ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26aug1127

An Asymmetric Edge-Cloud Decoupled Architecture for Decomposition-Based PM2.5 Forecasting on Resource-Constrained Microcontrollers

Napatsorn Songsangka[1]

[1]Independent Researcher Bangkok, Thailand

Publication Date: 2026/09/03

Abstract: While hybrid CEEMDAN-LSTM models can improve PM2.5 forecasting accuracy, their ensemble-based decomposition and iterative sifting stages make direct execution of the complete pipeline impractical on resource-constrained microcontrollers such as ESP32-class edge nodes due to high SRAM requirements. To resolve this computational bottleneck and preserve decomposition benefits, we propose an asymmetric edge-cloud decoupled architecture that assigns CEEMDAN decomposition to the server while physically deploying a lightweight INT8-quantized LSTM on the ESP32-S3 microcontroller. In the proposed online workflow, the server decomposes historical PM2.5 observations and transmits the (168,14) decomposed feature tensor to the ESP32-S3 over Wi-Fi for multi-horizon inference. This data transfer requires a payload of only 2.30 KB per cycle. In the present experiments, server-side feature generation and network transmission were emulated offline, whereas the recurrent inference stage was implemented and executed on the ESP32-S3. Evaluated on a three-year Bangkok dataset, CEEMDAN-LSTM achieves an R^2 of 0.864 at the 1-hour horizon and 0.595 at 6 hours, outperforming the raw-input baselines at both horizons, with a larger margin at 6 hours. On-device ESP32-S3 evaluation yielded an R^2 of 0.824 and an average device-side inference latency of 995.48 ms. The serialized INT8 model occupies only 68.38 KB, representing a 54.1% memory reduction compared to the offline Float32 reference without significant accuracy degradation. These results demonstrate the physical feasibility of the recurrent inference stage and support the proposed asymmetric task allocation for multi-scale PM2.5 forecasting on resource-constrained microcontrollers.

Keywords: *PM2.5 Forecasting, on-Device Inference, Asymmetric Architecture, CEEMDAN, Long Short-Term Memory (LSTM), TinyML, ESP32-S3, Bangkok Air Quality.*

How to Cite: Napatsorn Songsangka (2026) An Asymmetric Edge-Cloud Decoupled Architecture for Decomposition-Based PM2.5 Forecasting on Resource-Constrained Microcontrollers. *International Journal of Innovative Science and Research Technology*, 11(8), 2554-2561. https://doi.org/10.38124/ijisrt/26aug1127

I. INTRODUCTION

Fine particulate matter (PM2.5) is among the most consequential environmental risk factors for human health, contributing to millions of premature deaths annually [1]. Metropolitan regions like Bangkok, Thailand, frequently record PM2.5 concentrations exceeding WHO guidelines due to intense vehicular traffic, agricultural biomass burning, and regional industrial activities [2]. Accurately forecasting these levels is therefore critical, yet PM2.5 time series are nonlinear and non-stationary and contain temporal structures at multiple scales, including short-term fluctuations, diurnal cycles, synoptic variations, and seasonal trends [3, 4].

Traditional statistical models may struggle to represent nonlinear and long-range temporal dependencies in PM2.5 series. Consequently, LSTM-based architectures have been widely adopted for air-quality forecasting because of their ability to model sequential dependencies [4–6]. However, models trained directly on raw PM2.5 series may have difficulty jointly representing high-frequency fluctuations and low-frequency trends.

To overcome this bottleneck, empirical decomposition methods are used to pre-process raw signals. Complete Ensemble Empirical Mode Decomposition with Adaptive Noise (CEEMDAN) improves on this by reducing mode-mixing while ensuring exact signal reconstruction [7, 8]. The resulting Intrinsic Mode Functions (IMFs) provide a data-adaptive multi-scale representation of the original signal. Combining CEEMDAN with LSTM has been shown to improve PM2.5 forecasting by decomposing the original series into components with different frequency characteristics before prediction [3, 9].

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26aug1127

Executing inference on edge devices can reduce cloud-side workload and enable predictions to be consumed directly by local applications [10, 11]. However, microcontroller units impose strict memory, computational, and power constraints [11–13]. Although TensorFlow Lite Micro supports neural-network inference on such devices [12], CEEMDAN remains difficult to execute locally because its ensemble-based sifting repeatedly performs extrema detection, spline interpolation, and intermediate-buffer operations [7, 8]. Existing CEEMDAN-based PM2.5 forecasting studies primarily evaluate predictive accuracy on workstation or server-class platforms [3, 4, 14, 15]. Edge-assisted air-quality forecasting has also been investigated, optimizing a deep-learning model for hourly PM2.5 prediction on resource-constrained edge devices and introducing a multi-task offloading strategy to distribute computational tasks between edge devices and servers [16]. That approach focuses on model optimization and task offloading rather than partitioning signal decomposition and recurrent inference across server and microcontroller resources. Consequently, the physical implementation and evaluation of multi-channel recurrent inference for CEEMDAN-based PM2.5 forecasting on ESP32-class devices remain insufficiently investigated.

To address this implementation gap, this paper introduces an Asymmetric Edge–Cloud Decoupled Architecture that assigns computationally intensive CEEMDAN decomposition to the server while deploying a compact recurrent forecasting model on the ESP32-S3. The server generates a multi-scale feature tensor from the available PM2.5 history, while the microcontroller executes a two-layer, multi-channel LSTM without local signal decomposition. The novelty lies in the hardware-aware task allocation and physical realization of recurrent inference rather than in a new CEEMDAN or LSTM algorithm. This task division is asymmetric: the computationally intensive feature extraction is centralized on the server, while the resource-constrained recurrent prediction is executed locally on the microcontroller. The main contributions of this study are as follows: (1) an asymmetric architecture combining causal server-side CEEMDAN with edge-side INT8 inference; (2) hardware deployment of a 33,185-parameter, two-layer LSTM processing a (168,14) tensor on ESP32-S3; (3) joint processing of 13 IMFs and one residual within a single model; (4) evaluation of multi-horizon accuracy, serialized model size, on-device accuracy, and device-side latency.

II. PROPOSED METHODOLOGY

➢ *Data Acquisition and Causal Window Preparation*

In this study, hourly PM2.5 concentrations were obtained from the US Embassy monitoring station in Bangkok, Thailand (January 2021 to December 2023). Prior high-resolution atmospheric and dispersion modelling has shown that road-traffic emissions and seasonal transboundary biomass burning are dominant contributors to ambient particulate pollution in the Bangkok Metropolitan Region [17]. These data were recorded via Beta Attenuation Monitors following US EPA protocols [18] and retrieved from the AirNow database. Records flagged with operational anomaly codes were excluded prior to preprocessing to prevent sensor malfunction artifacts from corrupting the spectral sifting process. Because CEEMDAN requires a continuous, gap-free input for its cubic-spline envelope interpolation, all missing values across 141 discontinuous sensor downtime segments were filled using linear interpolation, with 128 segments (90.8%) lasting 24 hours or fewer. The resulting dataset contains 22,165 valid hourly observations. The cleaned series was chronologically partitioned into training (80%), validation (10%), and test (10%) subsets. For each forecast origin, only observations available up to that time were used to generate the input, while the prediction target was the PM2.5 concentration at the corresponding 1-hour or 6-hour horizon. After decomposition, the latest 168 samples of the resulting components formed the LSTM input. The MinMaxScaler was fitted on the training samples and applied unchanged to validation and test data.

➢ *CEEMDAN Decomposition*

The Bangkok PM2.5 signal is shaped by overlapping temporal dynamics, from daily emission cycles to slowly varying seasonal patterns, motivating a data-adaptive decomposition before neural-network training. The foundation of this approach is Empirical Mode Decomposition (EMD) [19], which recursively extracts local oscillatory components via cubic-spline envelope interpolation. However, EMD is susceptible to mode mixing, where a single oscillatory mode may be split across multiple Intrinsic Mode Functions (IMFs) or distinct modes may blend within one IMF [7, 8].

CEEMDAN addresses this by injecting adaptive noise at successive decomposition stages rather than averaging noise-perturbed copies directly, yielding a cleaner separation of oscillatory modes alongside a residual trend component [7, 8].

Using the causal sample-generation procedure defined in Section II.A, CEEMDAN was applied using only PM2.5 observations available up to each forecast origin. The PyEMD implementation used 50 ensemble trials and a noise coefficient of 0.05, producing 13 IMFs and one residual. The latest 168 samples of these components formed the fixed (168,14) input tensor for predicting the raw concentration at the corresponding forecast horizon.

Compared with the raw signal, the decomposition provides a structured multi-channel input in which high-frequency IMFs represent rapid fluctuations, lower-frequency components describe slower variations, and the residual captures the long-term trend. This representation allows the forecasting model to learn from multiple apparent temporal scales within a single input tensor.

➢ *Asymmetric Edge-Cloud Deployment Architecture*

The central deployment constraint is the memory demand of ensemble decomposition on the target microcontroller. Even for a minimal 168-sample working window, retaining 50 trials of 14 floating-point components requires $50 \times 14 \times 168 \times 4 = 470{,}400$ bytes, while an additional $50 \times 168 \times 4 = 33{,}600$ bytes are required for the

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26aug1127

ensemble-noise matrix. These buffers total 504,000 bytes, approximately 492.2 KB, before accounting for residuals, spline envelopes, extrema arrays, and sifting buffers. Under the considered implementation, this approaches the nominal on-chip SRAM capacity of the ESP32-S3 and motivates assigning CEEMDAN to the server.

As shown in Fig. 1, CEEMDAN decomposition and feature preparation are assigned to the server. At each forecasting cycle, the server processes only the PM2.5 observations available at that time, extracts the latest 168 samples of 13 IMFs and one residual, applies training-fitted normalization and INT8 quantization, and generates a (168,14) tensor. This tensor contains 168 × 14 = 2,352 elements, corresponding to 2,352 bytes (2.30 KB) in INT8 format. This represents a 75% reduction from the 9,408-byte (9.19 KB) Float32 equivalent, excluding communication-protocol overhead. The INT8 tensor is then transmitted to the ESP32-S3 over Wi-Fi for recurrent inference.

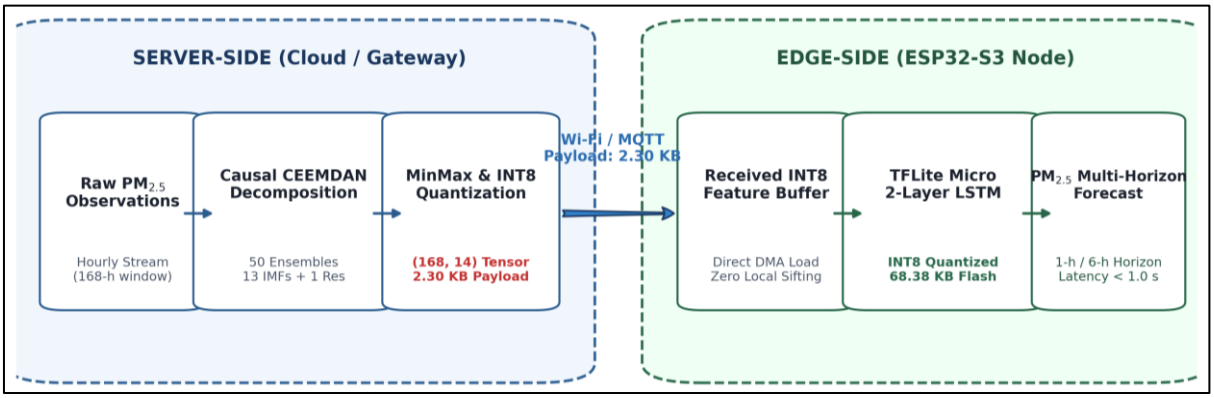

Fig 1 Intended Workflow of the Proposed Asymmetric Edge-Cloud Architecture. The Server Generates the Normalized INT8 (168,14) Feature Tensor Through Causal CEEMDAN Processing, while the Physical ESP32-S3 Executes the Quantized LSTM Inference Stage without Local Signal Decomposition.

Specifically, a clear distinction must be made between the offline training phase and the online inference phase. During offline training, the entire historical PM2.5 time series is partitioned into training, validation, and test sets, and the CEEMDAN decomposition is performed to prepare the training database. In the online prediction phase, no dataset partitioning is performed. Instead, the server maintains a sliding lookback window of the most recent 168 hourly PM2.5 observations, applies CEEMDAN decomposition to this active window to extract the latest multi-scale features, applies the training-fitted normalization and the same INT8 quantization parameters obtained during post-training quantization, and transmits only the resulting single (168, 14) INT8 feature tensor to the ESP32-S3. The edge device receives the pre-quantized tensor, loads it directly into the TensorFlow Lite Micro input buffer, and executes the recurrent forecasting model to predict the PM2.5 concentration at the target horizon. Signal decomposition, normalization, and input quantization are therefore performed on the server, reducing the computational workload of the ESP32-S3.

On the server side, CEEMDAN performs ensemble generation, extrema detection, spline interpolation, and iterative sifting, after which normalization and INT8 quantization produce the multi-scale input tensor. These operations remain on the server because they require repeated signal buffers and are not part of edge execution. On the edge side, the ESP32-S3 retains only the current input tensor, loads it directly into the TensorFlow Lite Micro input buffer, and executes the two-layer LSTM forward pass directly on the received tensor, with no decomposition or repeated preprocessing performed at the edge. The quantized scalar output is then converted back to PM2.5 concentration units using the stored quantization and scaling parameters. This asymmetric allocation confines memory-intensive feature generation to the server and resource-bounded recurrent inference to the microcontroller. It reduces edge memory and computation, but retains server dependence for generating multi-scale features.

The edge forecaster is a compact multi-channel LSTM with 33,185 trainable parameters. The (168,14) input passes through two cascaded LSTM layers [5], followed by a 16-unit ReLU dense layer and a linear output. The same architecture was used at both horizons, but a separate scalar-output model was trained for each horizon. The complete configuration is summarized in Table 1.

Table 1 Layer Configuration and Parameter Counts of the Proposed CEEMDAN-LSTM Model.

Module	No.	Layer	Activation	Output Shape	Parameters
Input	1	Input	—	(168, 14)	0
Feature Extraction	2	LSTM	Tanh	(168, 64)	20,224
Feature Extraction	3	LSTM	Tanh	(32,)	12,416
Regression	4	Dense	ReLU	(16,)	528
Regression	5	Dense	Linear	(1,)	17
Total Parameters					33,185

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26aug1127

Models were optimized offline using Adam [20] with MSE loss, mini-batches of 64, up to 100 epochs, and early stopping (patience = 15) on validation loss. Post-training integer quantization used 100 representative training windows to produce the TFLite model deployed with TensorFlow Lite Micro [12]. In the present experiments, the causal feature tensors were pre-computed and supplied to the physical ESP32-S3, allowing model conversion, firmware integration, on-device accuracy, and device-side latency to be evaluated without implementing live server-edge transmission. Together, the asymmetric task allocation and INT8 deployment enable multi-scale recurrent inference on the ESP32 while confining decomposition overhead to the server and limiting edge-side model storage.

➢ *Evaluation Metrics*

To quantify forecasting accuracy, we employed three complementary standard metrics: the coefficient of determination (R^2), Root Mean Square Error (RMSE), and Mean Absolute Error (MAE). While R^2 indicates the proportion of target variance captured by the model, both RMSE and MAE assess the absolute magnitude of prediction errors in native concentration units (μg/m³). These mathematical indicators are defined in Equations (1)–(3):

$$R^2 = 1 - [\sum_{i=1}^{n} (y_i - \hat{y}_i)^2] / [\sum_{i=1}^{n} (y_i - \bar{y})^2] \quad (1)$$

$$RMSE = \sqrt{[(1/n) \sum_{i=1}^{n} (y_i - \hat{y}_i)^2]} \quad (2)$$

$$MAE = (1/n) \sum_{i=1}^{n} |y_i - \hat{y}_i| \quad (3)$$

Where n represents the total sample size, y_i and $\hat{y}_i$ denote the actual and predicted concentrations for the i-th instance, and $\bar{y}$ is the average of all observed values.

III. EXPERIMENTAL RESULTS AND DISCUSSIONS

➢ *Multi-Horizon Forecasting Accuracy*

Before comparing forecasting accuracy, we first inspect the organization of the CEEMDAN components supplied to the LSTM. Fig. 2 presents an illustrative decomposition of the complete Bangkok series: the early IMFs exhibit rapid oscillations, whereas the later IMFs vary more slowly and the residual follows the long-term evolution. This visible progression represents the practical multi-scale advantage examined here, because short-lived fluctuations and background variation are separated rather than combined within a single raw trajectory. The visually distinct oscillatory scales are also qualitatively consistent with CEEMDAN's intended reduction of mode mixing. The figure is used only to illustrate the resulting representation. All forecasting inputs and metrics were generated causally using observations available at each forecast origin. Instantaneous-frequency distributions, mode-separation indices, and IMF orthogonality were not quantified, so Fig. 2 shows the general pattern rather than the actual extent of mode-mixing reduction or whether each IMF corresponds to a distinct physical process.

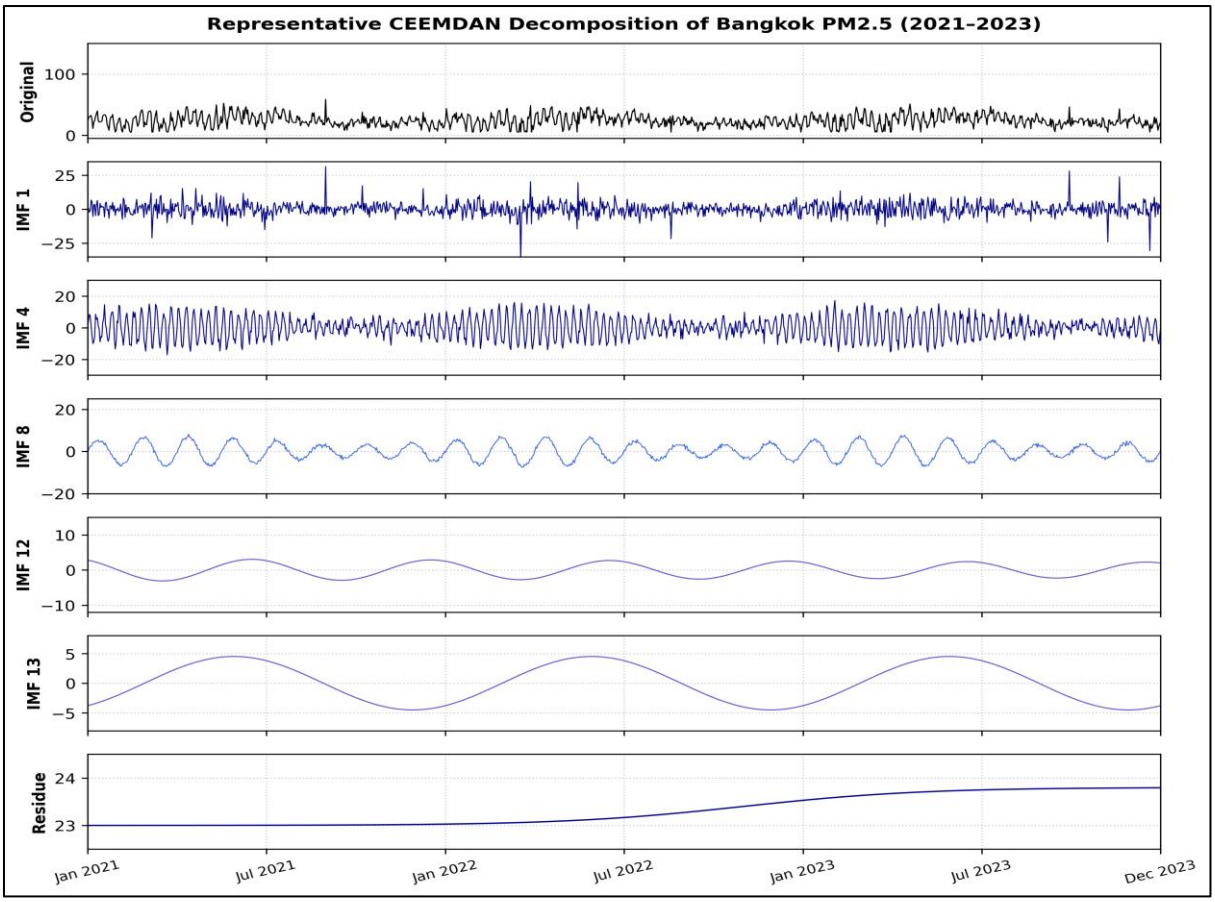

Fig 2 Illustrative CEEMDAN Decomposition of the Complete Bangkok PM2.5 Series into Representative IMFs and the Residual.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26aug1127

To evaluate the forecasting value of the decomposed representation, CEEMDAN-LSTM was compared with two raw-input baselines at the 1-hour and 6-hour horizons. The Random Forest used 50 trees and a maximum depth of 10, selected by grid search on the validation set. The standalone LSTM used the same recurrent and dense-layer configuration and training settings as the proposed model but received a single-channel raw 168-hour sequence, resulting in fewer parameters, as reported in Table 3. Accordingly, the comparison evaluates the complete decomposed multi-channel input strategy rather than an isolated CEEMDAN effect under identical parameter counts. Table 2 reports RMSE, MAE, and R^2 on the full held-out test set.

Table 2 Forecasting Performance Comparison on the Full Held-Out Bangkok Test Set.

Horizon	Model	RMSE (µg/m³)	MAE (µg/m³)	R^2
1 hour	Random Forest	5.77	4.05	0.678
1 hour	LSTM (Standalone)	6.02	4.19	0.649
1 hour	CEEMDAN-LSTM (Proposed)	3.75	2.63	0.864
6 hour	Random Forest	9.42	6.04	0.268
6 hour	LSTM (Standalone)	9.69	6.34	0.226
6 hour	CEEMDAN-LSTM (Proposed)	7.01	4.47	0.595

Table 2 shows that CEEMDAN-LSTM achieved lower RMSE and MAE and higher R^2 than both raw-input baselines at the evaluated horizons. At 1 hour, it obtained R^2 = 0.864 and RMSE = 3.75 µg/m³, compared with R^2 = 0.649 and RMSE = 6.02 µg/m³ for LSTM and R^2 = 0.678 and RMSE = 5.77 µg/m³ for Random Forest. At 6 hours, all models showed lower predictive performance, but CEEMDAN-LSTM retained R^2 = 0.595, compared with 0.226 for LSTM and 0.268 for Random Forest. These results indicate an accuracy advantage for the decomposed multi-channel input under the evaluated configuration, with a larger R^2 margin at the longer horizon.

Similar forecasting gains from decomposition-based PM2.5 inputs have been reported in previous studies [3, 9, 14, 15]. In the present study, a plausible interpretation is that separating rapid fluctuations from more slowly varying behavior provides the LSTM with a structured multi-channel representation. No IMF-group ablation or feature-attribution analysis was conducted, so the results do not show whether specific low-frequency IMFs are responsible for the longer-horizon improvement.

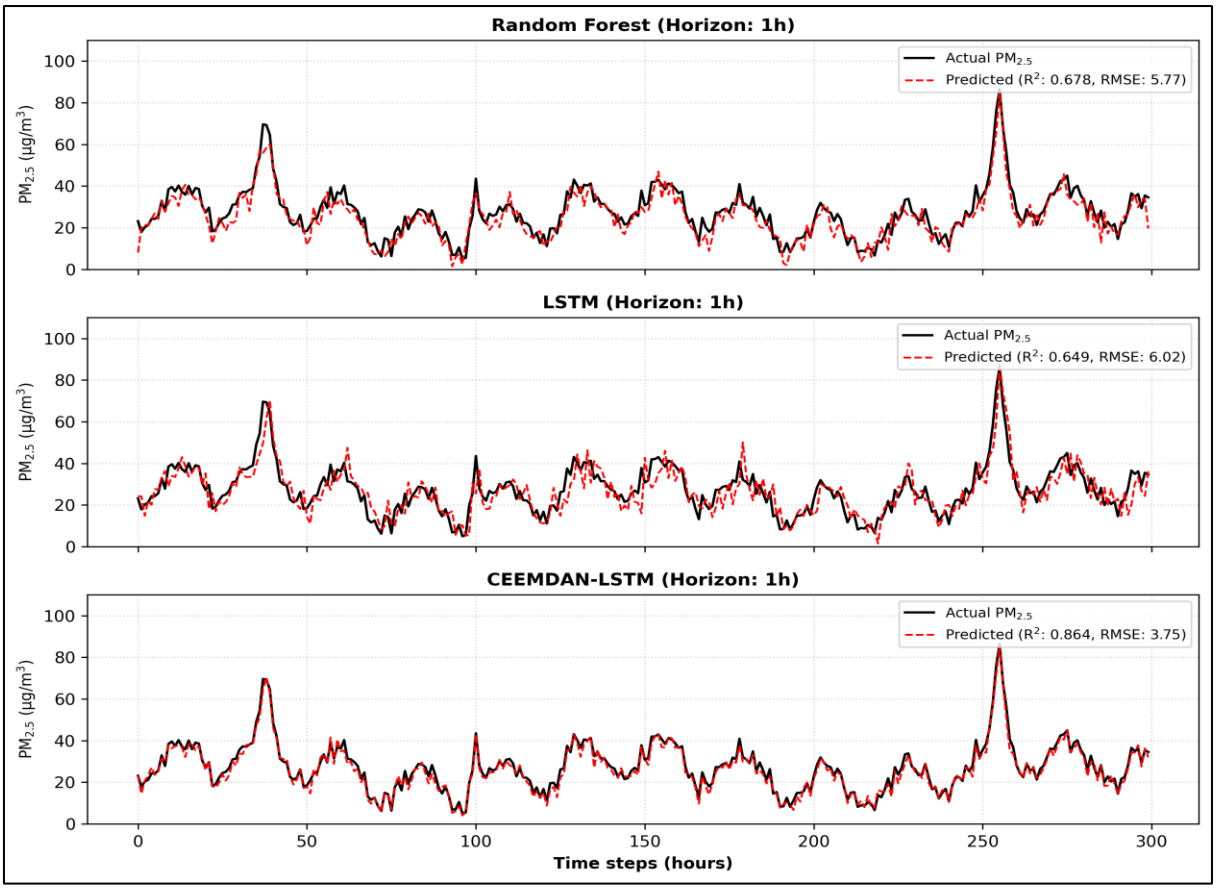

Fig 3 Predicted and Actual PM2.5 Concentrations Over a 300-Hour Segment of the Held-Out Test Set at the 1-Hour Horizon: (a) Random Forest, (b) Standalone LSTM, and (c) CEEMDAN-LSTM.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26aug1127

Fig. 3 provides a closer view of the 1-hour forecasts over a 300-hour segment of the test set. Within this interval, CEEMDAN-LSTM follows the observed trajectory and several rapid concentration increases more closely than the two raw-input baselines. This visual agreement is consistent with the full-test results in Table 2 and suggests better tracking of short-term variation, although a separate peak-event analysis would be needed to quantify this behavior.

➢ *Model Deployment and Profiling on ESP32-S3*

Having established forecasting performance, we next assess serialized model storage and physical execution on the ESP32-S3. Table 3 compares the proposed asymmetric strategy against alternative task allocations and characterizes the edge inference models. For an edge-only approach with on-device CEEMDAN, the estimated decomposition buffer for a 50-trial ensemble on a 168-sample window is approximately 492 KB, which exceeds the nominal ESP32-S3 SRAM and renders it infeasible. The proposed asymmetric allocation avoids this bottleneck, requiring only 2.30 KB of data transfer per cycle and 68.38 KB for the INT8 model, while maintaining partial offline capability if the last received feature tensor is cached on the device. Increasing the input from one to 14 channels raises the parameter count from 29,857 to 33,185, representing an 11.1% increase. Following TFLite INT8 conversion, the serialized multi-channel model size is reduced by 54.1% to 68.38 KB. Note that this value denotes serialized TFLite model size, not total firmware size or runtime SRAM use, as the available storage budget depends on the selected application partition [21]. As an analytical reference, 14 independent copies of the single-channel INT8 model would require approximately 911.68 KB, illustrating the storage advantage of joint multi-channel processing.

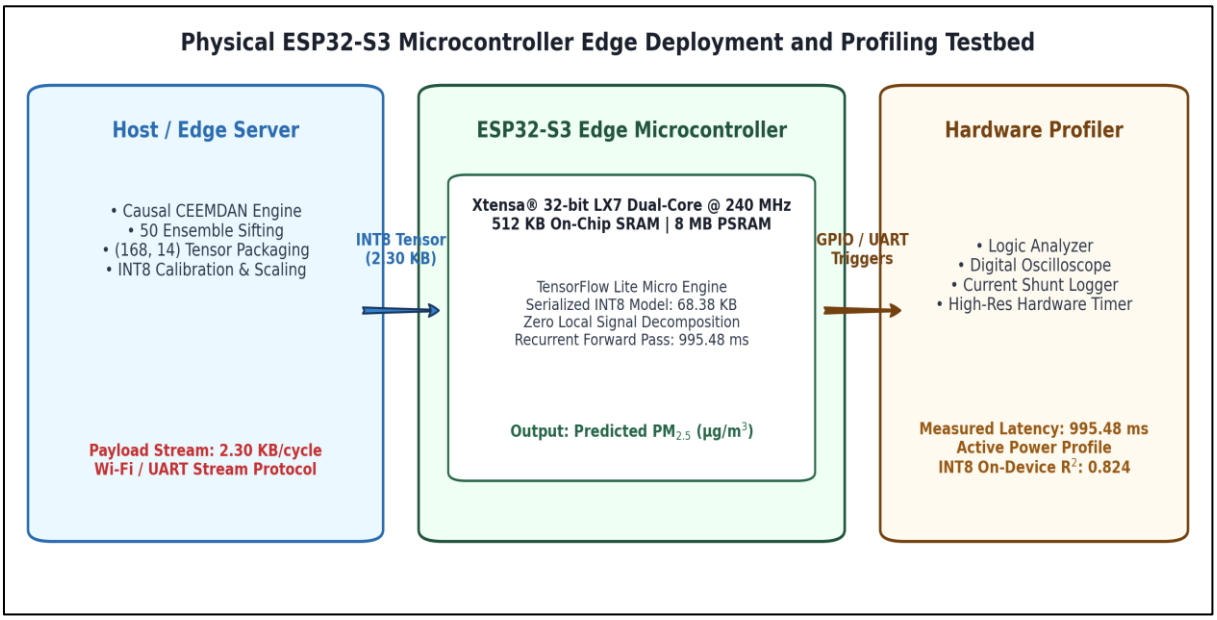

Fig 4 Physical ESP32-S3 Edge Deployment, Data Streaming, and Hardware Measurement Profiling Testbed.

Table 3 Deployment Strategy Comparison and Serialized Model Characteristics. Parameters, Float32 size, and INT8 Size Refer to the Edge Inference Component. Edge RAM Reflects the Decomposition Buffer Requirement. R^2 for Cloud-Only and Edge-Only (LSTM) are Offline References from Table 2. Only Proposed Method was Physically Measured on-Device (INT8, ESP32-S3).

Strategy	Parameters	Float32 Size	INT8 Size	Edge RAM	Data / Cycle	R^2	Offline Mode
Cloud-only (CEEMDAN-LSTM)	—	—	—	≈ 0	≈ 4 B	0.864	No
Edge-only (LSTM)	29,857	135.96 KB	65.12 KB	Low	0	0.649	Yes
Edge-only (CEEMDAN-LSTM)	—	—	—	≈ 492 KB	0	—	No
Proposed Asymmetric Method	33,185	148.96 KB	68.38 KB	Low	2.30 KB	0.824	Partial

The INT8 CEEMDAN-LSTM was then executed on the ESP32-S3 device using pre-computed (168,14) feature tensors. Evaluated on the same held-out test set, the on-device INT8 implementation achieved $R^2 = 0.824$, corresponding to an absolute decrease of 0.040 compared to the offline Float32 reference. Because the evaluation samples and network architecture were held fixed, this performance gap is entirely attributable to INT8 quantization and on-device numerical execution. The average device-side inference latency, measured from receipt of a feature tensor to production of the dequantized prediction, was 995.48 ms. This is roughly 0.03% of the 1-hour forecast interval, well within the timing budget of the intended hourly forecasting cycle.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26aug1127

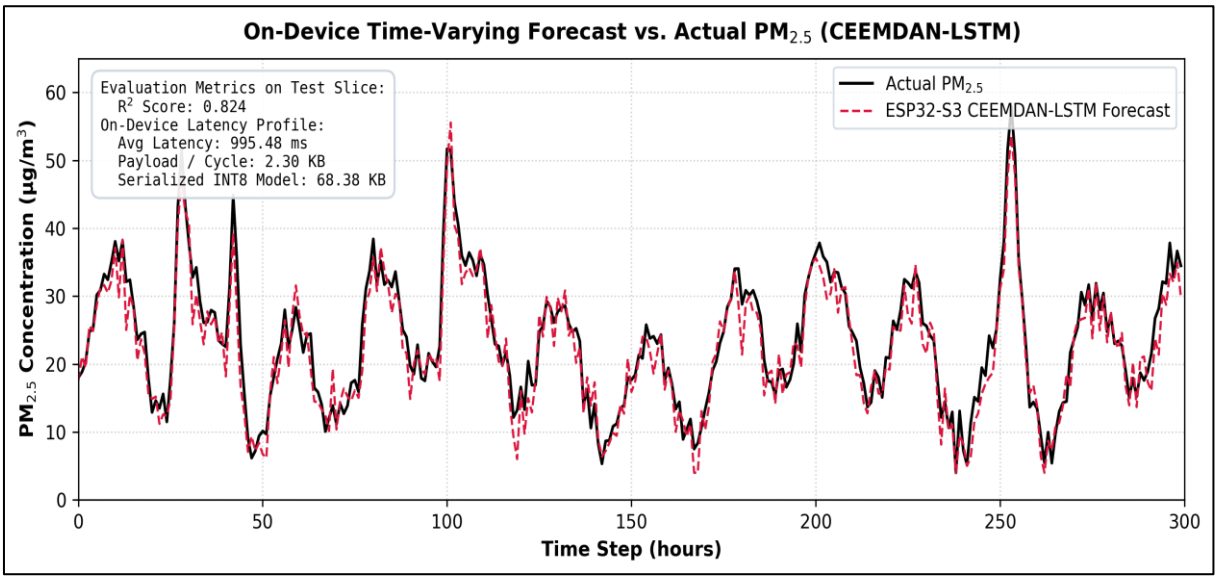

Fig 5 On-Device Time-Varying Forecast Versus Actual PM2.5 Concentrations on Physical ESP32-S3 Hardware for the 1-Hour Horizon Using INT8-Quantized CEEMDAN-LSTM. Full-Test Evaluation Yielded an on-Device R^2 of 0.824 and an Average Device-Side Latency of 995.48 ms.

IV. CONCLUSION

This study presents a hardware-aware asymmetric edge-cloud architecture that assigns causal CEEMDAN feature generation to a server and physically deploys a two-layer, multi-channel, INT8-quantized LSTM on the ESP32-S3. A single recurrent model jointly processes 13 IMFs and one residual, allowing the edge device to use multi-scale information without performing local signal decomposition. Offline evaluation showed that CEEMDAN-LSTM outperformed the raw-input baselines, achieving $R^2 = 0.864$ and $R^2 = 0.595$ at the 1-hour and 6-hour horizons, respectively. After model conversion and quantization, the serialized model size was reduced to 68.38 KB, representing a 54.1% reduction. On the same held-out test set, the deployed INT8 implementation achieved an on-device $R^2 = 0.824$, an absolute decrease of 0.040, with an average device-side latency of 995.48 ms and a data transfer payload of only 2.30 KB per forecast cycle. These findings support the proposed hardware-aware task allocation and the physical feasibility of recurrent inference on the target microcontroller.

In the present experiments, causal feature tensors were pre-computed and supplied to the ESP32-S3, so only the recurrent inference stage was evaluated on physical hardware, and the reported latency excludes server-side decomposition and network transmission. Several aspects were also not quantified in this study, including mode-separation quality of the CEEMDAN output, the contribution of individual IMF groups to the longer-horizon accuracy gain, and peak-event tracking performance. Future work will integrate the full communication pipeline, quantify mode-separation and IMF-group contributions, examine CEEMDAN consistency under continuous causal operation, characterize system-level performance including latency and energy consumption, and extend validation to additional monitoring sites and geographic regions to assess framework transferability.

ACKNOWLEDGMENT

This research is supported by the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand, under the Graduate Research and Research Assistantship Support Program.

REFERENCES

[1]. Cohen, A.J., Brauer, M., Burnett, R., Anderson, H.R., Frostad, J., Estep, K., Balakrishnan, K., Brunekreef, B., Dandona, L., Dandona, R., Feigin, V., Freedman, G., Hubbell, B., Jobling, A., Kan, H., Knibbs, L., Liu, Y., Martin, R., Morawska, L., Pope, C.A., Shin, H., Straif, K., Shaddick, G., Thomas, M., van Dingenen, R., van Donkelaar, A., Vos, T., Murray, C.J.L., Forouzanfar, M.H.: Estimates and 25-year trends of the global burden of disease attributable to ambient air pollution: an analysis of data from the Global Burden of Diseases Study 2015. The Lancet. 389, 1907–1918 (2017).

[2]. Phairuang, W., Suwattiga, P., Hongtieab, S., Inerb, M., Furuuchi, M., Hata, M.: Characteristics and source apportionment of ambient PM2.5 in Bangkok, Thailand. Atmospheric Pollution Research. 12, 101185 (2021).

[3]. Ban, W., Shen, L.: PM2.5 Prediction Based on the CEEMDAN Algorithm and a Machine Learning Hybrid Model. Sustainability (Switzerland). 14, (2022).

[4]. Zaini, N., Ean, L.W., Ahmed, A.N., Malek, M.A.: A systematic literature review of deep learning neural network for time series air quality forecasting.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26aug1127

Environmental Science and Pollution Research. 29, (2022).
[5]. Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory. Neural Comput. 9, 1735–1780 (1997).
[6]. Qin, D., Yu, J., Zou, G., Yong, R., Zhao, Q., Zhang, B.: A Novel Combined Prediction Scheme Based on CNN and LSTM for Urban PM2.5 Concentration. IEEE Access. 7, 20050–20059 (2019).
[7]. Torres, M.E., Colominas, M.A., Schlotthauer, G., Flandrin, P.: A complete ensemble empirical mode decomposition with adaptive noise. In: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 4144–4147 (2011).
[8]. Colominas, M.A., Schlotthauer, G., Torres, M.E.: Improved complete ensemble EMD: A suitable tool for biomedical signal processing. Biomed. Signal Process. Control. 14, 19–29 (2014).
[9]. Ameri, R., Hsu, C.C., Band, S.S., Zamani, M., Shu, C.M., Khorsandroo, S.: Forecasting PM 2.5 concentration based on integrating of CEEMDAN decomposition method with SVM and LSTM. Ecotoxicol. Environ. Saf. 266, 115572 (2023).
[10]. Alajlan, N.N., Ibrahim, D.M.: TinyML: Enabling of Inference Deep Learning Models on Ultra-Low-Power IoT Edge Devices for AI Applications. Micromachines (Basel). 13, (2022).
[11]. Abadade, Y., Temouden, A., Bamoumen, H., Benamar, N., Chtouki, Y., Senhaji Hafid, A.: A Comprehensive Survey on TinyML. IEEE Access. 11, 96892–96922 (2023).
[12]. David, R., Duke, J., Jain, A., Reddi, V.J., Jeffries, N., Li, J., Kreeger, N., Nappier, I., Natraj, M., Regev, S., Rhodes, R., Wang, T., Warden, P.: TensorFlow Lite Micro: Embedded machine learning on TinyML systems. In: Proceedings of the 4th MLSys Conference. , San Jose, CA (2021).
[13]. Warden, P., Situnayake, D.: TinyML Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers. O'Reilly Media, CA (2019).
[14]. Cai, P., Zhang, C., Chai, J.: Forecasting hourly PM2.5 concentrations based on decomposition-ensemble-reconstruction framework incorporating deep learning algorithms. Data Science and Management. 6, 46–54 (2023).
[15]. Teng, M., Li, S., Xing, J., Song, G., Yang, J., Dong, J., Zeng, X., Qin, Y.: 24-Hour prediction of PM2.5 concentrations by combining empirical mode decomposition and bidirectional long short-term memory neural network. Science of the Total Environment. 821, (2022).
[16]. Sun, C., Li, J., Sulaiman, R., Alotaibi, B.S., Elattar, S., Abuhussain, M.: Air Quality Prediction and Multi-Task Offloading based on Deep Learning Methods in Edge Computing. J. Grid Comput. 21, (2023).
[17]. Aman, N., Manomaiphiboon, K., Pengchai, P., Assareh, N., Dejchanchaiwong, R.: High-resolution spatial modeling of traffic-related air pollution in Bangkok, Thailand. Atmos. Environ. 246, 118090 (2021).
[18]. Dhammapala, R.: Analysis of fine particle pollution data measured at 29 US diplomatic posts worldwide. Atmos. Environ. 213, 367–376 (2019).
[19]. Huang, N., Shen, Z., Long, S., Wu, M., Shih, H., Zheng, Q., Yen, N.-C., Chao Tung, C., Liu, H.: The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences. 454, 903–995 (1998).
[20]. Kingma, D., Ba, J.: Adam: A Method for Stochastic Optimization. In: International Conference on Learning Representations (2015).
[21]. Espressif Systems: ESP-IDF Programming Guide: Partition Tables, https://docs.espressif.com/projects/esp-idf/, last accessed 2026/07/11.