# Optimizing TinyML with quantization and distillation for Indoor Localisation in Resource-Constrained Environments

**Muecid Islam Arian, Md. Mahamudul Hasan, Md. Nafiz Un Nabi Ishan**

*Department of Computer Science and Engineering, North South University*

**Email:** `muecid.arian@northsouth.edu`, `mahamudul.hasan02@northsouth.edu`, `nafiz.ishan@northsouth.edu`

**Abstract**

Artificial intelligence has grown at a never-before-seen pace, and recent developments in computing technologies have led to significant discoveries across numerous academic fields. However, the high computing requirements of conventional AI models lead to substantial expenses for energy use, hardware requirements, and carbon emissions. To overcome these constraints, Tiny Machine Learning (TinyML) has emerged as a viable platform that provides AI capabilities to extremely low-power and resource-constrained edge devices. By employing model compression, quantization, pruning, and knowledge distillation to provide efficient inference independent of cloud connectivity, TinyML lowers latency and improves data privacy. By examining important tactics, topologies, and optimization methods including Transformer and Mamba models, this paper offers a thorough overview of TinyML and AI on edge devices. Our review surveys existing research in the medical and IoT domains while highlighting challenges such as restricted computational power, energy efficiency, and on-device learning. Ultimately, this review identifies pathways focused on enhancing collaboration between edge intelligence and AI implementation.

## 1 Introduction

Artificial Intelligence (AI) has become an integral part of modern life—from gaming and healthcare monitoring to natural language processing (NLP), computer vision (CV), social media analytics, and autonomous driving. However, these capabilities come at a cost: training and deploying large-scale deep learning models require massive computational resources, specialized hardware, and high energy consumption. This challenge is especially critical in developing nations where such infrastructure is limited.

As AI models increase in size and complexity, their carbon footprint and operational costs rise proportionally. Even when cloud-based systems are used, compute time and energy usage remain significant barriers to sustainable AI deployment. To address these challenges, Tiny Machine Learning (TinyML) has emerged as a transformative paradigm that moves intelligence from centralized servers to low-power, resource-constrained **edge devices**.

TinyML enables neural networks to run efficiently on microcontrollers (MCUs) and embedded systems with power consumption in the milliwatt or even microwatt range—thousands of times lower than traditional CPUs or GPUs. This shift reduces latency and bandwidth requirements, enabling continuous and unplugged operation in environments with limited connectivity. Beyond computational efficiency, TinyML supports local data processing, minimizing privacy risks and aligning with global data protection standards.

Applications range from real-time industrial anomaly detection [6], to healthcare analytics and smart-home monitoring [5]. Research leveraging optimized architectures such as Transformer and Mamba demonstrates that TinyML can balance model accuracy and efficiency through quantization, pruning, and knowledge distillation—key enablers for real-world, on-device intelligence.

## 2 Background

### 2.1 Tiny Machine Learning (TinyML)

TinyML bridges embedded systems and artificial intelligence by enabling ML inference directly on MCUs and low-power IoT hardware. Unlike traditional cloud-based AI, TinyML performs inference locally, allowing models to handle image classification, speech recognition, and indoor localization using limited resources (typically under 1 MB Flash and 320 KB RAM) [1].

This approach overcomes IoT limitations related to high data transmission costs, delays, and energy consumption. Frameworks such as **TinyFormer**, **MCUNet**, and **SparseEngine** demonstrate how modern architectures—Transformers, CNN hybrids, and Mamba models—can be compressed for MCU deployment, enabling scalable and privacy-preserving edge intelligence [3, 4].

#### 2.1.1 Reducing Latency

TinyML reduces inference latency by processing data directly at the edge. For instance, BLE RSSI-based indoor localization allows position estimation on-device without relying on the cloud. **TinyFormer** achieved up to 12× faster inference than CMSIS-NN using sparsity-aware deployment [3]. Similarly, knowledge distillation (KD) enables smaller quantized student models to run faster while maintaining high accuracy [2].

### 2.1.2 Offline Capability

Offline functionality is vital in regions like Bangladesh with limited internet access. Edge AI enables autonomous decision-making without constant connectivity [1]. Devices such as ESP32 can infer user location using BLE signals entirely offline, ensuring continuity in healthcare and emergency systems.

### 2.1.3 Improving Privacy and Security

By performing inference locally, TinyML ensures sensitive data—like health or location information—remains on-device, reducing interception risks and aligning with privacy regulations. TinyFormer's edge-deployed inference pipeline exemplifies this privacy-preserving AI approach.

### 2.1.4 Low Energy Consumption

Energy efficiency is central to TinyML. Techniques such as quantization (FP32 $\rightarrow$ INT8) and block pruning significantly reduce multiply–accumulate operations, lowering energy use. Sparse Transformer inference consumed only 14% of CMSIS-NN's energy while maintaining 96% accuracy [3]. KD models trained by [2] achieved similar performance with up to 10$\times$ lower power use.

### 2.1.5 Reducing Cost

TinyML cuts costs by replacing high-end hardware with affordable microcontrollers like ESP32 and Raspberry Pi Pico (under $5). Open-source deployment tools such as TensorFlow Lite Micro eliminate licensing fees, fostering affordable, locally built solutions in developing regions.

## 3 Hardware Constraints

Embedded AI deployment in developing regions faces limitations in cost, energy, and component availability. Systems must balance processing power and efficiency while using locally available components.

## 4 Software Optimization

Due to constrained resources in MCUs like STM32 and ESP32, software optimization is critical for achieving real-time inference. The key objective is minimizing model size, com-

Table 1: Component Constraints and System Impact

| Component | Constraint | Impact on System | Availability (BD) |
|---|---|---|---|
| STM32L4 MCU | Limited RAM (64 KB) and moderate processing power | Restricts model size and inference complexity | High |
| ESP32-S3 | Variable Wi-Fi/BLE stability under interference | Possible data packet loss during transmission | High |
| BLE Beacon (JDY-23 / HM-10) | Short communication range (10–30 m) and signal attenuation in dense areas | Reduced RSSI accuracy in indoor environments | High |
| Raspberry Pi Pico W | Moderate availability and higher power demand | Suitable only as fixed gateway or logger | Medium |
| 3.7V Li-ion Battery | Limited operation time and recharging requirement | Affects portability and continuous monitoring | High |

putation, and energy use while maintaining accuracy. Two main approaches—**Knowledge Distillation** and **Quantization**—are most effective.

Table 2: Model Optimization Techniques for TinyML Deployment

| Technique | Description | Effect on Model | Deployment Benefit |
|---|---|---|---|
| Knowledge Distillation | Transfers knowledge from a large teacher model to a small student model | Reduces parameter count while preserving accuracy | Enables use of smaller models on low-power devices |
| Quantization | Converts 32-bit floating-point weights to 8-bit integers | Decreases model size and inference time | Minimizes memory use and improves on-device performance |

## 5 Recent Research in TinyML

Recent studies emphasize TinyML's adaptability in various domains. **Zeynali et al. (2025)** demonstrated non-invasive blood glucose estimation using PPG signals with ResNet34 and CNN-LSTM architectures, optimized for STM32 MCUs with real-time inference under 6.4 seconds. **Attri et al. (2025)** developed a TinyML-based MobileNetV2 model for plant disease detection, achieving 93.1% accuracy while maintaining high energy efficiency. **Suwannaphong et al. (2025)** explored quantization and distillation for indoor localization using Transformer and Mamba architectures, reducing memory by 50% while maintaining accuracy. These studies underline TinyML's expanding impact on healthcare, agriculture, and localization.

## 6 Conclusion

Tiny Machine Learning (TinyML) is reshaping the AI landscape by enabling low-power embedded devices to perform real-time inference efficiently. By combining quantization, knowledge distillation, pruning, and model compression, TinyML makes AI more accessible and sustainable. Studies like TinyFormer, Apprentice KD, and recent Mamba adaptations validate that edge AI can maintain accuracy within extreme memory limits while enhancing privacy, reducing latency, and minimizing energy use.

In the context of indoor localization, TinyML enables BLE RSSI-based systems to deliver accurate, low-cost, and offline positioning capabilities. The combination of quantization and distillation allows Transformer and Mamba models to operate effectively on 32–64 KB RAM devices such as ESP32 and STM32. As research evolves, future directions will include quantization-aware training, pruning, and hardware-software co-design to push the boundaries of edge intelligence further.

## References

[1] D. Situnayake and J. Plunkett, *AI at the Edge: Solving Real-World Problems with Embedded Machine Learning*. O'Reilly Media, 2022.

[2] A. Mishra and D. Marr, "Apprentice: Using Knowledge Distillation Techniques to Improve Low-Precision Network Accuracy," *arXiv preprint arXiv:1711.05852*, 2017.

[3] J. Yang et al., "TinyFormer: Efficient Sparse Transformer Design and Deployment on Tiny Devices," *IEEE Transactions on Computers*, 2025.

[4] "Optimising TinyML for Resource-Constrained Edge Environments," *IEEE Access*, 2023.

[5] T. Suwannaphong, F. Jovan, I. Craddock, and R. McConville, "Optimising TinyML with Quantization and Distillation of Transformer and Mamba Models for Indoor Localisation on Edge Devices," *Scientific Reports*, vol. 15, p. 10081, 2025.

[6] L. S. Martinez-Rau, Y. Zhang, B. Oelmann, and S. Bader, "TinyML Anomaly Detection for Industrial Machines with Periodic Duty Cycles," *IEEE Sensors Applications Symposium (SAS)*, 2024.

[7] M. Zeynali, K. Alipour, B. Tarvirdizadeh et al., "Non-invasive Blood Glucose Monitoring Using PPG Signals and Implementation with TinyML," *Scientific Reports*, 2025.

[8] I. Attri, L. K. Awasthi, and T. P. Sharma, “TinyML for Plant Disease Detection: Efficient Edge AI Solutions for Apple and Mango Leaves,” *Procedia Computer Science*, vol. 258, pp. 2870–2877, 2025.

[9] M. Pujari, A. Goel, and A. K. Pakina, “Efficient TinyML Architectures for On-Device Small Language Models: Privacy-Preserving Inference at the Edge,” *International Journal of Science and Technology*, vol. 3, no. 3, 2024.