Sensor Calibration Model Balancing Accuracy, Real-time, and Efficiency

Jinyong Yun[1], **Hyungjin Kim**[1], **Seokho Ahn**[1], **Euijong Lee**[2], **Young-Duk Seo**[1*]

[1]Department of Electrical and Computer Engineering, Inha University, Incheon 22212, South Korea
[2] Departments of Computer Science, Chungbuk National University, Chungbuk 28644, South Korea
12191632Y@inha.edu, flslzk@inha.edu, sokho0514@inha.edu, kongjjagae@cbnu.ac.kr, mysid88@inha.ac.kr

Abstract

Most on-device sensor calibration studies benchmark models only against three macroscopic requirements (i.e., accuracy, real-time, and resource efficiency), thereby hiding deployment bottlenecks such as instantaneous error and worst-case latency. We therefore decompose this triad into eight microscopic requirements and introduce SCARE (**S**ensor **C**alibration model balancing **A**ccuracy, **R**eal-time, and **E**fficiency), an ultra-compressed transformer that fulfills them all. SCARE comprises three core components: (1) Sequence Lens Projector (SLP) that logarithmically compresses time-series data while preserving boundary information across bins, (2) Efficient Bitwise Attention (EBA) module that replaces costly multiplications with bitwise operations via binary hash codes, and (3) Hash optimization strategy that ensures stable training without auxiliary loss terms. Together, these components minimize computational overhead while maintaining high accuracy and compatibility with microcontroller units (MCUs). Extensive experiments on large-scale air-quality datasets and real microcontroller deployments demonstrate that SCARE outperforms existing linear, hybrid, and deep-learning baselines, making SCARE, to the best of our knowledge, the first model to meet all eight microscopic requirements simultaneously.

1 Introduction

With the widespread adoption of Internet of Things (IoT) devices, it is now feasible to deploy large-scale sensor networks for environmental monitoring. To keep costs low, these networks often rely on inexpensive sensors, which are prone to nonlinear distortions, noise, and sampling delays (Koziel et al. 2025). Consequently, a post-processing calibration step for low-cost sensor data is essential, and, in particular, real-time on-device calibration at the edge is critical. On-device sensor calibration upgrades low-cost sensor by mapping raw readings from an inaccurate, low-cost one to values that align with those of a high-quality reference sensor. Traditionally, research on on-device sensor calibration has focused on three requirements: accuracy, real-time performance, and resource efficiency (Ahn et al. 2025).

However, these macroscopic requirements alone fail to capture the subtle bottlenecks that arise during real-world deployment, such as sampling delays, and hardware (HW) compatibility issues. For example, a model with a low RMSE (Root Mean Squared Error) may still incur large instantaneous errors when environmental conditions change abruptly. (Wan et al. 2020). To bridge this gap, we decompose the three conventional triad into eight microscopic requirements (Section 2.1). All eight must be satisfied to guarantee trustworthy on-device operation, but existing calibration models typically fulfill only a subset of them (Concas et al. 2021; Villanueva et al. 2023; Ahn et al. 2025).

To address these challenges, we propose SCARE (**S**ensor **C**alibration model balancing **A**ccuracy, **R**eal-time, and **E**fficiency), an ultra-compressed transformer designed to satisfy all eight requirements on resource-constrained devices such as microcontroller units (MCUs). SCARE first applies a bin-level patching scheme that groups tokens into bins and captures salient patterns spanning multiple bins, thereby lowering attention cost without significant information loss (Section 3.3). It then leverages an efficient bitwise attention mechanism to reduce energy consumption by eliminating redundant operations (Section 3.4). Finally, SCARE adopts an efficient optimization strategy that delivers robust performance at low-cost without complex auxiliary objectives during training (Section 3.5). Collectively, these designs make SCARE compatible with practical HW while preserving the essential balance among accuracy, real-time, and resource efficiency.

Our main contributions are as follows:

- **Discovery**: We decompose the three overarching requirements into eight fine-grained requirements, providing the first comprehensive checklist for on-device calibration.
- **Model**: We present SCARE, an ultra-compressed transformer that couples a Sequence Lens Projector with Efficient Bitwise Attention to deliver high accuracy and efficiency on resource-constrained MCUs.
- **Proof**: Experiments on a large-scale air-quality dataset and real embedded-device deployments demonstrate that SCARE meets all eight requirements and achieves state-of-the-art calibration performance.

2 Related Work

We identify eight microscopic requirements for on-device sensor calibration and assess the degree to which existing

*Corresponding author.

Category	Model Type	Model	Accuracy		Real-time			Resource Efficiency		
			Mean	Instant	Mean	Max	Sampling	Complexity	Size	Compatibility
Timeseries Forecasting	Linear	DLinear	O	X	O	O	X	O	O	O
	Transformer	PatchTST	O	X	△	O	X	O	△	O
	Hybrid	Mamba	O	X	△	O	X	O	△	O
Hash-based	Trainable Hash	Ecoformer	△	X	O	O	X	O	O	X
Calibration	Linear	NLinear	X	X	O	O	O	O	O	O
	Hybrid	SenDaL	O	O	△	X	O	X	X	△
	Transformer	TESLA	O	O	△	O	O	O	△	△
	Transformer	**SCARE**	O	O	O	O	O	O	O	O

Table 1: Comparison of related studies including SCARE against the eight microscopic requirements. **Symbols**: O = Satisfied, △ = Partially satisfied, X = Not satisfied. **Accuracy**: Mean / Instant = mean accuracy / instantaneous accuracy. **Real-time**: Mean / Max = mean inference latency / maximum inference latency. **Resource efficiency**: Complexity, Size, Compatibility = computational complexity, memory size, and hardware compatibility, respectively.

approaches satisfy them, as summarized in Table 1.

2.1 Microscopic Requirements

Most on-device sensor-calibration studies benchmark their models using only three broad criteria: accuracy, real-time performance, and resource efficiency (Rahman et al. 2020; Ahn et al. 2024, 2025). This high-level triad obscures important challenges during deployment. It hides fine-grained bottlenecks such as instantaneous error spikes and worst-case latency, and ignores the strict memory and energy budgets of MCUs. Consequently, models that look competitive on paper can still underperform in real-world deployments (Ibrahim et al. 2024).

To close these gaps, we define eight microscopic requirements and conduct an in-depth analysis of operational bottlenecks across three representative model classes: time-series forecasting, hash-based, and sensor calibration models. For every requirement, Table 1 assigns one of three ratings: 'Satisfied (O)' when the criterion is explicitly built into the design and empirically verified; 'Partially satisfied (△)' when it is considered but either lacks rigorous experimentation or underperforms in validation; and 'Not satisfied (X)' when it is entirely omitted.

The eight microscopic requirements map onto three macroscopic dimensions (i.e., accuracy, real-time performance, and resource efficiency) as follows:

- **Accuracy**: Prior works (Concas et al. 2021; Ahn et al. 2024) usually report only mean error. A rigorous evaluation must also include instantaneous error, which is the model's responsiveness to abrupt changes, because a model optimized solely for mean error can still miss local spikes and incur large transient mistakes.
- **Real-time**: A single latency figure is insufficient to evaluate real-time performance. Even when the mean inference latency may be low, it can suddenly jump at certain times, raising the maximum inference latency. This can make the sampling interval unstable or excessively long, breaking the timing between data points (Yang et al. 2021; Zhao et al. 2025). Therefore, comprehensive assessment requires mean inference delay, maximum inference delay, and sampling interval delay.
- **Resource efficiency**: This dimension decomposes into computational complexity, memory footprint, and HW compatibility. High complexity quickly exhausts the limited CPU cycles and battery budget on edge devices (Ni, Wu, and Wang 2025; Liu et al. 2022), and oversized models overflow on-chip memory, forcing slower off-chip accesses (Lin et al. 2020). Finally, mismatched HW compatibility leaves unsupported operators or memory layouts that stall or fail on the MCU (Liang et al. 2023).

2.2 Time-series Forecasting Models

Time-series forecasting models are usually optimized solely to minimize average error, yet this narrow objective prevents them from achieving instantaneous accuracy. Moreover, nonlinear operations, especially self-attention, cause inference latency to spike as input sequences lengthen, destabilizing the sampling interval and preventing the models from meeting real-time sampling requirements.

DLinear (Zeng et al. 2023) lowers computational complexity with purely linear predictors, but its inability to model nonlinear patterns leads to large instantaneous errors. PatchTST (Nie et al. 2023) reduces sequence length by grouping tokens, but still incurs quadratic self-attention cost and cannot guarantee a stable sampling interval under resource constraints. Mamba (Dao and Gu 2024) employs a selective state-space structure with time-varying matrices to capture long-term patterns efficiently, but it requires specially optimized kernels when parallel processing units (*e.g.,* GPUs) are unavailable and often suffers from high memory consumption and computational latency on resource-constrained devices.

Our contributions. SCARE integrates a patching-based sequence compression with bitwise attention to ensure low computational complexity while maintaining structural conciseness that guarantees high accuracy with minimal layers, effectively addressing the sampling interval delay.

2.3 Hash-based Models

Hash-based models (Kitaev, Kaiser, and Levskaya 2020; Liu et al. 2022) reduce the computational cost of attention by grouping tokens or approximating similarities with hash codes, but they inherently risk degrading accuracy. Hash functions that cannot adapt quickly to input distribution shifts are vulnerable to sudden changes, and sampling delays can force the model to generate hash keys from outdated inputs, causing transient prediction errors to increase sharply. Moreover, most hash-based models have yet to be evaluated on resource-constrained devices such as MCUs.

EcoFormer (Liu et al. 2022) projects queries and keys into a lower-dimensional space via hashing, achieving linear-complexity attention and notable energy savings. However, the representational capacity diminishes on short sequences, leading to significant performance degradation. In addition, the model depends on specialized GPU kernels for its auxiliary operations, which limits practical adoption on resource-constrained devices.

Our contributions. SCARE uses a sequence-wide patching strategy that preserves accuracy while remaining compatible with resource-constrained HW. In addition, its dynamically updated bitwise attention addresses sampling interval delay and adapts robustly to sudden distribution shifts.

2.4 Sensor Calibration Models

Sensor calibration models have largely focused on macroscopic requirements for practical IoT systems. NLinear (Zeng et al. 2023; Ahn et al. 2024) normalizes each time-series to its final value, yet its strictly linear design incurs large errors during nonlinear fluctuations. SenDaL (Ahn et al. 2024) blends linear and deep-nonlinear components to reduce mean error, but the added depth inflates worst-case latency. TESLA (Ahn et al. 2025) lowers both mean and instantaneous error through logarithmic patching and an efficient attention mechanism, but its memory layout conflicts with MCU's Direct Memory Access (DMA) functions, undermining HW compatibility. Consequently, existing approaches continue to face fundamental trade-offs among accuracy, real-time performance, and resource efficiency.

Our contributions. SCARE satisfies all eight microscopic requirements by reducing both mean and instantaneous errors through a novel patching strategy and by refining its attention mechanism to guarantee real-time performance and HW compatibility on resource-constrained devices.

3 Efficient On-Device Sensor Calibration Model: SCARE

This section first defines the sensor calibration problem and introduces SCARE (**S**ensor **C**alibration model balancing **A**ccuracy, **R**eal-time, and **E**fficiency), a method designed to satisfy eight core requirements across three key challenges.

3.1 Definition of Sensor Calibration Problem

Sensor calibration is the process of comparing low-cost sensor outputs with reference sensor measurements, learning corrections to reduce distortion and bias, and improve accuracy. (Yu et al. 2020; Ahn et al. 2024, 2025)

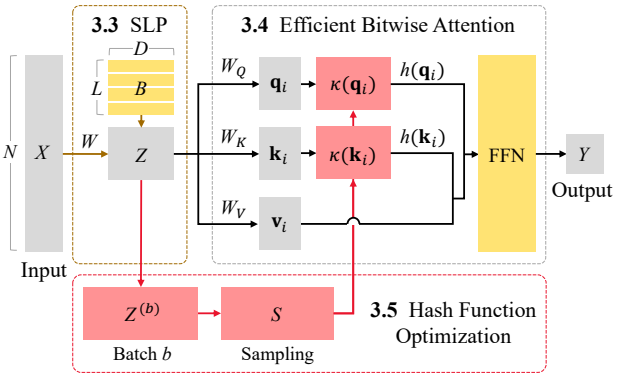

Figure 1: Overview of SCARE, composing (i) Sequence Lens Projector (SLP), (ii) Efficient Bitwise Attention (EBA), and (iii) Hash Function Optimization.

The calibration model is trained by comparing the low-cost sensor's input vectors $\mathbf{z}_j$ against simultaneous measurements r_j from a high-precision reference sensor Y. Defining a neural network $f(\cdot\,;\theta)$ with parameters θ, we solve:

$$\theta^* = \arg\min_{\theta\in\Theta} \sum_{j\in\mathcal{T}} \mathcal{J}\big(r_j,\ f(\mathbf{z}_j;\theta)\big) \tag{1}$$

where $\mathcal{J}$ is an objective function.

Reference sensors can reduce bias due to distribution mismatch, and eight additional microscopic indicators of accuracy, real-time, and resource efficiency must be considered when deploying models to embedded devices.

3.2 Overview of SCARE

Figure 1 illustrates the overall structure of SCARE (**S**ensor **C**alibration model balancing **A**ccuracy, **R**eal-time, and **E**fficiency) for on-device sensor calibration in IoT systems. SCARE is an ultra-compressed transformer designed to satisfy eight microscopic requirements across accuracy, real-time, and resource efficiency.

SCARE constructs dual embeddings using local and global weight matrices, following the approach of Ahn et al. 2025, to capture both instantaneous sensor variations and long-term trends while reducing the input complexity. The input is then processed through three core components. First, we propose the Sequence Lens Projector (SLP) (Section 3.3) to compress entire sequences, which compresses sequences across multiple focal distances to reduce computational complexity. Next, we introduce Efficient Bitwise Attention (EBA) (Section 3.4), which enables efficient computation through binarized hash codes. EBA replaces expensive multiplications with efficient additions, significantly reducing energy consumption for these resource-constrained environments. Finally, we employ effective optimization strategies (Section 3.5) through dynamic sampling and kernel-based approximation without complex auxiliary objectives. These components work as integrated modules, making SCARE applicable to practical sensor calibration while satisfying all eight microscopic requirements.

3.3 Sequence Lens Projector

Applying attention directly to long time-series data incurs a computational cost of $\mathcal{O}(N^2)$. To mitigate this, various patching methods have been proposed to compress sequence-level information to improve both accuracy and efficiency (Chen et al. 2025; Nie et al. 2023). Most studies rely on hard pooling, which divides each segment into fixed-size bins and calculates representative values using averages or weighted sums (Qiu et al. 2025).

This approach has fundamental limitations. First, segment-wise processing makes it difficult to capture sudden changes across bin boundaries. Second, the process of batch-wise synthesis across the entire sequence incurs imbalanced information levels between bins, causing important patterns to be averaged out and lost. Recently, TESLA's proposed logarithmic patching enables flexible pooling, but the non-uniform patch lengths hinder the implementation of optimized computational structures. Moreover, its incompatibility with DMA and vector operations leads to inference delays on practical HW devices such as MCUs.

To address these issues, we propose Sequence Lens Projector (SLP). The core idea is that while existing methods are similar to observing the sequence through multiple small lenses with limited scope, SLP observes the entire sequence through a single large lens with multiple focal distances simultaneously. By assigning global bias to each bin, SLP ensures that bins reflect global context rather than relying solely on local features, allowing significant patterns to contribute across multiple bins. This design reduces boundary loss by preserving patterns that span across adjacent bins, and alleviates inter-bin imbalance by preventing certain bins from becoming over- or under-represented. Additionally, SLP utilizes a single matrix structure to optimize all bins jointly, achieving efficient information compression without additional parameters.

Specifically, SLP operates as follows:

$$Z = \mathrm{SLP}(X) = \left[W^\top X + B\right] \in \mathbb{R}^{L\times D}, \tag{2}$$

where $X \in \mathbb{R}^{N\times D}$ is the input, $W \in \mathbb{R}^{N\times L}$ is the compression weight matrix, and $L = \lceil \log_2 N \rceil$ is the number of lenses. Each lens is represented by a column vector of W, indicating the importance of each time step. The bias matrix $B \in \mathbb{R}^{L\times D}$ provides distinct offsets for each feature dimension and lens, improving accuracy.

During training, SLP automatically optimizes W and b through gradients of the calibration loss. This reduces mean error and prevents sudden spikes in instantaneous error during abrupt signal changes, ensuring stable performance. Consequently, SLP improves both accuracy and efficiency while ensuring compatibility and minimizing inference delays even in resource-constrained environments such as MCUs.

3.4 Efficient Bitwise Attention

In sensor calibration, nonlinear models typically leverage standard multi-head self-attention (MHSA) due to their compressed tokens through patching (Ahn et al. 2025). However, this approach may incur computational overhead in resource-constrained environments such as on-device deployment scenarios. In this regard, we propose Efficient Bitwise Attention (EBA) that employs hash attention to optimize energy and memory efficiency and reduces unnecessary operations practical sensor calibration.

Hash attention reduces the cost of self-attention by converting tokens into hash codes so that only similar tokens interact (Kitaev, Kaiser, and Levskaya 2020; Liu et al. 2022). We first leverage the sign function (Rastegari et al. 2016; Liu et al. 2018) and straight-through estimator (STE) to replace multiplications with bitwise operations, reducing computational cost and energy consumption. And we employ the radial basis function (RBF) kernel (Liu et al. 2022) during training to improve the precision of hash approximation. Although the RBF kernel requires additional computation, the added cost is limited due to the input sequence compressed by SLP.

Formally, given samples $S = [\mathbf{s}_1, \cdots, \mathbf{s}_m] \in \mathbb{R}^{m\times D}$ (as detailed in Section 3.5) with projection matrix $A \in \mathbb{R}^{m\times c}$, the hash function $h : \mathbb{R}^D \to \{-1, +1\}^c$ used to binarize the vector $\mathbf{x}$ to a c-bit binary vector is defined as:

$$h(\mathbf{x}) = \mathrm{sign}\left[\left(\boldsymbol{\kappa}(\mathbf{x}) - \mu \cdot \mathbf{1}\right)^\top A\right] \tag{3}$$

where $\kappa(\mathbf{x}) = \left[\kappa(\mathbf{x}, \mathbf{s}_1), \cdots, \kappa(\mathbf{x}, \mathbf{s}_m)\right]^\top \in \mathbb{R}^m$ is a vector consisting of RBF kernel values between the vector $\mathbf{x}$ and the support samples S. $\mu = \frac{1}{m}\sum_{i=1}^{m} \kappa(\mathbf{x}, \mathbf{s}_i)$ is the mean kernel value, used as a zero-centering normalizer, and $\mathbf{1} \in \mathbb{R}^m$ a vector of all ones.

For pattern analysis, we adopt single-head self-attention instead of MHSA to simplify attention calculation. This is because SLP-based compressed hash embeddings provide sufficient information. Specifically, this contributes to optimization by modularizing SLP-based feature processing and single-head attention-based pattern analysis.

Formally, in the previous section, we obtain representations for L bins as $\mathrm{SLP}(X) = [\mathbf{z}_1, \cdots, \mathbf{z}_L]$. Using these bin representation, we compute the query $\mathbf{q}_i = \mathbf{z}_i W_Q$, key $\mathbf{k}_i = \mathbf{z}_i W_K$, and value $\mathbf{v}_i = \mathbf{z}_i W_V$ for each position i, where $W_Q, W_K, W_V \in \mathbb{R}^{D\times D}$ is a learnable parameter. Then the attention for query $\mathbf{q}_j$, based on keys $K = \{\mathbf{k}_i\}$ and values $V = \{\mathbf{v}_i\}$, is defined as:

$$\mathrm{Attn}(\mathbf{q}_j, K, V) = \frac{h(\mathbf{q}_j)^\top \mathbf{M}(K, V)}{h(\mathbf{q}_j)^\top \overline{\mathbf{k}}} \tag{4}$$

where $\mathbf{M}(K, V) = \sum_{i=1}^{L} h(\mathbf{k}_i) \otimes \mathbf{v}_i$ denotes the key–value memory map, and $\overline{\mathbf{k}} = \sum_{i=1}^{L} h(\mathbf{k}_i)$ is the sum of the key vector. We omit the bias term in attention for clarity.

Note that this formulation complements the SLP mechanism, reducing attention complexity to $\mathcal{O}(\log N)$. Furthermore, since both the inner and outer products are reduced to signed additions over binary vectors. For example, multiplying by 1 preserves the original value, while multiplying by -1 simply flips its sign. Moreover, the memory map $\mathbf{M}(K, V)$ and the key sum $\overline{\mathbf{k}}$ can be precomputed and reused without modification. As a result, SCARE improves both computational efficiency and HW compatibility.

Finally, we employ a standard feed-forward network (FFN). While recent SOTA approach (Ahn et al. 2025) has demonstrated performance using a single linear layer as a FFN, we leverage the FFN to reconstruct meaningful representations from the binarized computations. Although this incurs additional overhead, it remains efficient due to the significant cost reductions achieved through token compression and hash attention.

3.5 Hash Function Optimization

Optimization of the binary hash function h presents two major challenges. First, the discrete nature of the sign function causes gradient propagation problems, which we address with STE. Second, the exponential growth of possible hash code combinations makes training unstable. To mitigate this problem, existing research has introduced auxiliary learning such as Hamming affinity (Liu et al. 2022), but this increases training costs and hinders end-to-end learning.

In this study, we utilize the reduced token count by SLP as a key solution. The compressed tokens significantly reduce the search space of hash code combinations, allowing sufficient binarization performance to be achieved with RBF kernel-based similarity approximation alone. Therefore, effective learning is possible with the objective function alone without complex auxiliary learning. Nevertheless, optimization using the full dataset still requires considerable training time and computational cost.

To reduce this, we propose a dynamic support set sampling strategy that randomly extracts a small number of samples from each mini-batch during training. Specifically, from lens representation $Z^{(b)} = \mathrm{SLP}(X^{(b)})$ in each mini-batch b, we uniformly extract m samples to construct $S^{(b)} = \{s_1^{(b)}, s_2^{(b)}, \ldots, s_m^{(b)}\}$. Notably, this significantly reduces computational overhead while reflecting diverse query-key relationships, and achieves robust performance without overfitting to specific patterns. During inference, we use a fixed set to improve efficiency. We omit the batch index b in Section 3.4 for simplicity, because all batches follow the same procedure.

Consequently, this study effectively solves the hash function optimization problem by integrating SLP-based compressed token structure with STE, RBF kernel, and support set sampling strategy. It supports end-to-end learning without separate auxiliary learning, and ensures stability and reliability in extreme situations such as sudden input changes, enabling efficient deployment in practical sensor calibration.

4 Experiment

We introduce the datasets and training setup, then outline baselines, evaluation metrics, and implementation details.

Evaluation setup. Following the experimental protocol of (Ahn et al. 2025), our experiments use a large-scale calibration dataset comprising sensor readings from three cities—Antwerp (**Ant.**), Oslo (**Oslo**), and Zagreb (**Zag.**)—each recording three particulate-matter measures (**PM_{10}**, **$PM_{2.5}$**, and **PM_1**). We assume each region contains multiple sensors of the same type, each uniquely named. Within each region and for each feature, we sort the sensors alphabetically, assign the second-to-last sensor to validation, the last to testing, and use all others for training.

Baselines. To compare the performance of SCARE, we carefully select commonly used state-of-the-art (SOTA) models across various tasks in IoT systems. For time-series forecasting, we include DLinear (Zeng et al. 2023), PatchTST (Nie et al. 2023), and Mamba (Dao and Gu 2024), which have demonstrated strong predictive capabilities but have not been thoroughly evaluated for sensor calibration. For hash-based approach, we adopt EcoFormer (Liu et al. 2022), which leverages trainable hash attention to generate compact query–key codes. For sensor calibration, we compare NLinear (Zeng et al. 2023), SenDaL (Ahn et al. 2024), and TESLA (Ahn et al. 2025), with TESLA achieving SOTA results in this task (Detailed explanations for each baseline in the supplementary material).

Evaluation metrics. Performance is evaluated as follows. Accuracy is measured by RMSE to penalize large deviations. Responsiveness to sudden changes is assessed via RMSE over the top 5% of time points with largest absolute first difference $|\Delta x|$. Real-time behavior is measured by 50 repeated inferences to obtain mean, max, and jitter (sample standard deviation) of latency, capturing throughput, deadline safety, and timing variation. Computational cost is expressed in $\mathcal{O}(\cdot)$ to show how inference time scales with input length independent of HW. Memory demand is estimated as the peak activation size (sum of retained tensor elements $\times$ bytes per element) in kilobytes. HW suitability is measured by the maximum CPU utilization in repeated inference to indicate typical load and potential short-term spikes.

Implementation. The on-device environment was CPU-based (AMD EPYC 7513). Models were developed with TensorFlow 2.14 and trained for 10 epochs using 32 batch size, mean squared error (MSE) loss, and Adam optimizer, following common setups (Liu et al. 2024; Ahn et al. 2025). Trained models were converted to TensorFlow Lite FlatBuffers and deployed on an Arduino Nano 33 BLE Sense.

5 Experimental Results

This section details our experimental results.

Microscopic requirement. Table 2 demonstrates that SCARE outperforms existing baseline models in meeting all microscopic requirements. It achieves the highest performance across pollutant types even under rapid and temporary distribution shifts. While linear models such as DLinear and NLinear typically enable faster inference, SCARE offers near-linear response times with substantial accuracy gains. In terms of resource efficiency, it exhibits the lowest computational complexity among nonlinear models and surpasses all benchmarks in both maximum CPU utilization and minimum runtime memory requirements.

Ablation study. We introduced six architectural combinations and evaluated their impacts, as summarized in Table 3. First, switching from simple tokenization (Exp 1) to the SLP patching method (Exp 2) shown better results across all metrics, including accuracy. Second, integrating local and global

		Accuracy				Real-time			Resource Efficiency		
		Mean			Instant	Mean	Max	Sampling	Complexity	Memory	Compatibility
Category	Models	RMSE ↓ (μg m^{-3}) PM_{10}	$PM_{2.5}$	PM_1	Top-5% RMSE ↓ (μg m^{-3})	Mean Latency ↓ (ms)	Max Latency ↓ (ms)	Latency Jitter ↓ σ (ms)	Computational Complexity ↓ (Big-O)	Peak Activation ↓ (KB)	CPU Util Max ↓ (%)
Timeseries Forecasting	DLinear	17.81	8.41	4.49	35.23	1.31	1.44	**1.83**	$\mathbf{O(N)}$	28	50.0
	PatchTST	15.23	6.29	2.71	31.34	5.60	6.63	9.75	$\mathbf{O((N/P)^2)}$	139	100.0
	Mamba	16.06	6.37	3.14	32.80	269.45	272.43	15.96	$\mathbf{O(N)}$	17616	14.6
Hash	Ecoformer	16.90	7.13	3.51	35.41	14.43	15.28	8.18	$\mathbf{O(N)}$	1005	100.0
Calibration	NLinear	19.33	9.96	6.13	34.36	**1.28**	**1.38**	3.03	$\mathbf{O(N)}$	**18**	100.0
	SenDaL	15.76	6.87	3.14	38.77	32.09	118.23	18.32	$\mathbf{O(N^2)}$	9614	9.5
	TESLA	14.21	6.19	2.82	27.92	2.14	2.47	4.47	$\mathbf{O((\log N)^2)}$	121	8.8
	SCARE	**14.03**	**5.67**	**2.51**	**27.66**	1.63	1.72	2.11	$\mathbf{O(N \log N)}$	115	**7.1**

Table 2: Evaluation of category-specific representative models with respect to eight microscopic requirements. Calibration accuracy was computed as the mean RMSE over a 360 length window for each pollutant (PM10, PM2.5, and PM1) using data from three sites (Ant., Oslo, and Zag.). Detailed definitions of the remaining metrics are provided in section 4. For each metric, the highest performance is marked in **bold**, and the second highest in underlined.

Exp	Methodology				RMSE ↓			Max Latency ↓	Peak Activation ↓	CPU Util Max ↓
	Embedding	Patching	Attention	Sampling	PM_{10}	$PM_{2.5}$	PM_1			
1	Local	None	MHA	X	14.87	**5.49**	2.59	120.32	9444.9	100.0
2	Local	SLP	MHA	X	14.65	6.05	2.56	**1.69**	**109.3**	66.7
3	Local + Global	SLP	MHA	X	14.30	5.90	2.82	3.01	133.3	75.0
4	Local + Global	SLP	HA	X	14.13	5.93	3.11	4.23	124.2	13.7
5	Local + Global	SLP	EBA	X	14.18	5.97	2.90	3.15	118	12.4
6*	Local + Global	SLP	EBA	O	**14.03**	5.67	**2.51**	1.72	115.0	**7.1**

Table 3: Comparison of architectural combinations in terms of mean accuracy, maximum inference latency, memory, and HW compatibility when the input window length is 360. Mean accuracy is calculated as RMSE averaged across the three regions (Ant., Oslo, Zag.) for each particulate matter category. Exp 6* is our proposed model. In Attention part, MHA denotes multi-head attention, HA denotes conventional hash attention (Liu et al. 2022), and EBA denotes efficient bitwise attention.

embeddings (Exp 3) improved mean accuracy compared to using only local embeddings (Exp 2), indicating these embeddings complement each other effectively. Third, substituting conventional hash attention (Exp 4) with proposed EBA (Exp 5 and 6) mitigated information loss while significantly reducing maximum inference latency, peak CPU utilization, and minimum runtime memory requirements.

Model adaptability. Table 2 evaluates adaptability of SCARE to sudden input changes. Segments with error deviations from prior inputs were sampled with predefined thresholds. Typically, Ant contributes to model inaccuracies with small error deviations, while Zag complicates calibration with segments with large error deviations. Specifically, DLinear and EcoFormer demonstrate that conventional linear and nonlinear approaches struggle with the calibration task, and the SOTA model TESLA shows limited performance on Ant. Conversely, SCARE consistently offers robust performance across all cities and thresholds, with its advantage over other models becoming increasingly evident at various deviation rates.

On-device performance. To assess on-device performance in IoT environments, we evaluated the models on a MCU for deployment and inference latency, as summarized in Figure 3. DLinear is compact and capable of handling long sequences but provides poor accuracy. TESLA showed high accuracy with inference speeds close to DLinear, but it failed to operate on ultra-long time-series inputs (1440 sequence length). Conversely, SCARE not only achieved SOTA accuracy but also maintained latency similar to DLinear across all sequence lengths, while achieving a smaller Flatbuffer size than TESLA.

6 Discussions and Limitations

We discuss the implications and challenges of our findings.

Robust accuracy. As shown in Table 3, shifting from simple tokenization (Exp 1) to the SLP operation method (Exp 2) led to a statistically significant increase in mean accuracy. This improvement indicates that SCARE more effectively preserves salient patterns within each bin, resulting in overall performance gains. Moreover, our dynamic sam-

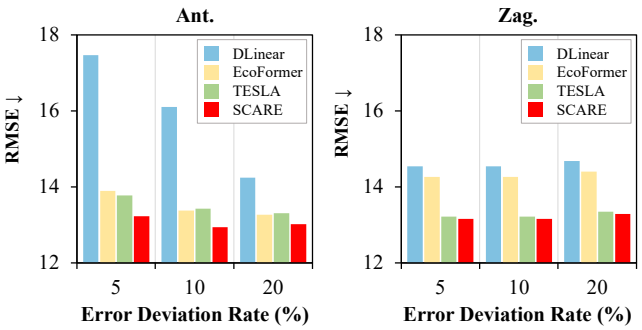

Figure 2: RMSE comparisons within ±30 time points around change points, under error change thresholds of 5%, 10%, 20% on the Ant. and Zag. datasets.

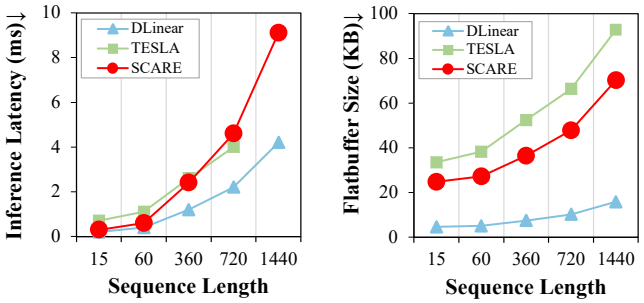

Figure 3: Performance on Arduino Nano 33 BLE Sense. The graphs show inference latency and flat-buffer size across varying windows (15, 60, 360, 720, 1440). Missing points indicate that on-device measurement was not possible.

pling strategy significantly enhances model adaptability, as evidenced by substantial improvements of instant accuracy shown in Figure 2 and Table 3. These results demonstrate that SCARE not only consistently outperforms both linear and nonlinear baselines but also ensures robustness under sudden distribution shifts.

Real-time performance. SCARE achieves the lowest mean and max latency among all nonlinear models in Table 2. This efficiency stems from reduced token count processed by the SLP stage, which significantly decreases bitwise attention operation during the EBA stage. Moreover, SCARE exhibits minimal sampling jitter in Table 2, indicating stable and consistent calibration cycles.

MCU environments require optimized architecture due to the resource constraints such as limited parallelism. While Mamba achieves linear complexity, it requires parallel operations that impose high RAM usage on CPU. Conversely, SCARE employs single-head attention to eliminate parallel operation, ensuring fast inference on MCUs without additional memory overhead.

HW compatibility. In Table 2, the CPU utilization results show that SCARE achieves the lowest CPU load and memory usage during inference. This efficiency results from the synergy between SLP operations and EBA mechanisms, which maintain linear computational complexity while minimizing overall computation. The SLP's reliance on a single weight matrix W and global bias for uniform bin handling integrates seamlessly with DMA and vector processing units, enabling highly efficient MCU execution. Table 3 confirms that applying SLP simultaneously reduces peak activated memory and CPU utilization when comparing Exp 1 (vanilla transformer) and Exp 2 (SLP). Moreover, Figure 3 demonstrates that while TESLA models fail to execute ultra-long sequences on Arduino due to memory constraints despite robust calibration performance on standard sequences, SCARE reliably handles extended data streams through its simplified EBA layer, confirming strong hardware compatibility for real-time embedded deployments.

Architectural trade-offs. SCARE is deliberately engineered as an ultra-compressed architecture optimized for embedded devices, intentionally omitting complex components such as a standard MHSA. While this structural simplification enables real-time processing in resource-constrained environments (e.g., MCUs), it imposes limitations on representational capacity and flexibility compared to elaborate models designed for high-end platforms. This design prioritizes real-time responsiveness and stability in resource-scarce settings, allowing SCARE to deliver consistent, efficient performance on low-specification hardware. In contexts with abundant computational resources, model complexity could be increased to pursue further accuracy gains; however, this study focuses on ensuring reliability in edge environments.

Learning limitations. SCARE relies on offline retraining and static TensorFlow Lite deployments, preventing devices from correcting seasonal or long-term sensor drift until the next firmware update. This limitation increases operation and maintenance costs and risks service outages in areas with poor connectivity. Future work will incorporate lightweight on-device continual learning through fine-tuning SLP-compressed parameters at low precision, along with server-edge federated calibration to enable real-time adaptation to sensor drift.

7 Conclusion

In this work, we introduce SCARE, an efficient on-device sensor calibration transformer for low-cost sensors. We first recognize that the conventional accuracy, real-time, resource efficiency triad masks real-world bottlenecks, and therefore decomposed it into eight microscopic requirements to better reflect the realities of on-device operation. SCARE meets eight core requirements through three core components: (i) SLP that effectively compresses time-series data while preserving cross-bin patterns, reducing attention complexity to O(N log N); (ii) EBA that reduces computational and memory complexity to near-linear scale through hash-attention mechanism with single-head attention; and (iii) hash function optimization, which a dynamic sampling strategy ensure both MCU compatibility and millisecond-level inference with robust performance. Evaluations on MCU

units deployments show that SCARE satisfies all eight microscopic requirements while simultaneously outperforming state-of-the-art linear, hybrid, and deep-learning calibrators in accuracy, real-time, and resource efficiency.

References

Ahn, S.; Kim, H.; Lee, E.; and Seo, Y.-D. 2024. SenDaL: An Effective and Efficient Calibration Framework of Low-Cost Sensors for Daily Life. *IEEE Internet of Things Journal*, 11(11): 20619–20630.

Ahn, S.; Kim, H.; Shin, S.; and Seo, Y.-D. 2025. Real-Time Calibration Model for Low-Cost Sensor in Fine-Grained Time Series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 3–11.

Chen, X.; Qiu, P.; Zhu, W.; Li, H.; Wang, H.; Sotiras, A.; Wang, Y.; and Razi, A. 2025. Sequence complementor: Complementing transformers for time series forecasting with learnable sequences. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 15913–15921.

Concas, F.; Mineraud, J.; Lagerspetz, E.; Varjonen, S.; Liu, X.; Puolamäki, K.; Nurmi, P.; and Tarkoma, S. 2021. Low-Cost Outdoor Air Quality Monitoring and Sensor Calibration: A Survey and Critical Analysis. *ACM Trans. Sen. Netw.*, 17(2).

Dao, T.; and Gu, A. 2024. Transformers are SSMs: Generalized Models and Efficient Algorithms Through Structured State Space Duality. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, 1–13. Vienna, Austria: PMLR.

Ibrahim, J.; Li, Y.; Chen, H.; and Yuan, D. 2024. Holistic evaluation metrics for federated learning. In *2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, 2282–2287. IEEE.

Kitaev, N.; Kaiser, Ł.; and Levskaya, A. 2020. Reformer: The efficient transformer. In *Proceedings of the International Conference on Learning Representations*.

Koziel, S.; Pietrenko-Dabrowska, A.; Wojcikowski, M.; and Pankiewicz, B. 2025. Efficient field correction of low-cost particulate matter sensors using machine learning, mixed multiplicative/additive scaling and extended calibration inputs. *Scientific Reports*, 15(1): 18573.

Liang, Y.; Wang, Z.; Xu, X.; Tang, Y.; Zhou, J.; and Lu, J. 2023. Mcuformer: Deploying vision tranformers on microcontrollers with limited memory. *Advances in Neural Information Processing Systems*, 36: 8501–8512.

Lin, J.; Chen, W.-M.; Lin, Y.; Gan, C.; Han, S.; et al. 2020. Mcunet: Tiny deep learning on iot devices. *Advances in neural information processing systems*, 33: 11711–11722.

Liu, J.; Pan, Z.; He, H.; Cai, J.; and Zhuang, B. 2022. Ecoformer: Energy-saving attention with linear complexity. *Advances in Neural Information Processing Systems*, 35: 10295–10308.

Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; and Long, M. 2024. iTransformer: Inverted Transformers Are Effective for Time Series Forecasting. In *International Conference on Learning Representations*.

Liu, Z.; Wu, B.; Luo, W.; Yang, X.; Liu, W.; and Cheng, K.-T. 2018. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *Proceedings of the European conference on computer vision (ECCV)*, 722–737.

Ni, C.; Wu, J.; and Wang, H. 2025. Energy-aware edge computing optimization for real-time anomaly detection in IoT networks. *Applied and Computational Engineering*, 139: 42–53.

Nie, Y.; H. Nguyen, N.; Sinthong, P.; and Kalagnanam, J. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *International Conference on Learning Representations*.

Qiu, T.; Xie, Y.; Niu, H.; Xiong, Y.; and Gao, X. 2025. Enhancing Masked Time-Series Modeling via Dropping Patches. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 20077–20085.

Rahman, M.; Rahman, M.; Hossain, M.; Al-Fuqaha, A.; and Wang, Y. 2020. A systematic review on performance evaluation metric selection method for IoT-based applications–ScienceDirect. *Journal of Network and Computer Applications*.

Rastegari, M.; Ordonez, V.; Redmon, J.; and Farhadi, A. 2016. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, 525–542. Springer.

Villanueva, E.; Espezua, S.; Castelar, G.; Diaz, K.; and Ingaroca, E. 2023. Smart Multi-Sensor Calibration of Low-Cost Particulate Matter Monitors. *Sensors*, 23(7): 3776.

Wan, C.; Santriaji, M.; Rogers, E.; Hoffmann, H.; Maire, M.; and Lu, S. 2020. Accurate learning for energy and timeliness. In *2020 USENIX annual technical conference (USENIX ATC 20)*, 353–369.

Yang, Z.; Nahrstedt, K.; Guo, H.; and Zhou, Q. 2021. Deeprt: A soft real time scheduler for computer vision applications on the edge. In *2021 IEEE/ACM Symposium on Edge Computing (SEC)*, 271–284. IEEE.

Yu, H.; Li, Q.; Geng, Y.-a.; Zhang, Y.; and Wei, Z. 2020. Airnet: A calibration model for low-cost air monitoring sensors using dual sequence encoder networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 1129–1136.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2023. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, 11121–11128.

Zhao, Z.; Hu, Y.; Yang, G.; Gong, Z.; Shen, C.; Zhao, L.; Li, W.; Liu, X.; and Qu, W. 2025. SLOpt: Serving real-time inference pipeline with strict latency constraint. *IEEE Transactions on Computers*.