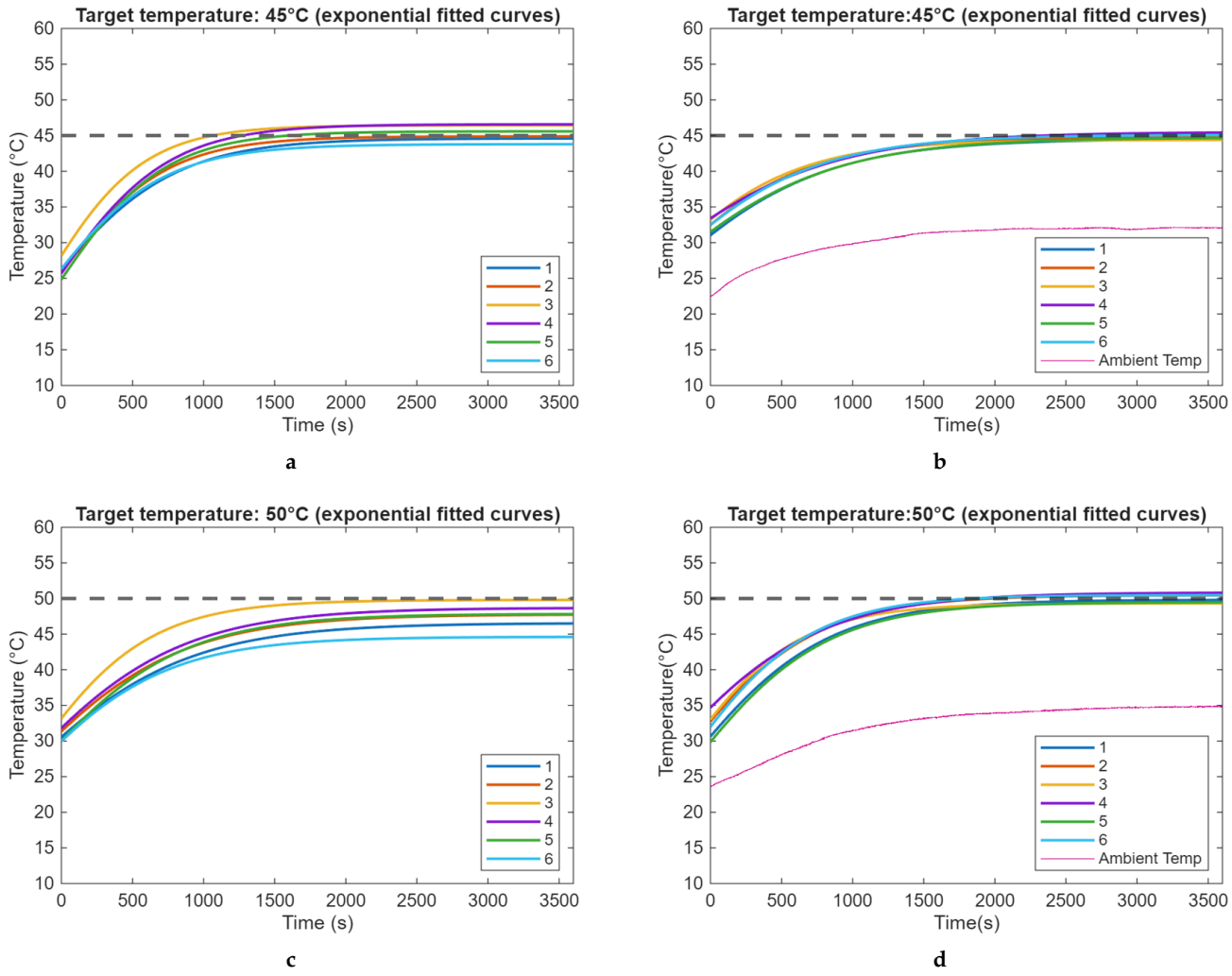


Figure 10. Exponential temperature response curves of six ESP32-S3 devices under AI-based thermal management at different target temperatures and control strategies. (**a**) Logistic regression control with a 45 °C target. (**b**) Neural network control with a 45 °C target. (**c**) Logistic regression control with a 50 °C target. (**d**) Neural network control with a 50 °C target. The black dashed line denotes the temperature limit defined by the control algorithm. Subfigures (**a**,**c**) are adapted from our previous work [7], published by MDPI, 2025.

These observations suggest that the neural network predictor, though trained with a limited future horizon (30 s), generalizes well to full-cycle thermal evolution and provides sufficient stability for feedback-based scheduling. The prediction-guided scheduling approach results in more consistent thermal behavior across devices, which is particularly important in tightly stacked architectures.

To assess the stability of the neural network predictor beyond point metrics, the distribution of prediction errors was evaluated across the full test dataset. The residuals exhibit a zero-centered, narrowly concentrated distribution with low variance, indicating that the model does not produce large or systematic errors under different operating conditions. More than 95% of the predictions fall within ± 0.5 °C of the ground truth, consistent with the reported MAE of 0.18 °C and RMSE of 0.29 °C. The high coefficient of determination ($R^2 = 0.996$) further confirms that the predictor captures the dominant thermal dynamics with minimal unexplained variance. Together, these statistics demonstrate that the model is both accurate and statistically stable for short-term thermal forecasting.

6.2. Task Completed Rounds Comparison

To evaluate overall task throughput under different control strategies, the number of completed task rounds was recorded for each configuration. Figure 11 summarizes the results across seven conditions: three fixed-frequency baselines (80 MHz, 160 MHz, and 240 MHz), logistic regression-based control at two thermal targets (45 °C and 50 °C), and the proposed NN and MoE-based control under the same two temperature targets.

At 80 MHz, 160 MHz, and 240 MHz, the system completed 180, 360, and 534 task rounds, respectively, with corresponding maximum temperatures of 39 °C, 49 °C, and 56 °C. While higher frequencies yield greater throughput, they also result in higher thermal stress. In particular, the 240 MHz configuration exceeds common thermal safety margins.

Under thermal control, the logistic regression baseline completed 493 rounds at 45 °C and 542 rounds at 50 °C, with observed maximum temperatures of 48 °C and 51 °C. The NN and MoE-based scheduling achieved slightly higher throughput: 514 rounds at 45 °C and 572 rounds at 50 °C. Both targets were respected, with peak temperatures held at 45 °C and 51 °C, respectively.

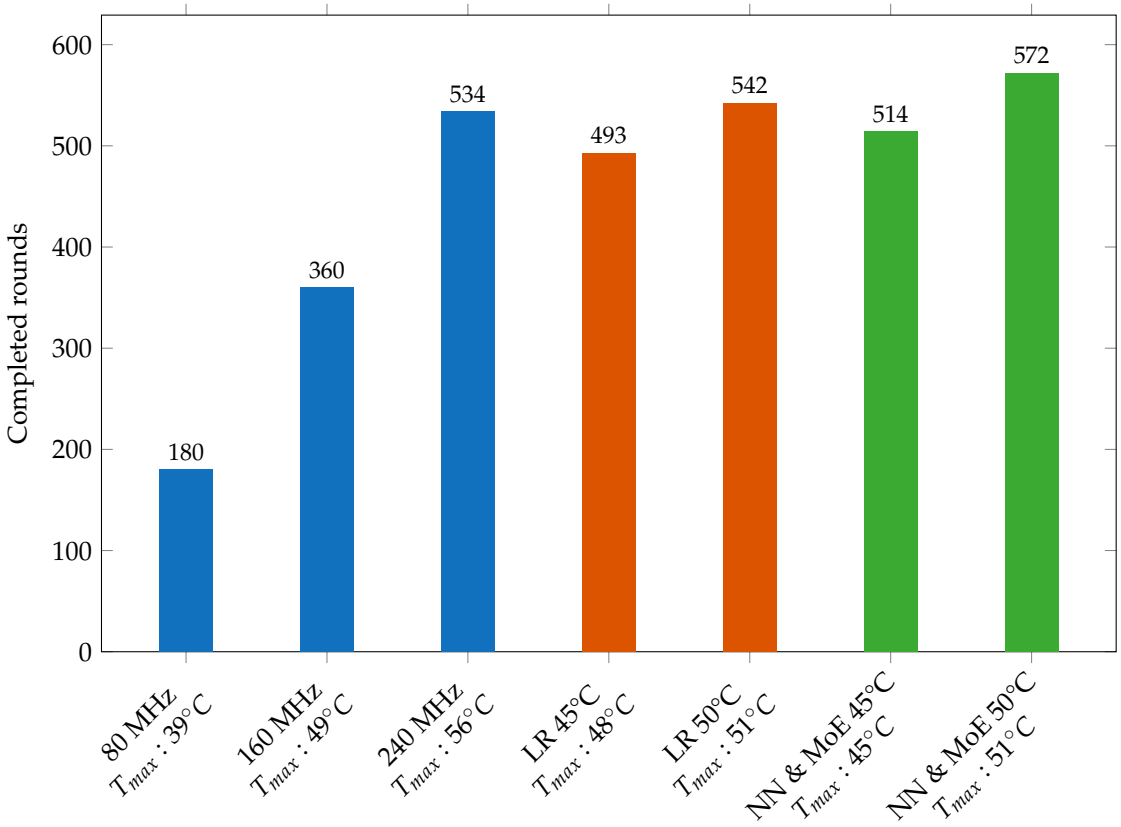


Figure 11. Total number of completed task rounds under different control strategies. The first three bars represent fixed-frequency baselines at 80 MHz, 160 MHz, and 240 MHz. The next two bars show results from logistic regression-based scheduling at thermal targets of 45 °C and 50 °C. The final two bars represent the performance of this newly proposed neural network and MoE-based scheduling technique under the same thermal constraints. Peak temperatures for each case are annotated below the x-axis labels.

These results can be interpreted from a benefits and costs perspective. The benefit corresponds to the amount of completed work, reflected in the total number of task rounds achieved under each thermal limit. The cost corresponds to the thermal load required to sustain this throughput, expressed through the peak temperature and the temperature variation across devices. When these measures are considered together, the method examined

in this study delivers higher task throughput under the same thermal constraints while maintaining similar thermal load compared with the fixed frequency and logistic regression baselines. The computation effort required for model inference remains small relative to the control period, indicating that the added modeling does not introduce a significant operational burden.

7. Augmented Reality Digital Twin Demonstration

To further illustrate the proposed self-referencing digital twin, an augmented reality (AR) demonstration was implemented on the Microsoft HoloLens 2 platform. The AR interface was developed in Unity 2022.3.62f1, with the HoloLens 2 connected wirelessly to the host computer. Real-time data from the ESP32-S3 stack, including SoC temperature and operating frequency, were transmitted to Unity and shared directly with the HoloLens 2 device.

Figure 12 shows the layered structure of the AR digital twin. The Physical Model refers to the actual ESP32-S3 hardware stack. The Digital Model is a virtual 3D wireframe model rendered in HoloLens 2. The Labels provide real-time overlays of temperature and CPU frequency.

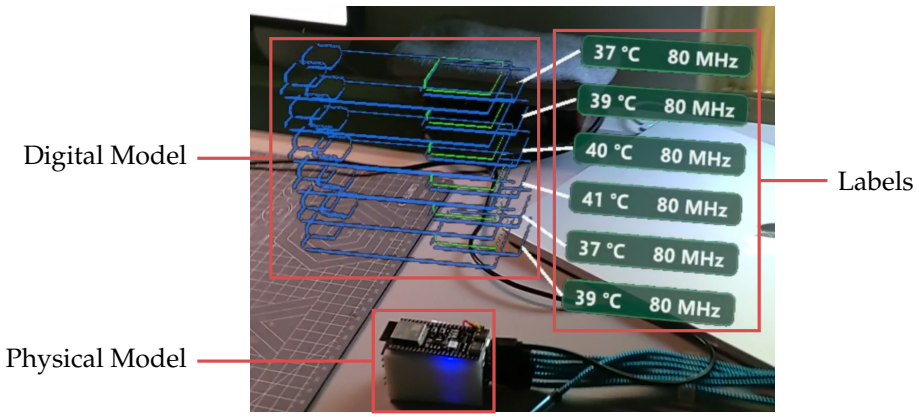


Figure 12. Structure of the AR-based digital twin. The Physical Model corresponds to the real ESP32-S3 hardware stack, the Digital Model represents its virtual replica in HoloLens, and the Labels provide real-time overlays of temperature and operating frequency. Together, these components create a synchronized multi-layered interface for monitoring and control.

To provide a clearer view of the digital model, Figure 13 presents a close-up rendering in Unity, highlighting the wireframe structure that corresponds to each ESP32-S3 layer.

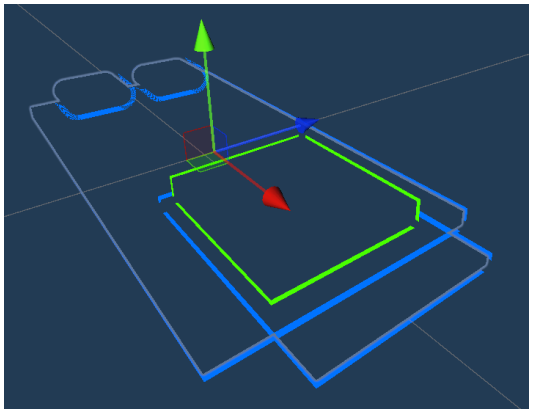


Figure 13. Close-up view of the digital model in Unity. The wireframe highlights the CPU block (green) and surrounding structure (blue). Arrows indicate the local coordinate axes, with red for X, green for Y, and blue for Z.

In normal operation, the AR environment displays each ESP32-S3 device as a block outline with attached labels showing real-time temperature and frequency. The digital twin demo also provides an interactive control panel accessible through the HoloLens 2, enabling manual adjustment of CPU frequency and toggling of AI-based scheduling. Standard MRTK 3.0 features allow resizing, repositioning, and gesture-based interaction with the digital twin.

When the AI-enabled toggle button is active, the digital twin visualization adapts the color of each CPU block according to the deviation from the predefined target temperature. Devices closer to the target appear red, while cooler devices shift toward green. This thermal-aware coloring highlights the proximity of each device to its safety threshold and allows users to quickly assess the system's overall thermal stability. Hardware-level feedback is provided by onboard WS2812 LEDs mounted on each ESP32-S3 device. The LEDs report the current operating frequency, with red corresponding to 240 MHz, yellow to 160 MHz, and blue to 80 MHz. This dual feedback—combining AR overlays with physical LED indicators—reinforces system transparency and enhances user awareness of both thermal and computational states. Figure 14 illustrates the AI-enabled AR interface.

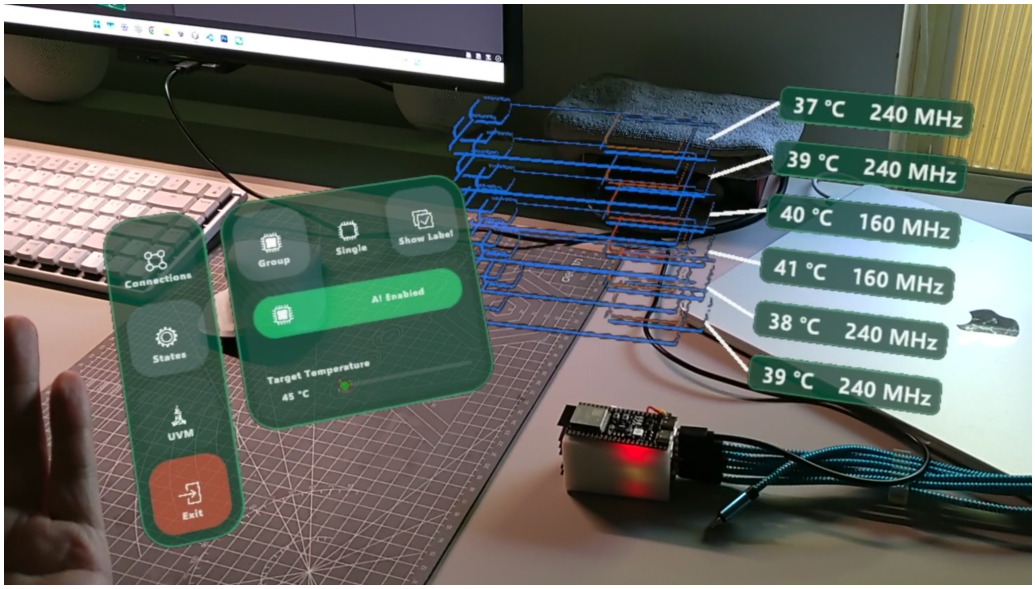


Figure 14. AR demonstration of the ESP32-S3 digital twin under AI-enabled scheduling. Block outline colors represent deviation from the target temperature (green = below target, red = near target). The control panel allows enabling or disabling AI mode and adjusting parameters. When AI is disabled, ESP32 devices fall back to a default 80 °C threshold, with onboard WS2812 LEDs indicating frequency states (red = 240 MHz, yellow = 160 MHz, blue = 80 MHz).

8. Discussion

The experimental results demonstrate that combining neural network-based temperature prediction with a Mixture-of-Experts (MoE) task scheduler provides an effective strategy for thermal management in stacked ESP32-S3 devices. Acting as a self-referencing digital twin, the system continuously observes device states, forecasts temperature trends, and adjusts task allocation before thermal violations occur. This closed-loop adjustment enables stable operation even in densely packed configurations with limited airflow. In one-hour tests under both 45 °C and 50 °C limits, all six devices maintained temperatures near their respective thresholds, with no overshoot or thermal instability after initial warm up.

In terms of throughput, the digital twin-based scheduling method achieves performance comparable to or better than fixed-frequency operation. Unlike fixed schemes that assign equal work regardless of thermal status, the proposed system anticipates heating

trends and reduces the workload of devices likely to overheat, while assigning more tasks to cooler or less active devices. This predictive load balancing maintains overall system performance without exceeding thermal constraints, and does so without requiring static rules or manual configuration. These results confirm that a lightweight digital twin can enhance both reliability and efficiency in low-cost, thermally constrained edge systems.

Unlike a digital shadow, which only mirrors system states without feedback, the proposed framework acts as a real-time self-referencing digital twin. The AR model is continuously updated with live data, but it also provides control interfaces that allow users or AI scheduling algorithms to directly influence the operation of the physical ESP32-S3 stack. This bidirectional interaction, monitoring and control, is the defining feature that distinguishes a digital twin from a static visualization model.

9. Conclusions

This work presents a self-referencing digital twin system for thermally aware task scheduling in a stacked ESP32-S3 platform. The system integrates a Mixture-of-Experts (MoE) model for asynchronous task allocation and a neural network for short-term temperature prediction. By continuously monitoring device-level features such as temperature, frequency, position, and task history, the digital twin dynamically adjusts workload distribution before thermal limits are violated.

Both models were trained using data collected from real hardware under fixed-frequency and frequency-switching conditions. The MoE model was supervised using measured throughput to estimate per-device task capacity without requiring global coordination. The neural network predicts each device's temperature 10 s ahead, enabling proactive adjustments. Experimental results show that the system maintains thermal stability and achieves throughput comparable to, or better than, fixed scheduling under 45 °C and 50 °C limits.

Future work will extend this digital twin framework to the Raspberry Pi Zero 2 stack, which introduces higher power density and thermal challenges due to its 1 GHz multi-core processor. This transition will allow further validation of the method's scalability to more powerful edge platforms, and support broader applications in real-time thermal and energy management for vertically integrated embedded systems.

Author Contributions: Conceptualization, Y.L.; Methodology, Y.L.; Software, Y.L.; Validation, Y.L.; Writing—original draft, Y.L.; Writing—review and editing, P.S.S.; Supervision, T.X. and D.H. All authors have read and agreed to the published version of the manuscript.

Funding: U.S. National Science Foundation: 2119485, 2345851 and Semiconductor Research Corporation 2023-PK-3181

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: Author Parth Sandeepbhai Shah is employed by the Intel Corporation. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Grieves, M.; Vickers, J. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In *Transdisciplinary Perspectives on Complex Systems*; Kahlen, F.J., Flumerfelt, S., Alves, A., Eds.; Springer: Cham, Switzerland, 2017; pp. 85–113.
2. Tao, F.; Zhang, M.; Liu, Y.; Nee, A.Y.C. Digital Twin Driven Smart Design. In *Digital Twin Driven Smart Manufacturing*; Tao, F., Ed.; Springer: Singapore, 2024; pp. 1–29.
3. Lau, J.H. Overview and Outlook of Through-Silicon Via (TSV) and 3D Integrations. *Microelectron. Int.* **2011**, *28*, 8–22. [CrossRef]

4. Singh, N.; Kumar, A.; Srivastava, K.; Yadav, N.; Singh, R.; Verma, A.S.; Gehlot, A.; Yadav, A.K.; Kumar, T.; Pandey, K.; et al. Challenges and Opportunities in Engineering of Next Generation 3D Microelectronic Devices: Improved Performance, Higher Integration Density. *Nanoscale Adv.* **2024**, *6*, 6044–6060. [CrossRef] [PubMed]
5. Wang, C.; Vafai, K. Heat Transfer Enhancement for 3D Chip Thermal Simulation and Prediction. *Appl. Therm. Eng.* **2024**, *236*, 121499. [CrossRef]
6. Chang, Y.-W. Physical Design Challenges in Modern Heterogeneous Integration. In Proceedings of the 2024 International Symposium on Physical Design, Taipei, Taiwan, 12–15 March 2024; pp. 125–134.
7. Liu, Y.; Shah, P.S.; Tian, X.; Dryver, H. An Approach to Thermal Management and Performance Throttling for Federated Computation on a Low-Cost 3D ESP32-S3 Package Stack. *Computers* **2025**, *14*, 147. [CrossRef]
8. Joseph, J.M. Networks-on-Chip for Heterogeneous 3D Systems-on-Chip. Ph.D. Thesis, Martin Luther University Halle-Wittenberg, Halle, Germany, 2019.
9. Chen, S.; Zhang, H.; Ling, Z.; Zhai, J.; Yu, B. The Survey of Chiplet-Based Integrated Architecture: An EDA Perspective. *arXiv* **2024**, arXiv:2411.04410.
10. Ramamoorthi, V. Multi-Objective Optimization Framework for Cloud Applications Using AI-Based Surrogate Models. *J. Big-Data Anal. Cloud Comput.* **2021**, *6*, 23–32.
11. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
12. Yakoubi, S. Sustainable Revolution: AI-Driven Enhancements for Composite Polymer Processing and Optimization in Intelligent Food Packaging. *Food Bioprocess Technol.* **2025**, *18*, 82–107. [CrossRef]
13. Li, D.-D.; Li, H.-L.; Hu, C.; Jiang, H.; Cao, J. Quasi-Projective Synchronization of Discrete-Time Fractional-Order Delayed Memristive Neural Networks with Uncertainties. *IEEE Trans. Cybern.* **2026**, *56*, 414–426. [CrossRef] [PubMed]
14. Roohi, M.; Mirzajani, S.; Haghighi, A.R.; Basse-O'Connor, A. Robust Design of Two-Level Non-Integer SMC Based on Deep Soft Actor-Critic for Synchronization of Chaotic Fractional-Order Memristive Neural Networks. *Fractal Fract.* **2024**, *8*, 548. [CrossRef]
15. Jacobs, R.A.; Jordan, M.I.; Nowlan, S.J.; Hinton, G.E. Adaptive mixtures of local experts. *Neural Comput.* **1991**, *3*, 79–87. [CrossRef] [PubMed]
16. Shazeer, N.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q.; Hinton, G.; Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv* **2017**, arXiv:1701.06538.
17. Eigen, D.; Ranzato, M. Learning factored representations in a deep mixture of experts. *arXiv* **2014**, arXiv:1312.4314. [CrossRef]
18. Ma, X.; Zhao, P.; Yi, X.; Chen, Z.; King, I. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD), London, UK, 19–23 August 2018; pp. 1930–1939.
19. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS), Vancouver, BC, Canada, 8–14 December 2019; pp. 8024–8035.