International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

TinyML in Embedded Systems: A Review on Applications, Challenges, and Research Gaps

Mellyn Grace P. Asuncion:
Bachelor of Science in Computer Engineering
North Cotabato, Philippines

Shane S. Palencia:
Bachelor of Science in Computer Engineering
North Cotabato, Philippines

Heart Krysteljen R. Sabio
Bachelor of Science in Computer Engineering
Bukidnon, Philippines

Abstract- Tiny Machine (TinyML) is an emerging technology that aims to run machine learning models directly on low-power embedded devices with limited memory and computational resources, such as microcontrollers and Internet of Things (IoT) devices. Tiny Machine Learning (TinyML) differs from conventional machine learning techniques that offload processing to the cloud, requiring powerful computing resources, by enabling lightweight models that run interference on the device with low memory, energy, and computational needs. This facilitates edge-based data processing, enabling low-latency responses, enhanced privacy, and reduced data transmission costs.

This research provides a survey of Tiny Machine Learning (TinyML) for embedded system, including use cases, challenges and opportunities. It also offers a qualitative comparison between popular Tiny Machine Learning (TinyML) framework and methods such us TensorFlow Lite Micro, Edge Impulse, Arduino TinyML, PyTorch Mobile, and model optimization strategies (quantization and pruning). This analysis considers the performance metrics of memory and energy efficiency, scability, development simplicity, and support for model complexity.

The results shows that Tiny Machine Learning (TinyML) greatly improves the performance and efficiency of embedded systems in a range of applications, such as healthcare monitoring, smart homes, and industrial automation. Abut it also poses challenges such as hardware constraints, standardization, security, and the trade- off between model performance and efficiency. The research underscores the role of optimization techniques in facilitating deployment of Tiny Machine Learning (TinyML) and suggests further work should focus on efficient algorithm development, dedicated hardware and scalable Tiny Machine Learning (TinyML) architectures.

In conclusion, Tiny Machine Learning (TinyML) shows promise in revolutionizing embedded systems through the creation of smart, autonomous, and efficient devices. Ongoing research is likely to enhance system efficiency, stability and scalability, positioning Tiny Machine Learning (TinyML) as a critical technology in the development of edge computing and IoT systems.

I. INTRODUCTION

The world of embedded systems now dominates everyday life. Your watch and your thermostat together with factory assembly lines operating three states away use hardware that people considered inadequate for their time. System needs to maintain battery power for several months or multiple years between charging sessions. Machine learning was never designed for this world. The classic ML workflow requires real computing resources which include a beefy server and cloud infrastructure and systems that have extra processing capacity. The process of implementing that solution in a microcontroller presented multiple challenges. The majority of machine learning technology remained inaccessible during that time period.

Tiny Machine Learning (TinyML) has become a solution to this issue because it allows machine learning models to operate on microcontrollers and other lowpower hardware platforms. TinyML develops machine learning models which execute local inference on embedded systems while using the least amount of computational power. The TinyML overview in the Journal of STE Research shows that this method enables intelligent data processing to occur directly on IoT devices without needing extensive cloud connectivity. The integration of TinyML with embedded systems provides several advantages. The system achieves improved real-time decision-making capabilities while experiencing reduced power consumption and enhanced data protection through its decreased network communication requirements. The features of TinyML make it ideal for use in voice recognition and anomaly detection and predictive maintenance and smart agriculture and wearable health monitoring applications.

The implementation of machine learning on devices with severe resource restrictions creates multiple technical difficulties according to its advantages. Embedded systems have strict limitations in memory, storage, computational capability, and battery life. Researchers develop new techniques which include model compression and quantization and efficient neural network

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

designs to create machine learning systems that can function on microcontrollers. Review paper examines the current TinyML research in embedded systems and current technological trends with a view to establishing essential research gaps and describing the potential research directions in the future. The ability to comprehend these elements aids in directing upcoming developments in intelligent embedded system technology.

II. RELATED WORK

Researchers have investigated the use of TinyML technology for embedded systems and Internet of Things applications through multiple studies. The research studies work to enhance three essential elements of machine learning models which operate on devices that possess restricted computational capabilities.

Study [1] The researchers executed an extensive survey which studied different TinyML applications that included healthcare monitoring and smart agriculture and environmental sensing and anomaly detection systems. Their research established a TinyML research classification system which showed the present research directions and development frameworks and main obstacles that researchers face when using machine learning on embedded systems. The research primarily focused on research classification and secondary was given to research evaluation of implementation performance.

In a similar manner, Study [20] talked about the concepts of TinyML such as the optimization of neutral networks by making them smaller and their application to small computers. The paper described the application of TinyML of Internet of Things (IoT) applications such as wearable health monitors and smart homes. The research well-explained the designs and methods of TinyML, but it conducted very little experimentation on its concepts.

The other review Study [16] provided a review of tools, software challenges and system development challenges. The study looked at TinyML systems and provided a guide to current technologies used in machine learning on small devices.

Even though it gave a look at TinyML systems the research did not compare specific applications in detail. Study [7] introduced the TensorFlow Lite Micro system, which lets machine learning work on computers. Their work showed how small software can let neural network models work in small memory spaces. However, the system mainly just works for running models not for training them on the device because of hardware limits.

Study [19] implemented gesture and speech recognition using an Arduino Nano 33 BLE microcontroller with models trained on the Edge Impulse platform. Their study showed that TinyML models can do real-time gesture recognition and keyword spotting on low-power devices. Never the less the work was mostly a test, not a big system implementation. TinyML models can be used in areas like smart homes. TinyML systems have challenges, like limited memory and processing power. TinyML tools, like TensorFlow Lite Micro make it easier to use machine learning, on devices.

Study [4] TinyML can be used for gesture recognition and speech recognition using a kind of computer program on an Arduino microcontroller. They found that TinyML systems can fare well when it comes to identifying keywords achieving approximately 97% accuracy in activities such as the identification of particular words. The system could not process more complex speaking tasks but was only useful in simple ones such as identifying keywords.

Study [15] suggested a TinyML system to verify who is speaking. It used a learning model that could learn and adapt on small devices. Their research showed that it is possible to do some learning on the device. With this progress the approach needed more computing power and energy.

Study [6] did a review of research on TinyML and machine learning. They explained how TinyML works, from designing the model to putting it on a device and working with the hardware. However, they did not include real-life examples of TinyML being used.

The people who did Study [14] looked at ways to make TinyML models work better like making the numbers smaller getting rid of extra parts and designing the neural networks to be more efficient. They found out that it is really important to make sure the models are accurate and do not use much energy or take too long when they are used on microcontrollers.

Next was Study [5] which examined how TinyML could be used in embedded vision systems in robots and devices capable of operating independently. They demonstrated that the small neutral networks are capable of achieving things, such as identifying pictures on devices which are not very strong. They simply stared at the vision usage of TinyML and did not look at other applications o TinyML. All of these studies combined demonstrate that TinyML can be quite helpful to turn the embedded systems into smarter ones. They also demonstrate the fact that not everything is alright, such as hardware is not good enough, there is not a sufficient amount of tests conducted and no standardized methods of model creation of TinyML.

All these issues imply that TinyML should be further researched. Limitations of TinyML such as hardware constraints and lack of benchmarking research remain still to be solved. There is much potential in TinyML. It requires improvement to be better.

III. METHODOLOGY

The review paper uses a literature review method to examine existing research about TinyML applications in embedded systems. The study gathered and assessed research articles from open access journals and conference papers and technical reports which dealt with TinyML and embedded machine learning and edge AI systems. Researchers executed the literature search through

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

academic databases which included ScienceDirect and MDPI and Atlantis Press and arXiv.

The search used keywords which included "TinyML" and "embedded machine learning" and "edge AI" and "microcontroller machine learning" and "TinyML IoT applications". The research team picked only those studies which examined TinyML applications and TinyML architectures and TinyML frameworks and TinyML optimization techniques. The research team used different criteria to analyze the selected works which comprised to research objectives and methodology and key contributions and limitations of the study.

The research team summarized the papers that they classified into groups to identify general research trends and technological advances and gaps in research in TinyML implementation on embedded systems. The approach provides a structured overview of the literature as it also demonstrates the opportunities that can be explored in the future regarding TinyML technology.

IV. COMPARATIVE ANALYSIS

This section a qualitative comparison of popular TinyML frameworks, platforms and optimization methods used in embedded devices. The analysis assesses key performance metrics, including memory footprint, processing efficiency, energy efficiency, development ease. scalability, deployment time, hardware support, and support for model complexity. These criteria are critical in TinyML as embedded system have tight resource constraints of memory, processing capabilities and are battery-powered [6] [8].

In contrast to conventional machine learning systems, which are largely dependent on cloud technologies, TinyML frameworks are designed to facilitate on-device learning, thereby avoiding latency issues, enhancing privacy, and reducing data transfer. But different frameworks or techniques have varying trade-offs between performance, flexibility and efficiency, so it is important to compare them to determine the best fit for different use cases.

TABLE I. Qualitative Comparison of TinyML Frameworks and Approaches

Criteria	TensorFlow Lite Micro	Edge Impulse	Arduino TinyML	PyTorch Mobile	Optimized Models (Quantized/ Pruned)
Memory Usage	Very Low	Low	Very Low	Medium	Very Low
Power Efficiency	High	High	High	Medium	Very High
Ease of Use	Moderate	Very Easy	Easy	Moderate	Difficult
Model Complexity Support	Low–Moderate	Moderate	Low	High	Moderate–High
Deployment Speed	Moderate	Fast	Fast	Moderate	Slow
Hardware Compatibility	High	High	Moderate	Moderate	High
Scalability	Moderate	High	Low	High	High
Development Flexibility	Moderate	Low	Low	High	Very High
Best Use Case	Embedded systems	Rapid development	Simple IoT tasks	Complex edge AI	Performance optimization

Fig. 1. *Table I is a qualitative overview of TinyML technologies according to their characteristics. We use qualitative indicators (e.g., Low, Medium, High) rather than quantified values to capture the results from previous studies. This enables us to compare how each approach fares compared to others in resource-limited environments.*

TensorFlow Lite Micro

TensorFlow Lite Micro is a lightweight framework for inference of machine learning models on resource-constrained microcontrollers. It allows for on-device inference without internet connectivity, so it can be used in real-time inference tasks such as sensor data processing and anomaly detection [7].

Advantages

TensorFlow Lite Micro has a very small memory footprint, making it suitable for low-resource systems such as those based on microcontrollers. It allows for real-time inference with low latency, which is crucial for embedded systems that need to perform in real-time, without relying on cloud services. It's also well-supported in the TensorFlow ecosystem, offering developers access to numerous resources such as tools, documentation, and user communities.

Disadvantages

Reliable performance od TensorFlow Lite Micro is limited because it cannot support complex neutral network architectures, limiting the model users can use. The system demands models to be trained prior to deployment because it cannot train models on-device due to microcontroller hardware constraints. The system requires developers to pre-optimize the models because it requires model quantization.

Edge Impulse

Edge Impulse is an integrated TinyML development platform that streamlines the entire workflow from data acquisition to deployment. It is widely adopted in both academic and industrial environments due to its simplicity and efficiency.

Advantages

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Edge Impulse provides a complete end-to-end solution for TinyML development, covering everything from data collection to model deployment. The development process becomes more efficient and user-friendly because automation tools enable quicker project completion. The system supports multiple embedded hardware platforms which provide users with freedom to select devices that suit their specific needs.

Disadvantages

The Edge Impulse system needs cloud processing to train its models, which creates problems when internet access is limited and when users need to maintain complete data security. The platform restricts users from accessing fundamental model optimization techniques, which prevents advanced users from achieving their desired performance tuning results. The platform requires users to pay for certain advanced features, which increases their total development expenses.

Arduino TinyML

Arduino TinyML is an architecture that enables use of microcontroller by using machine learning models on Arduino Nano 33 BLE boards. The system is set to deal with the simple real-time functions which involve gesture detection and keyword identification [19].

Advantages

The Arduino TinyML hardware platforms offers developers and students an affordable hardware platforms that they can readily tap into their projects. The system allows developers to interface their projects with the sensors and the Internet of Things devices, and it is a key requirement of embedded systems development. The system provides a useful starting point to the beginners, and its user-friendly nature can be useful in small-scale projects.

Disadvantages

The system has a number of drawbacks such as the fact that it relies on cloud-based processing to train the system which can in turn reduce the offline capabilities and dependence on the internet connectivity. It also has less control over optimization and low-level models. And hence, is less user-friendly to users who require highly optimized performance. Moreover, more advanced features can be available only with subscription, which can be limited to the users with a tight budget.

PyTorch Mobile

The framework PyTorch Mobile allows users to run their PyTorch models on edge devices. PyTorch Mobile has more design flexibility than TinyML specific tools, and it supports complex technological designs.

Advantages

PyTorch Mobile's powerful model capabilities make it an excellent choice for applications that require advanced processing. This tool is used by researchers as it allows developers to build on existing resources to develop their own projects. This platform helps developers to build on existing resources to develop their own projects. This platform helps developers move their focus from model creation to deployment processes, thereby improving their productivity.

Disadvantages

PyTorch Mobile has poor edge device performance due to its requirement for extra memory and power, which restrains its performance. It does not have characteristics that support working with ultra-low-power microcontrollers and therefore does not work well in the high constrained environment. The system performs poorly when operating in an environment with resource restrictions.

Optimized Models (Quantization/Pruning)

TinyML systems depend on model optimization methods which include quantization and pruning and knowledge distillation. The methods decrease model size and processing needs while they sustain sufficient accuracy performance [9] [11] [14].

Advantages

The use of optimized models results in lower memory requirements and reduced power usage which proves essential for running machine learning on devices with limited resources. The solution enables users to operate advanced models on devices with restricted computing power because it decreases resource demands. The techniques boost system performance and efficiency, which makes embedded AI applications more usable in real-world scenarios.

Disadvantages

Advanced knowledge is necessary for beginners to implement optimization techniques, which creates challenges for them. The system introduces accuracy trade-offs which can reduce model performance. The process requires extra time to develop because developers must validate and fine-tune the optimized models in addition to their regular work.

V. DISCUSSION

The result of this research underline the increasing relevance of TinyML in enabling smart embedded systems and edge computing. TinyML enables machine learning models to be run directly on microcontrollers and other low-power devices, minimising the need for cloud connectivity. This not only helps reduce latency, increase privacy, decrease network bandwidth and enable real-time decision making, all of which are essential in today's Internet of Things (IoT) systems [6], [8], [17].

The comparative study also reveals that different TinyML platforms and techniques offer different performance levels based on system requirements. For example, TensorFlow Lite Micro is well suited for low-resource environments, providing fast on-device inference and low memory footprint. On the other hand, other

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

platforms such as Edge Impulse streamline the development of TinyML and faster deployment. But the research shows some trade-offs. Easier-to-use frameworks like Arduino TinyML work well for simple tasks but lack the computational power needed for more advanced applications. Meanwhile, more powerful platforms, such as PyTorch Mobile, support more complex models but require more computing power, which may not be ideal for very-low-power devices. These insights suggest that there is no universal answer to TinyML, and the framework selection should be tailored to the application's requirements and constraints.

A further insight is that model optimization approaches, like quantization and pruning, play a vital role in facilitating TinyML applications. They help shrink models and their computational demands while preserving sufficient accuracy, enabling the deployment of machine learning models on resource-constrained devices [9], [11], [14]. Without these techniques, the use of many of today's machine learning models would be unfeasible for embedded devices.

While TinyMl has its benefits, it also has its drawbacks. Hardware constraints, interoperability and security are all major challenges. What's more, most existing applications are short-lived or prototype with little work done on the long-term feasibility and adaptability of such systems in real-world scenarios. This suggests a need for further research in topics including energy-efficient hardware design, development platforms, adaptive model learning and secure model deployment.

In conclusion, the findings shows that TinyML has the potential to revolutionize embedded systems by delivering efficient, smart and autonomous operation of devices. But to realize this potential, design trade-offs need to be carefully considered and ongoing improvements in hardware and software technologies are needed.

VI. Future Research

The future of TiyML will focus on improving embedded machine learning through development efforts that focus on hardware and software. One such avenue is the design of to hardware accelerators that are dedicated for execution of neutral networks on microcontrollers. These accelerators are able to improve computational efficiency while allowing devices with limited resources to run complex computational models [8]. A further direction is the creation of more efficient neutral network architectures that can be used in TinyML. Scientists are using lightweight models that are accurate and have low computational and memory requirements. The architectures will allow AI capabilities to be executed on embedded devices [9] [11]. The use of automatic model optimization tools will become common in many industries. The tools will automate the process of converting large models to an optimized TinyML format use by companies in their embedded systems. Automation makes development easier and allows more developers to work with TinyML technologies without having to know much about model compression techniques [13].

We will see more TinyML systems integrated with edge computing platforms in the future. The mixed architecture that performs local interferences as the main processing method and only communicates with the cloud when needed enables efficient and scalable development. The systems allow local processing for most data and cloud-based training and analytics when necessary.

VII. CONCLUSION

This research provided an overview of the state of TinyML in embedded system through a review of its use cases, limitations, and research opportunities, and a comparative analysis of popular TinyML frameworks and optimization techniques. This research shows that TinyML is an important steps in the convergence of machine learning and embedded systems, allowing for intelligent data processing at the edge.

The framework comparison revealed that various TinyML platforms (such as TensorFlow Lite Micro, Edge Impulse, Arduino TinyML, and PyTorch Mobile) have different strengths and weaknesses based on the system's application requirements. Compact frameworks are suitable for low-resource environments, whereas more versatile frameworks allow for the use of larger models, but at the expense of increased resource demands. Furthermore, model optimization methods like quantization and pruning are critical for enhancing performance and enabling execution on resource-constrained devices.

While TinyML holds great potential, there are still some challenges to be addressed, such as hardware limitations, lack of standardization, security, and limited long-term assessment in real-world scenarios. Overcoming these limitations is critical for the widespread deployment and scalability of TinyML systems.

To improve the effectiveness, efficiency, and security of TinyML systems, future research should look into development of efficient algorithms, hardware accelerators, standard frameworks, and life long learning that can adapt to the changes in real-world deployment of TinyML systems. Additionally, the combination of TinyML with edge and cloud computing in hybrid models in a potential avenue for efficiency and scalability.

Finally, TinyML could play a key role in the development of embedded systems and loT. Through its ability to perform smart processing on resource-limited devices, it facilitates the creation of smarter, faster, and more efficient systems in multiple areas, such as health care, smart cities, environmental sensing, and automation.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

REFERENCES

[1] M. Pavan, A. Meloni, L. Capogrosso, F. Fummi and M. Cristani, "TinySV: Speaker Verification with On-Device Learning," *IEEE Internet of Things Journal*, pp. 1-12, 2024.

[2] L. Capogrosso, D. S. Cheng, F. Cunico, F. Fummi and M. Cristani, "A Machine Learning-Oriented Survey on TinyML," arXiv Research Repository, Ithaca, New York, 2023.

[3] A. Patel and F. Al-Fayez, "TinyML on the Edge: Model Compression and Energy-Latency Trade-offs.," *Future Internet*, p. 1–20, 2025.

[4] Y. Abadade, A. Temouden, H. Bamoumen, N. Benamar, Y. Chtouki and A. S. Hafid, "A Comprehensive Survey on TinyML," *IEEE Acces*, pp. 96892 - 96922, 2023.

[5] R. Wainbuch, "TinyML: Deploying Machine Learning on Microcontrollers for IoT Applications," *Journal of Science, Technology and Engineering Research*, pp. 1-10, 2024.

[6] P. Ray, "A Review on TinyML: State-of- the-Art and Prospects," *Journal of King Saud University-Computer and Information Sciences*, pp. 1595-1623, 2022.

[7] R. David, J. Duke, A. Jain, V. J. Reddi, N. Jeffries, J. Li and P. Warden, "TensorFlow Lite Micro: Embedded Machine Learning on TinyML Systems," in *Proceedings of machine Learning Systems(MLSys)*, San Jose, California, 2021.

[8] V. Viswanatha, A. C. Ramachandra, R. Prasanna, P. C. Kakarla, V. Simha and N. Mohan, "Gesture and Speech Recognition Using Arduino Nano 33 BLE with TinyML," in *International Conference on Embedded systems and Applications*, Las Vegas, Nevada, 2022.

[9] A. Barovic and A. Moin, "TinyML for Speech Recognition Using Quantized Convolutional Neural Networks," *Sensors*, pp. 1-18, 2025.

[10] M. Beltrán-Escobar, T. E. Alarcón, J. Y. Rumbo-Morales, S. López, G. Ortiz-Torres and F. D. J. Sorcia-Vázquez, "A Review on Resource-Constrained Embedded Vision Systems-Based Tiny Machine Learning for Robotic Applications," *Sensors*, pp. 1-24, 2024.

[11] C. Banbury, V. J. Reddi, P. Warden and e. al., "Benchmarking TinyML Systems: Challenges and Directions," Massachusetts Institute of Technology, Cambridge, USA, 2021.

[12] M. A. Al-Mamun, M. A. Hossain and e. al., "Tiny Machine Learning and On-Device Inference: A Survey," *Sensors*, pp. 1-25, 2025.

[13] A. Wibisono, A. S. Nugroho and M. S. Hidayat, "A Systematic Literature Review of TinyML," in *Atlantis Press Conference Proceedings*, Paris, France, 2023.

[14] P. Warden and D. Situnayake, TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers, Sebastopol, California: O'Reilly Media, 2019.

[15] H. Nguyen, S. Kim and J. Lee, "Emerging Trends and Strategic Opportunities in TinyML," *Neurocomputing*, pp. 1-15, 2025.

[16] J. o. S. R. E. Team, "TinyML IoT Microcontrollers," *Journal of STE Research*, pp. 1-9, 2024.

[17] S. R. N. Reddy and A. Kumar, "TinyML Meets IoT: A Survey," *IEEE Internet of Things Journa*, pp. 1-19, 2023.

[18] H. M. Fahim and M. Chowdhury, "TinyML IoT Large-Scale Review," *Future Internet*, pp. 363-381, 2023.

[19] M. K. Hasan and A. Rahman, "Vision TinyML Systems," *Algorithms*, pp. 476-495, 2023.

[20] J. Park and Y. Kim, "Practical TinyML Implementation: Model Training and Deploymen," *Journal of Machine Learning Systems Research*, p. 1–12, 2023.

[21] A. Mishra and P. Panda, "Quantization and Efficient Neural Networks for TinyML," arXiv Research Repository, Ithaca, New York, 2023.

[22] L. Capogrosso, F. Fummi and M. Cristani, "ML-Oriented TinyML Surveys," arXiv Research Repository, Ithaca, New York, 2023.

[23] S. Han, J. Pool, J. Tran and Dally, "Efficient Neutral Networks for TinyML," arXiv Research Repository, Ithaca, New York, 2023.