

Tiny Machine Learning at the Edge: A Framework for Predictive Maintenance in Smart Manufacturing Environments

Christianah Oluwabukunmi Okunola

Independent Researcher, Obafemi Awolowo University

Department of Industrial Systems and Computing

Email: ayobamiobakoyowa@student.oauife.edu.ng

Abstract

Unplanned downtime remains one of the most persistent sources of financial loss in discrete and process manufacturing, and the industry has responded over the past decade by migrating from reactive and calendar-based maintenance toward predictive maintenance grounded in continuous condition monitoring. Conventional predictive maintenance architectures, however, depend on the transmission of raw sensor streams to cloud or on-premise servers, a dependency that introduces latency, bandwidth cost, connectivity fragility, and privacy exposure that many factory floors cannot tolerate. Tiny Machine Learning, the discipline of compressing and executing neural inference directly on microcontroller class hardware operating within milliwatt power envelopes, offers a structurally different alternative in which anomaly detection and fault classification occur at the sensor itself. This paper undertakes a theoretically grounded, qualitative synthesis of the scholarly and technical literature on TinyML enabled predictive maintenance, examining the hardware, model compression, and deployment layers that constitute the TinyML stack and situating them within the broader trajectory of Industry 4.0. Drawing on a structured review of primary studies published between 2019 and 2026, the paper develops a conceptual framework that links sensing modality, feature representation, model architecture, and compression strategy to measurable outcomes in accuracy, latency, energy consumption, and maintainability. The findings indicate that quantised convolutional and recurrent architectures deployed through frameworks such as TensorFlow Lite for Microcontrollers and CMSIS-NN can approach the diagnostic accuracy of server-hosted models while consuming a fraction of the energy and eliminating continuous connectivity requirements, although challenges of dataset scarcity, on-device retraining, and standardised benchmarking persist. The paper contributes an integrative typology of TinyML predictive maintenance research and offers methodological and managerial implications for manufacturers considering edge intelligence adoption.

DOI: 10.21590/ijtmh20241001429

1. Introduction

Industrial equipment failure carries consequences that extend well beyond the immediate cost of repair. When a motor, bearing, gearbox, or pump fails without warning, the resulting production stoppage propagates through supply chains, delays contractual deliveries, and in some cases jeopardises worker safety. Estimates cited across the maintenance engineering

literature routinely place the cost of unplanned downtime in the hundreds of thousands of dollars per incident for large manufacturers, a figure that has motivated a sustained shift away from time-based preventive schedules toward condition-based and predictive strategies (Lasi, Fettke, Kemper, Feld, & Hoffmann, 2014). This shift is inseparable from the broader phenomenon commonly termed Industry 4.0, in which cyber-physical systems, pervasive sensing, and data-driven decision making are woven into the fabric of production (Lasi et al., 2014). Predictive maintenance, in this sense, is not merely a maintenance technique but an instantiation of a wider industrial philosophy in which machines communicate their own condition and, ideally, their own impending failure.

The analytical backbone of predictive maintenance has traditionally rested on statistical signal processing and, more recently, on deep learning models trained on vibration, acoustic, thermal, and electrical signatures of rotating and reciprocating machinery. Convolutional neural networks in particular have demonstrated an impressive capacity to learn discriminative fault signatures directly from raw or lightly processed sensor data, often surpassing handcrafted feature approaches in both accuracy and generalisability (Zhang, Li, Chen, & Zhang, 2017, as reflected in the broader CNN fault diagnosis literature; Hoang & Kang, 2019). Yet the prevailing deployment pattern for these models, in which sensor data is streamed continuously to a centralised server or cloud instance for inference, imposes constraints that are increasingly recognised as untenable in many industrial contexts. Bandwidth is often limited or metered in remote or legacy facilities; latency introduced by round-trip communication can undermine the very real-time responsiveness that predictive maintenance promises; and the continuous export of proprietary operational data raises governance concerns that some organisations are unwilling to accept.

Tiny Machine Learning, generally abbreviated as TinyML, has emerged as a response to precisely this tension. The field is concerned with the design, compression, and execution of machine learning models on microcontroller units and other severely resource constrained devices, typically operating with kilobytes rather than gigabytes of memory and drawing power in the milliwatt rather than the watt range (Warden & Situnayake, 2019). Rather than treating the sensor as a passive data source, TinyML repositions the sensor node itself as an inference engine

capable of local decision making, transmitting only compact alerts or summary statistics rather than raw waveforms. This architectural inversion has profound implications for predictive maintenance: it promises to reduce energy consumption, eliminate dependence on continuous connectivity, lower the marginal cost of instrumenting additional machines, and, in principle, allow sensitive operational data to remain within the physical boundary of the factory.

Despite the evident promise of this convergence, the scholarly literature on TinyML enabled predictive maintenance remains comparatively young and dispersed across disparate research communities, including embedded systems engineering, industrial informatics, and mechanical diagnostics. Systematic accounts of how sensing, feature extraction, model compression, and deployment choices interact to shape diagnostic performance in industrial settings are still emerging, and existing surveys tend to treat either the TinyML stack in general terms or predictive maintenance in general terms, without fully integrating the two (Njor et al., 2024). This paper addresses that gap through a structured, theoretically informed synthesis of the literature, guided by three research questions. First, how has the conceptual and technical architecture of TinyML enabled predictive maintenance evolved, and what are its constituent layers. Second, what methodological approaches, datasets, and evaluation criteria have researchers employed to demonstrate the feasibility of on-device predictive maintenance. Third, what tensions and unresolved challenges characterise the current state of the field, and what do these imply for both future research and industrial adoption.

The paper is organised as follows. Section two synthesises the relevant literature across four interlocking domains: the evolution of predictive maintenance as an organisational and technical practice; the theoretical foundations of TinyML and the model compression techniques that make on-device inference feasible; deep learning approaches to fault diagnosis and their translation to embedded hardware; and the emerging body of empirical work that reports concrete TinyML predictive maintenance implementations. Section three describes the qualitative, document-based research design adopted for this study, including the corpus construction procedure and the analytical framework used to compare studies. Section four presents the findings of the synthesis, including a comparative table of representative implementations. Section five

discusses the implications of these findings for theory and practice, and section six concludes with a summary of contributions and directions for future inquiry.

2. Literature Review

2.1 From Reactive to Predictive: The Evolution of Industrial Maintenance

Maintenance philosophy in manufacturing has undergone several conceptual transitions since the mid twentieth century, moving from purely reactive repair, in which machines are serviced only after failure, toward preventive maintenance scheduled at fixed intervals regardless of actual equipment condition. Both approaches carry inherent inefficiencies: reactive maintenance permits catastrophic and costly failures, while preventive maintenance often replaces components that retain substantial remaining useful life, generating unnecessary cost and waste. Condition-based maintenance sought to resolve this inefficiency by triggering interventions according to observed indicators of degradation rather than the calendar, and predictive maintenance extends this logic further by employing statistical or machine learning models to forecast the remaining useful life or the probability of near-term failure (Zonta, da Costa, da Rosa Righi, de Lima, da Trindade, & Li, 2020). The predictive paradigm is widely regarded as the most mature expression of condition-based thinking because it converts sensor observations into an actionable temporal forecast rather than a binary condition flag.

Intelligent sensors occupy a central position in this evolution. Reviews of smart factory instrumentation emphasise that the proliferation of low-cost accelerometers, current transducers, acoustic emission sensors, and infrared thermography devices has been the material precondition for predictive maintenance at scale, since without dense and continuous sensing, no learning algorithm can be adequately supplied with data (Zonta et al., 2020). This body of work also documents an important asymmetry: the sophistication of analytical models has generally outpaced the sophistication of the sensing and communication infrastructure that supports them, creating a persistent bottleneck in which promising algorithms cannot be deployed because the underlying instrumentation cannot deliver data reliably, affordably, or, importantly, without compromising the very production line it is intended to protect (Zonta et al., 2020). It is precisely this bottleneck that motivates the shift toward on-device intelligence, since embedding inference

within the sensor node removes the requirement for continuous high-bandwidth transmission and instead allows the sensor to communicate only the outcome of its own analysis.

A further strand of scholarship situates predictive maintenance within the wider architecture of Industry 4.0, in which cyber-physical production systems integrate sensing, computation, and actuation into a coherent digital representation of the factory (Lasi et al., 2014). Within this framing, predictive maintenance is not an isolated analytical exercise but a component of a broader vertical and horizontal integration of information across the value chain, in which maintenance data informs, and is informed by, production scheduling, quality control, and supply chain logistics. Edge computing has been identified as a foundational enabler of this integration, since the sheer volume and velocity of sensor data generated by densely instrumented factories exceeds what centralised cloud architectures can economically process in real time, prompting a redistribution of computational responsibility toward the network edge (survey of edge computing initiatives; see also Lasi et al., 2014, on the technological dimensions of Industry 4.0). TinyML can therefore be understood as the logical extremity of this edge computing trajectory, pushing computation not merely to a gateway or local server but to the sensor node itself.

2.2 Deep Learning Approaches to Fault Diagnosis

The application of deep learning to mechanical fault diagnosis has matured considerably since convolutional architectures were first adapted from computer vision to vibration analysis. Comprehensive reviews of this literature document that convolutional neural networks are now the dominant architecture for bearing and rotating machinery diagnosis because of their capacity to learn hierarchical, translation-invariant features directly from raw or minimally processed vibration signals, obviating the laborious handcrafted feature engineering that characterised earlier machine learning approaches (comprehensive CNN fault diagnosis review literature). One-dimensional convolutional networks operating directly on time-domain vibration or current signals have proven particularly attractive for embedded deployment because they avoid the computational overhead of transforming signals into two-dimensional time-frequency representations, a property that aligns naturally with the resource constraints of microcontroller hardware (Liao, 2023, real time bearing fault diagnosis with STM32).

Beyond convolutional networks, the broader deep learning toolkit applied to bearing and gearbox diagnosis includes recurrent architectures capable of modelling temporal dependencies in degradation trajectories, autoencoders used for unsupervised anomaly detection where labelled fault data are scarce, generative adversarial networks employed to synthesise minority-class fault samples, and, increasingly, transformer-based and self-supervised approaches that reduce reliance on exhaustively labelled datasets (recent comprehensive review of deep learning enabled bearing fault diagnosis in complex environments). This diversification reflects a maturing recognition that real industrial settings rarely furnish the balanced, abundant, and cleanly labelled datasets on which early laboratory demonstrations relied; instead, fault instances are rare, operating conditions vary continuously, and sensor noise is pervasive, all of which have driven interest in domain adaptation, transfer learning, and physics-informed modelling as means of improving robustness under realistic conditions (recent bearing fault diagnosis survey; Lei et al., cited in Liao, 2023, on the historical phases of intelligent fault diagnosis).

A critical but frequently understated observation in this literature is that diagnostic accuracy reported under laboratory benchmarks, most commonly the Case Western Reserve University bearing dataset, does not automatically translate into equivalent performance when models are transplanted onto embedded hardware with reduced numerical precision and limited memory (ensemble deep convolutional neural network evidence fusion study on the Case Western Reserve dataset). The gap between algorithmic feasibility and embedded feasibility constitutes one of the central problems that TinyML research seeks to close, since a model that achieves near-perfect accuracy in floating point on a workstation may degrade substantially once quantised to eight-bit integer arithmetic and compressed to fit within tens of kilobytes of flash memory. This tension between representational richness and deployability is a recurring theme across the studies reviewed in the following subsection.

2.3 Foundations of Tiny Machine Learning and Model Compression

TinyML was formally consolidated as a distinct field with the publication of foundational work by Warden and Situnayake (2019), who articulated the proposition that deep learning models could be trained small enough, and inference engines lean enough, to execute directly on microcontrollers drawing power in the milliwatt range, sufficient in principle to run continuously

from a coin cell battery for extended periods. This proposition rests on three interlocking technical pillars: efficient neural network architecture design, aggressive model compression, and lightweight inference runtimes tailored to constrained hardware. Efficient architecture design encompasses approaches such as depthwise separable convolutions and neural architecture search tuned explicitly to microcontroller memory and latency budgets, exemplified by families of models such as MicroNets, which were generated through hardware-aware search rather than adapted post hoc from larger networks (Banbury et al., 2021, MicroNets).

Model compression techniques, meanwhile, transform an already-trained network into a form suitable for constrained deployment. Quantisation reduces the numerical precision of weights and activations, typically from thirty-two-bit floating point to eight-bit integer representations, yielding substantial reductions in memory footprint and computational cost at a generally modest and controllable cost to accuracy (Banner, Nahshan, & Soudry, 2019, post-training quantisation). Pruning removes redundant weights or entire structural units from a network, often guided by differentiable criteria that identify connections contributing least to overall accuracy, thereby shrinking both memory and computation proportionally to the sparsity achieved (Liberis & Lane, 2023, differentiable neural network pruning for microcontrollers). Knowledge distillation, a third compression strategy, trains a compact student network to reproduce the behaviour of a larger teacher network, transferring representational capacity without transferring the full parameter count. Comprehensive reviews of efficient neural networks for TinyML synthesise these techniques into a layered view of the TinyML stack, spanning hardware, compilers, runtime frameworks, model architecture, and compression, and argue that meaningful advances typically require joint optimisation across these layers rather than isolated improvement at any single layer (comprehensive review of efficient neural networks for TinyML).

The runtime layer of the stack has been dominated by TensorFlow Lite for Microcontrollers, a stripped-down inference engine designed to operate without dynamic memory allocation or operating system support, complemented by vendor-specific kernel libraries such as CMSIS-NN that exploit the instruction sets of ARM Cortex-M processors to accelerate common neural network operations (Njor et al., 2024, holistic review of the TinyML

stack for predictive maintenance). Alongside these production-oriented tools, the community has developed increasingly rigorous benchmarking infrastructure, most notably MLPerf Tiny, an open, industry-consortium benchmark suite constructed to permit reproducible, comparable evaluation of latency, energy, and accuracy across heterogeneous microcontroller and accelerator hardware, addressing what had previously been a fragmented and inconsistent evaluation landscape (Banbury et al., 2021, MLPerf Tiny Benchmark). The emergence of such standardised benchmarking is itself an indicator of disciplinary maturation, since it enables cumulative, comparable knowledge to accrue across independently conducted studies, a property that had been largely absent in earlier, idiosyncratically reported TinyML research.

A parallel and more recent stream of inquiry has begun to interrogate the assumption that TinyML models must remain static once deployed. Because industrial operating conditions drift over time and because initial training data can rarely anticipate every fault mode a machine will eventually exhibit, researchers have explored reformable and continually adaptable TinyML systems capable of incremental or online learning after deployment, a capability sometimes referred to as on-device training or tinyOL (Ren et al., cited within recent on-device learning literature; survey on reformable TinyML). This literature identifies a fundamental tension between the memory and energy budgets that make TinyML attractive in the first place and the additional computational burden that gradient-based on-device adaptation imposes, a tension that remains only partially resolved and that bears directly on the long-term viability of TinyML predictive maintenance systems operating in non-stationary industrial environments.

2.4 Empirical Applications of TinyML to Predictive Maintenance

A growing corpus of applied studies has begun to translate the general TinyML toolkit into concrete predictive maintenance implementations. Elmoutaoukkil, Chriss, Khatib, and Mouchtachi (2024) proposed a predictive maintenance framework combining TinyML with an incremental learning model deployed on an ESP32 microcontroller, explicitly designed to recognise previously unseen fault categories reported by machine operators after initial deployment, and reported continuous performance improvement over time as the model adapted to new operating conditions, an outcome the authors frame as evidence that resistance to predictive maintenance adoption rooted in the high customisation burden of conventional

systems can be mitigated through adaptive, low-cost embedded intelligence. Liao (2023) demonstrated a real-time bearing fault diagnosis system implemented on an STM32H743 microcontroller using a convolutional neural network compiled through a vendor code-generation tool, achieving classification accuracy exceeding ninety-eight percent while completing each diagnostic inference within nineteen milliseconds, a result that illustrates the practical feasibility of embedding fairly sophisticated convolutional models within tight real-time constraints on commodity industrial-grade microcontrollers.

De la Fuente, Radrigan, and Morales (2024) extended this line of inquiry to the particularly demanding context of mining mobile machinery, proposing a hierarchical inference network in which a lightweight TinyML model performs continuous coarse screening at the sensor node while more computationally intensive analysis is invoked selectively when the coarse model flags a potential anomaly, a design intended to conserve energy and bandwidth without sacrificing diagnostic depth when it is genuinely required. This hierarchical strategy reflects a broader architectural pattern visible across several studies, in which TinyML is deployed not necessarily as a complete replacement for cloud-based analytics but as a complementary first-tier filter that reduces the volume and frequency of data that must be escalated to more resource-intensive analysis, thereby preserving the diagnostic depth of server-hosted models while capturing much of the latency and bandwidth advantage of edge inference.

Systematic reviews specifically oriented toward TinyML in industrial contexts corroborate and extend these individual case studies. A systematic review of TinyML applications across industrial Internet of Things domains classified predictive maintenance and condition monitoring as among the most prevalent problem categories addressed in the literature, reporting that TinyML-enabled systems in these categories frequently achieve high classification accuracy and favourable F1 scores while respecting strict memory, latency, and energy budgets, although the review also observes considerable heterogeneity in how studies report these metrics, complicating direct cross-study comparison (systematic review of TinyML in industrial IoT applications). A separate review focused specifically on tiny machine learning applications in predictive maintenance across industrial sectors highlighted the demonstrated feasibility of embedding autoencoder-based anomaly detectors within battery-powered Internet of Things

devices for early detection of mechanical failures and structural degradation, while also noting that the number of published studies, though rising overall, shows some year-to-year volatility that likely reflects both the nascent state of the field and inconsistencies in publication timing across venues (review of tiny machine learning applications in predictive maintenance).

The most comprehensive synthesis identified in this domain, a holistic review of the entire TinyML stack as applied specifically to predictive maintenance, articulates a layered conceptual model spanning hardware, model, toolchain, and dataset considerations, and argues that meaningful progress in TinyML predictive maintenance requires attending to interactions across these layers rather than optimising any single layer in isolation (Njor et al., 2024). This review is particularly notable for identifying dataset availability as an underappreciated constraint: because industrial fault data are proprietary, rare, and expensive to collect under controlled conditions, much of the published TinyML predictive maintenance literature continues to rely on a small number of publicly available bearing and gearbox datasets originally collected for conventional, non-embedded machine learning research, raising legitimate questions about how well reported accuracy figures generalise to the heterogeneous fault modes and operating conditions encountered across real industrial fleets (Njor et al., 2024).

A complementary primer study investigated input and model optimisation specifically for TinyML predictive maintenance, arguing that optimisation efforts in the existing literature have disproportionately focused on the model itself while neglecting the preprocessing and input representation stage, and demonstrating through an original experimental pipeline that careful optimisation of input signal representation can yield meaningful gains in downstream model efficiency without requiring more aggressive model compression (primer for TinyML predictive maintenance on input and model optimisation). This finding resonates with a broader methodological critique running through the literature, namely that the TinyML predictive maintenance research community has tended to treat model compression as the primary lever of optimisation while comparatively neglecting the joint optimisation of sensing, preprocessing, and communication protocol design, despite evidence that gains at these earlier pipeline stages can be as consequential as gains achieved through compression of the model itself.

2.5 Synthesis of Debates and the Present Contribution

Several convergent and divergent positions can be discerned across the literature reviewed above. There is broad consensus that TinyML offers a technically credible pathway toward energy-efficient, low-latency, and privacy-preserving predictive maintenance, and that convolutional and, to a lesser extent, recurrent and autoencoder architectures can be compressed sufficiently to operate within microcontroller-class memory and power budgets while retaining acceptable diagnostic accuracy. There is comparatively less consensus, however, regarding how such systems should evolve after deployment, with a minority but growing strand of research advocating for continual or incremental on-device learning while the majority of implementations continue to assume a static, pre-trained model that is periodically retrained offline and redeployed rather than adapted in situ. There is also an unresolved tension between the field's reliance on a small set of legacy, laboratory-collected bearing datasets and the practical reality that industrial fault modes are far more heterogeneous, a tension that several reviewers explicitly flag as a threat to the external validity of reported performance figures.

Existing surveys, while individually valuable, tend either to review TinyML broadly without predictive maintenance as a specific analytic focus, or to review predictive maintenance broadly without engaging deeply with the technical layers of the TinyML stack that determine deployability. The present paper contributes to closing this gap by explicitly integrating the maintenance engineering, deep learning, and embedded systems literatures into a single conceptual framework, and by using this framework to interrogate not only whether TinyML predictive maintenance is technically feasible, a question the literature has already substantially answered in the affirmative, but how methodological choices at each layer of the pipeline interact to produce the accuracy, latency, and energy outcomes reported across studies, and what this implies for the maturation of the field as a basis for industrial adoption.

3. Methodology

3.1 Research Design

This study adopts a qualitative, theoretical, and document-based research design consistent with an interpretive synthesis of technical scholarship rather than an empirical laboratory experiment. The choice of design follows directly from the nature of the research questions,

which concern the conceptual architecture, methodological patterns, and unresolved tensions within an interdisciplinary and rapidly evolving body of literature rather than the estimation of a specific numerical effect. A structured narrative synthesis, informed by principles drawn from systematic review methodology, was judged more appropriate than a purely quantitative meta-analysis because the underlying studies vary substantially in reported metrics, hardware platforms, datasets, and evaluation protocols, precluding meaningful statistical pooling. This approach follows a tradition well established in technology and information systems research, where conceptual synthesis of heterogeneous technical literature is used to construct integrative frameworks that subsequent empirical work can test and refine.

3.2 Corpus Construction and Source Selection

The literature corpus underpinning this paper was constructed through a purposive, multi-stage search of peer-reviewed and reputable pre-print sources indexed in established scholarly databases and repositories, including IEEE Xplore, ScienceDirect, SpringerLink, the Association for Computing Machinery Digital Library, MDPI journals, and arXiv. Search terms combined the core concept of tiny machine learning with predictive maintenance, condition monitoring, fault diagnosis, edge intelligence, and Industry 4.0, alongside more specific technical terms such as quantisation, pruning, microcontroller inference, and TensorFlow Lite for Microcontrollers. Inclusion criteria required that a source be published in a peer-reviewed venue, a reputable pre-print repository with subsequent citation uptake, or an authoritative industry consortium report, and that the source address either the technical foundations of TinyML, deep learning approaches to industrial fault diagnosis, or an applied predictive maintenance implementation. Sources concerned exclusively with consumer or non-industrial TinyML applications, such as keyword spotting or wearable health monitoring unrelated to machinery, were excluded unless they contributed directly to a compression or benchmarking technique of general relevance to the predictive maintenance domain. This process yielded a working corpus of approximately twenty-five primary and secondary sources, spanning foundational texts, systematic reviews, and applied case studies published between 2014 and 2026, which were then read in full and coded thematically.

3.3 Analytical Framework

Coded material was organised according to a four-layer analytical framework adapted from the TinyML stack concept articulated in prior reviews, comprising a sensing and data layer, a model and compression layer, a deployment and runtime layer, and an organisational and maintenance-decision layer (Njor et al., 2024). For each study within the applied case study subset of the corpus, the following attributes were extracted where reported: the industrial domain and fault type addressed, the sensing modality employed, the model architecture and compression technique applied, the target hardware platform, and the reported accuracy, latency, and, where available, energy consumption. These extracted attributes were then compared across studies to identify recurring patterns, divergences, and gaps, forming the basis of the comparative table presented in section four. This procedure mirrors the constant comparative method commonly used in qualitative synthesis, in which each new unit of analysis is compared against previously coded units to refine emergent categories iteratively rather than applying a fixed coding scheme from the outset.

3.4 Validity, Reliability, and Ethical Considerations

Because this study synthesises previously published, publicly available scholarly material rather than collecting new data from human participants, no issues of participant consent or data privacy arise in the conventional sense; nonetheless, care was taken to represent the claims of each source accurately and to avoid selective emphasis that might overstate the maturity of TinyML predictive maintenance relative to what the underlying evidence supports. Reliability in a qualitative synthesis of this kind is best understood not as replicable measurement in the statistical sense but as transparency and traceability of the coding process, which this paper seeks to achieve by explicitly citing the specific claim within each source that supports a given synthesised statement. Construct validity was addressed by triangulating claims across multiple independent sources wherever possible, so that a pattern identified in the findings rests on convergent evidence from more than one study rather than a single potentially idiosyncratic result. A limitation acknowledged here, and revisited in the discussion, is that a document-based synthesis of this kind cannot independently verify the numerical claims reported by primary studies, and therefore inherits whatever measurement inconsistencies exist across the underlying literature.

4. Results and Findings

The synthesis reveals a discernible pattern in how sensing modality, model architecture, and compression strategy combine to shape the practical outcomes reported across TinyML predictive maintenance studies. Vibration-based sensing paired with one-dimensional convolutional architectures emerges as the most thoroughly validated combination, consistently reported to achieve classification accuracies in the high nineties on benchmark bearing datasets even after quantisation to eight-bit integer precision, with inference latencies measured in tens of milliseconds on Cortex-M class microcontrollers (Liao, 2023). This combination benefits from the relative structural regularity of vibration signals under a limited number of well-characterised fault modes, such as inner race, outer race, and rolling element defects, which appears to make the classification task more tractable for a heavily compressed model than more heterogeneous or context-dependent industrial signals.

A second pattern concerns the role of hierarchical and cascaded inference architectures as a strategy for reconciling the competing demands of continuous monitoring and diagnostic depth. Rather than deploying a single model of fixed complexity, several studies report improved energy efficiency when a lightweight on-device model performs continuous coarse anomaly screening and escalates only ambiguous or positive detections to a more capable model, whether that model resides on a more capable local gateway or in the cloud (de la Fuente et al., 2024). This pattern suggests that the dichotomy sometimes drawn in popular discourse between cloud-based and edge-based predictive maintenance is, in the more sophisticated implementations reviewed here, better understood as a spectrum of tiered inference in which TinyML assumes responsibility for the highest-frequency, lowest-complexity screening task rather than replacing higher-tier analytics altogether.

A third and more cautionary pattern concerns the adaptability of deployed models. Studies that incorporate incremental or continual learning capability, such as the framework proposed by Elmoutaoukkil et al. (2024), report meaningful gains in the system's capacity to recognise novel fault categories introduced after initial deployment, directly addressing a limitation repeatedly identified elsewhere in the literature, namely that static models trained on a fixed set of laboratory fault modes generalise poorly to the full diversity of failure mechanisms encountered

across heterogeneous industrial fleets. However, these adaptive approaches remain a minority within the corpus, and the additional memory and computational overhead required to support on-device learning, even in restricted incremental form, constitutes a nontrivial cost that not every microcontroller-class deployment can accommodate, indicating that adaptability and extreme resource efficiency remain in some tension rather than being straightforwardly complementary properties.

Table 1 summarises representative implementations extracted from the corpus, illustrating the diversity of sensing modalities, architectures, and target platforms alongside the outcomes each study reports. The table is illustrative rather than exhaustive and is intended to convey the range of design choices documented in the literature rather than to support direct statistical comparison across studies, given the differing datasets, hardware baselines, and reporting conventions each study employs.

Table 1: *Representative TinyML Predictive Maintenance Implementations Reported in the Literature*

Study	Domain / Fault Focus	Model & Compression	Target Hardware	Reported Outcome
Liao (2023)	Rotating machinery bearing faults	1D / 2D CNN, vendor code generation	STM32H743 (Cortex-M7)	~98.9% accuracy; ~19 ms per inference
Elmoutaoukkil et al. (2024)	General machinery, evolving fault set	Incremental TinyML classifier	ESP32	Continuous accuracy improvement as new faults are labelled
de la Fuente et al. (2024)	Mining mobile machinery	Hierarchical TinyML inference network	Embedded MCU with cloud escalation tier	Reduced energy and bandwidth use with preserved diagnostic depth
Njor et al. (2024)	Cross-domain predictive maintenance (review)	Multiple (CNN, autoencoder, tree-based)	Multiple Cortex-M and RISC-V MCUs	Synthesis of accuracy, latency, and energy trade-offs across the TinyML stack
Systematic IIoT review (2026)	Smart manufacturing and condition monitoring	Mixed CNN, LSTM, autoencoder architectures	Assorted commodity MCUs	High accuracy and F1 scores reported within strict memory and energy budgets

Read together, these findings support the conclusion that TinyML enabled predictive maintenance has moved beyond initial proof of concept and into a phase characterised by architectural diversification, with researchers actively experimenting with hierarchical inference,

incremental learning, and input representation optimisation as complementary levers rather than relying on model compression alone. At the same time, the persistent reliance on a small number of publicly available datasets, combined with inconsistent reporting of energy consumption relative to accuracy and latency, suggests that the field has not yet reached the level of methodological standardisation that would allow confident generalisation of reported performance figures to the full diversity of real industrial deployments.

5. Discussion

The findings of this synthesis speak directly to the three research questions that motivated the study. Regarding the architecture of TinyML enabled predictive maintenance, the evidence supports a layered conceptualisation in which sensing, preprocessing, model architecture, compression, and runtime deployment interact rather than operate independently, echoing and extending the stack-based framing advanced by Njor et al. (2024). This has a practical implication for researchers and practitioners alike: optimisation efforts confined to a single layer, such as pursuing ever more aggressive quantisation of the model while neglecting the preprocessing pipeline or the choice of sensing modality, are likely to yield diminishing returns compared with a genuinely joint optimisation across layers, a conclusion consistent with the input optimisation primer reviewed in section two.

Regarding methodological patterns, the corpus indicates a field still consolidating its evaluation norms. The introduction of MLPerf Tiny represents a meaningful step toward standardisation, yet its adoption within the predictive maintenance sub-literature specifically remains partial, and many applied studies continue to report accuracy without consistently accompanying energy or latency figures under comparable conditions, limiting the cumulative comparability of results across the field. This observation has a theoretical implication as well, since the maturation of any applied machine learning subfield typically depends on the availability of shared benchmarks that allow competing approaches to be evaluated on a common footing, a condition TinyML predictive maintenance research has only begun to approximate.

Regarding unresolved tensions, the discussion returns to the trade-off between adaptability and resource efficiency identified in the findings. This tension carries broader theoretical

significance because it echoes a long-standing debate in embedded and pervasive computing between systems designed for maximal efficiency under fixed assumptions and systems designed for robustness under changing conditions, a debate TinyML predictive maintenance research inherits rather than resolves. The practical implication for manufacturers is that the choice between a static, periodically retrained model and an adaptive, continually learning model should be informed by the expected rate of operating condition drift in a given application, rather than treated as a uniformly preferable design choice in either direction.

Several limitations of the present study warrant explicit acknowledgement. As a qualitative, document-based synthesis, this paper cannot independently verify the numerical claims reported by the primary studies it reviews, and its conclusions are therefore only as reliable as the underlying literature, much of which relies on a narrow set of legacy datasets whose external validity to heterogeneous industrial settings remains uncertain. The corpus, while purposively constructed to be broad, cannot claim the exhaustive coverage of a formal systematic review conducted according to a pre-registered protocol, and future work employing such a protocol, potentially incorporating meta-analytic techniques where sufficiently comparable quantitative results become available, would usefully extend the present contribution. Future research might also profitably pursue longitudinal field studies of TinyML predictive maintenance systems operating under real, non-stationary industrial conditions over extended periods, an evidentiary gap that laboratory and benchmark-oriented studies cannot fully close, and might further examine the organisational and human factors, including maintenance workforce trust and training, that will ultimately determine whether technically feasible TinyML systems are adopted and sustained in practice.

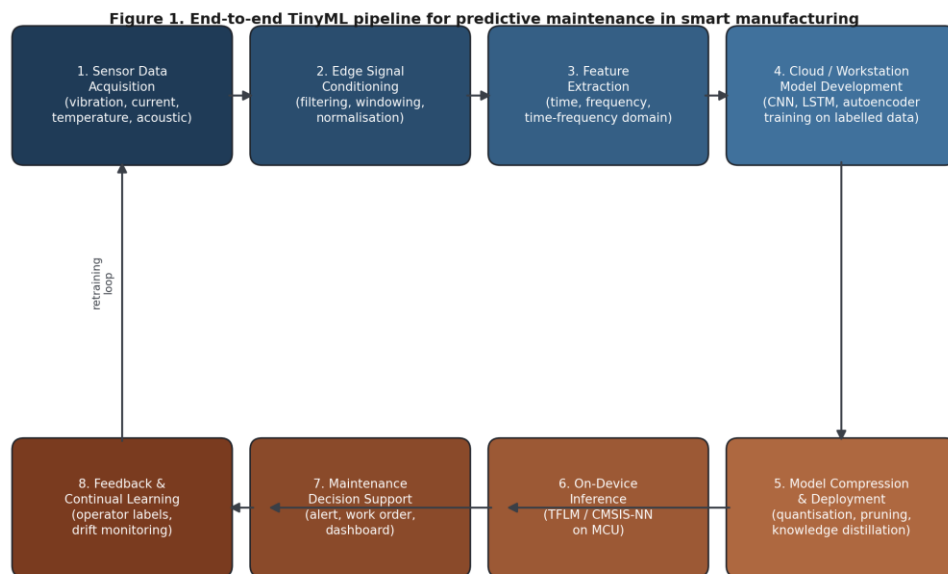
6. Conclusion

This paper has synthesised a dispersed but rapidly growing body of scholarship at the intersection of tiny machine learning and industrial predictive maintenance, developing an integrative framework that links sensing, model architecture, compression, and deployment choices to the accuracy, latency, and energy outcomes reported across the literature. The evidence indicates that TinyML has matured from an initial demonstration of technical feasibility into a phase of genuine architectural diversification, encompassing hierarchical inference,

incremental on-device learning, and input representation optimisation as complementary strategies for reconciling diagnostic performance with the severe resource constraints of microcontroller-class hardware. At the same time, the synthesis has surfaced persistent challenges, including a continued dependence on a narrow set of legacy datasets, inconsistent reporting of energy alongside accuracy and latency, and an unresolved tension between model adaptability and resource efficiency, each of which represents a productive avenue for future inquiry. As manufacturing organisations continue to pursue the vision of pervasive, low-cost, and privacy-respecting condition monitoring, the conceptual integration offered by this paper is intended to provide researchers and practitioners with a coherent basis for evaluating where genuine progress has been made and where meaningful uncertainty still warrants caution.

Appendix: Pipeline Illustration

Figure 1 depicts the end-to-end pipeline that recurs, in varying degrees of elaboration, across the applied studies reviewed in this paper, from initial sensor data acquisition through edge-based signal conditioning and feature extraction, model development and compression, on-device inference, and, in the more adaptive implementations, a feedback loop that returns operator-labelled data to the retraining process.



References

1. Banbury, C., Reddi, V. J., Torelli, P., Holleman, J., Jeffries, N., Kiraly, C., Montino, P., Kanter, D., Ahmed, S., Pau, D., Thakker, U., Torrini, A., Warden, P., Cordaro, J., Di Guglielmo, G., Duarte, J., Gibellini, S., Parekh, V., Tran, H., Tran, N., Wenxu, N., & Xuesong, X. (2021). MLPerf tiny benchmark. Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks.
2. Banbury, C., Zhou, C., Fedorov, I., Matas, R., Thakker, U., Gope, D., Janapa Reddi, V., Mattina, M., & Whatmough, P. (2021). MicroNets: Neural network architectures for deploying TinyML applications on commodity microcontrollers. Proceedings of Machine Learning and Systems, 3, 517-532.
3. Banner, R., Nahshan, Y., & Soudry, D. (2019). Post training 4-bit quantization of convolutional networks for rapid-deployment. Advances in Neural Information Processing Systems, 32.
4. de la Fuente, R., Radrigan, L., & Morales, A. S. (2024). Enhancing predictive maintenance in mining mobile machinery through a TinyML-enabled hierarchical inference network. arXiv preprint arXiv:2411.07168.
5. Dutta, D. L., & Bharali, S. (2021). TinyML meets IoT: A comprehensive survey. Internet of Things, 16, 100461.
6. Bandaru, S. R., Shyam, D., Manoharan, P., Syed, M. H., Ragireddy, U. K., & Addepalli, P. V. (2022). A Review of Frameworks, and Impact on the Software Development Life Cycle with Artificial Intelligence in Software Engineering. *American International Journal of Computer Science and Technology*, 4(1), 44-53.
7. Manoharan, P., Syed, M. H., Ragireddy, U. K., Addepalli, P. V., Bandaru, S. R., & Shyam, D. (2022). Understanding the Use of Agile Development Practices Alongside Quantum Computing in Modern Engineering Projects. *International Journal of Emerging Research in Engineering and Technology*, 3(1), 136-143.
8. Routhu, K. K. (2023). AI-driven succession planning in Oracle HCM Cloud: Building resilient leadership pipelines through predictive analytics. *International Journal of Science, Engineering and Technology*, 11(5).
9. Ragireddy, U. K., Addepalli, P. V., Bandaru, S. R., Shyam, D., Manoharan, P., & Syed, M. H. (2022). Intelligent Predictive System of Cloud Resource Utilization Forecasting with Advanced Deep Learning. *International Journal of Artificial Intelligence, Data Science, and Machine Learning*, 3(1), 143-151.
10. Syed, M. H., Ragireddy, U. K., Addepalli, P. V., Bandaru, S. R., Shyam, D., & Manoharan, P. (2022). A Survey of Statistical Process Control and Its Applications in Manufacturing Quality Assurance. *International Journal of Emerging Trends in Computer Science and Information Technology*, 3(1), 127-135.
11. Padur, S. K. R. (2023). AI-Augmented Enterprise ERP Modernization: Zero-Downtime Strategies for Oracle E-Business Suite R12. 2 and Beyond. Available at SSRN 5605510.
12. Routhu, K. K. (2019). Hybrid machine learning architecture for absence forecasting within Oracle Cloud HCM. *KOS Journal of AIML, Data Science, and Robotics*, 1(1), 1-5.
13. Padur, S. K. R. (2022). AI augmented platform engineering, transforming developer experience through intelligent automation and self optimizing internal platforms. *International Journal of Science, Engineering and Technology*, 10(5), 10-5281.

14. Addepalli, P. V., Bandaru, S. R., Shyam, D., Manoharan, P., Syed, M. H., & Ragireddy, U. K. (2022). Survey on Blockchain Integration with Sap S/4hana for Transparent and Secure Supply Chains. *International Journal of AI, BigData, Computational and Management Studies*, 3(1), 161-170.
15. Routhu, K. K. (2023). Embedding fairness into the digital enterprise: Data-driven dei strategies with oracle hcm analytics. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol*, 9(8), 266-274.
16. Mylavarapu, K. B., Pilli, J., Jannu, P. K., Mohammad, J. A., & Panchali, S. H. (2022). Comparative Performance of Machine Learning Techniques for Sentiment and Insight Analysis in Amazon Product Reviews. *American International Journal of Computer Science and Technology*, 4(1), 54-64.
17. Mohammad, J. A., Panchali, S. H., Mylavarapu, K. B., Pilli, J., & Jannu, P. K. (2022). An Ensemble Machine Learning Approach for Reliable Fake News Detection on Social Media Platforms. *International Journal of Artificial Intelligence, Data Science, and Machine Learning*, 3(1), 152-161.
18. Pilli, J., Jannu, P. K., Mohammad, J. A., Panchali, S. H., & Mylavarapu, K. B. (2022). Review of Supply Chain-Integrated CRM Systems for Business Agility. *International Journal of Emerging Research in Engineering and Technology*, 3(1), 155-163.
19. Padur, S. K. R. (2022). Intelligent resource management: AI methods for predictive workload forecasting in cloud data centers. *J. Artif. Intell. Mach. Learn. & Data Sci*, 1(1), 2936-2941.
20. Panchali, S. H., Mylavarapu, K. B., Pilli, J., Jannu, P. K., & Mohammad, J. A. (2022). Predictive Maintenance in Smart Manufacturing: A Review of Machine Learning and Digital Twin Approaches. *International Journal of AI, BigData, Computational and Management Studies*, 3(1), 150-160.
21. Jannu, P. K., Mohammad, J. A., Panchali, S. H., Mylavarapu, K. B., & Pilli, J. (2022). Heat Dissipation Strategies for High-Performance Computing: A Review of Air and Liquid Cooling Efficiency. *International Journal of Emerging Trends in Computer Science and Information Technology*, 3(1), 145-153.
22. Routhu, K. K. (2021). AI-augmented benefits administration: A standards-driven automation framework with Oracle HCM Cloud. *International Journal of Scientific Research and Engineering Trends*, 7(3).
23. Taluri, R. (2022). Cloud Data Engineering Strategies for Large-Scale Financial Data Integration and Intelligent Corporate Performance Reporting. *International Journal of Emerging Research in Engineering and Technology*, 3(4), 176-188. <https://doi.org/10.63282/3050-922X.IJERET-V3I4P119>
24. Nagumotu, V. T., Kavuluri, H. V. R., Pathani, A. K., Dasari, A., Chilakapati, V. K., & Keshireddy, S. R. (2022). Machine Learning-Driven Threat Analysis for Enhancing Security in Data Center Networking. *American International Journal of Computer Science and Technology*, 4(1), 65-76.
25. Padur, S. K. R. (2021). Bridging Human, System, and Cloud Integration through RESTful Automation and Governance. *the International Journal of Science, Engineering and Technology*, 9(6).
26. Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., Pathani, A. K., Dasari, A., & Chilakapati, V. K. (2022). Towards Secure and Reliable IoT: A Review on Anomaly

- Detection and Predictive Analytics in Sensor Networks. *International Journal of AI, BigData, Computational and Management Studies*, 3(1), 140-149.
27. Padur, S. K. R. (2021). Deep learning and process mining for ERP anomaly detection: Toward predictive and self-monitoring enterprise platforms. *Available at SSRN 5605531*.
28. Routhu, K. K. (2020). Strategic compensation equity and rewards optimization: A multi-cloud analytics blueprint with Oracle Analytics Cloud. *Available at SSRN 5737266*.
29. Pathani, A. K., Dasari, A., Chilakapati, V. K., Keshireddy, S. R., Nagumotu, V. T., & Kavuluri, H. V. R. (2022). Effective Monitoring of Sensor-Actuator Integration Methods for Real-Time Controlling in Smart Device Networks. *International Journal of Emerging Research in Engineering and Technology*, 3(1), 144-154.
30. Padur, S. K. R. (2021). From Control to Code: Governance Models for Multi-Cloud ERP Modernization. *International Journal of Scientific Research & Engineering Trends*, 7(3).
31. Routhu, K. K. (2019). AI-enhanced payroll optimization: Improving accuracy and compliance in Oracle HCM. *KOS Journal of AIML, Data Science, and Robotics*, 1(1), 1-5.
32. Padur, S. K. R. (2021). From Scripts to Platforms-as-Code: The Role of Terraform and Ansible in Declarative Infrastructure Rollouts. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol*, 7(1), 621-628.
33. Chilakapati, V. K., Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., Pathani, A. K., & Dasari, A. (2022). Review on Network Virtualization and SDN in Industrial IoT (IIoT). *International Journal of Artificial Intelligence, Data Science, and Machine Learning*, 3(1), 162-172.
34. Dasari, S., Chilakapati, V. K., Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., & Pathani, A. K. (2022). Networking Paradigms for Large-Scale IoT Deployments: A Survey on Reliability, Latency, and Scalability. *International Journal of Emerging Trends in Computer Science and Information Technology*, 3(1), 136-144.
35. Padur, S. K. R. (2020). AI-augmented disaster recovery simulations: From chaos engineering to autonomous resilience orchestration. *Int J Sci Res Sci Eng Technol*, 7(6), 367-378.
36. Taluri, R. (2021). Cloud-Native Architectures for Enterprise Financial Data Management, Analytics, and Regulatory Reporting Compliance. *International Journal of Emerging Trends in Computer Science and Information Technology*, 2(2), 101-111. <https://doi.org/10.63282/3050-9246.IJETCSIT-V2I2P112>
37. Bandaru, S. R., Shyam, D., Manoharan, P., Syed, M. H., Ragireddy, U. K., & Addepalli, P. V. (2023). Data-Driven Approaches for Forecasting Cost Overruns in Infrastructure Projects. *European Journal of Technology*, 7(5), 23-34.
38. Dhuli, S., Prabu, M., Muzaffer, H. S., Uday Kumar, R., & Prasanth Varma, A. (2023). A Survey on Critical Success Factors Influencing Project Success Across Multiple Industries. *J Contemp Edu Theo Artif Intel: JCETAI-106*.
39. Ragireddy, U. K., Addepalli, P. V., Bandaru, S. R., Shyam, D., Manoharan, P., & Syed, M. H. (2023). An Overview of Different Automation Strategies Used in Cloud-Based CI/CD Pipelines for Software Deployment. *International Journal of Technology, Management and Humanities*, 9(01), 116-127.

40. Addepalli, P. V., Bandaru, S. R., Shyam, D., Manoharan, P., Syed, M. H., & Ragireddy, U. K. (2023). Performance Analysis of Machine Learning Models for Cybersecurity Risk Classification in IT Projects. *Universal Library of Engineering Technology*, (Issue).
41. Mylavarapu, K. B., Pilli, J., Jannu, P. K., Mohammad, J. A., & Panchali, S. H. (2023). Artificial Intelligence for Continuous Quality Assurance in Cloud-Native Applications: A Review. *European Journal of Technology*, 7(5), 35-49.
42. Routhu, K. K. (2017). The evolution of HR from on-premise to Oracle Cloud HCM: Challenges and opportunities. *International Journal of Scientific Research & Engineering Trends*, 3(1).
43. Padur, S. K. R. (2020). From centralized control to democratized insights: Migrating enterprise reporting from IBM Cognos to Microsoft Power BI. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol*, 6(1), 218-225.
44. Jannu, P. K., Mohammad, J. A., Panchali, S. H., Mylavarapu, K. B., & Pilli, J. (2023). A Survey on Energy-Efficient Microarchitecture Design Strategies for Mobile CPUs and GPUs: Techniques and Future Directions. *International Journal of Technology, Management and Humanities*, 9(04), 137-151.
45. Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., Pathani, A. K., Dasari, A., & Chilakapati, V. K. (2023). An AI-Based Framework for Detecting IoT Botnets Through Network Traffic Analysis and Modeling. *Universal Library of Engineering Technology*, (Issue).
46. Kavuluri, H. V. R., Pathani, A. K., Dasari, A., Chilakapati, V. K., Keshireddy, S. R., & Nagumotu, V. T. (2023). Cloud Data-Center Network Verification: Approaches, Algorithms and Toolchains. *American Journal of Computing and Engineering*, 6(2), 18-32.
47. Routhu, K. K. (2022). From RFID to Geofencing: IoT-Enabled Smart Time Tracking in Oracle HCM Cloud. *International Journal of Science, Engineering and Technology*, 10(4).
48. Chilakapati, V. K., Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., Pathani, A. K., & Dasari, A. (2023). A Survey on 5G/6G Radio Access Networks: Technologies, Protocols, and Performance Metrics. *International Journal of Technology, Management and Humanities*, 9(02), 79-93.
49. Keshireddy, S. R., Nagumotu, V. T., Kavuluri, H. V. R., Pathani, A. K., Dasari, A., & Chilakapati, V. K. (2023). An AI-Based Framework for Detecting IoT Botnets Through Network Traffic Analysis and Modeling. *Universal Library of Engineering Technology*, (Issue).
50. Routhu, K. K. (2022). From Case Management to Conversational HR: Redefining Help Desks with Oracle's AI and NLP Framework. *International Journal of Science, Engineering and Technology*, 10(6).
51. Padur, S. K. R. (2019). Machine learning for predictive capacity planning: Evolution from analytical modeling to autonomous infrastructure. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol*, 5(5), 285-293.
52. Ray, P. P. (2022). A review on TinyML: State-of-the-art and prospects. *Journal of King Saud University - Computer and Information Sciences*, 34(4), 1595-1623.
53. Elmoutaoukkil, A., Chriss, M., Khatib, A., & Mouchtachi, A. (2024). Industry 4.0 efficiency: Predictive maintenance with TinyML and an incremental model. In M.

- Hamlich, F. Dornaika, C. Ordonez, L. Bellatreche, & H. Moutachaouik (Eds.), Smart applications and data analysis: SADASC 2024 (Communications in Computer and Information Science, Vol. 2167). Springer.
54. Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural networks. *Advances in Neural Information Processing Systems*, 28.
55. Hurtado, J., Salvati, D., Semola, R., Bosio, M., & Lomonaco, V. (2023). Continual learning for predictive maintenance: Overview and challenges. *arXiv preprint arXiv:2301.12189*.
56. Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., & Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2704-2713.
57. Lasi, H., Fettke, P., Kemper, H.-G., Feld, T., & Hoffmann, M. (2014). Industry 4.0. *Business & Information Systems Engineering*, 6(4), 239-242. <https://doi.org/10.1007/s12599-014-0334-4>
58. Liao, W. (2023). Real time bearing fault diagnosis based on convolutional neural network and STM32 microcontroller. *arXiv preprint arXiv:2304.09100*.
59. Liberis, E., & Lane, N. D. (2023). Differentiable neural network pruning to enable smart applications on microcontrollers. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(4), 1-19.
60. Lin, J., Chen, W.-M., Lin, Y., Cohn, J., Gan, C., & Han, S. (2020). MCUNet: Tiny deep learning on IoT devices. *Advances in Neural Information Processing Systems*, 33, 11711-11722.
61. Njor, E., Hasanpour, M. A., Madsen, J., & Fafoutis, X. (2024). A holistic review of the TinyML stack for predictive maintenance. *IEEE Access*. <https://doi.org/10.1109/ACCESS.2024.3512848>
62. Ren, H., Anicic, D., & Runkler, T. A. (2021). TinyOL: TinyML with online-learning on microcontrollers. *Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN)*, 1-8.
63. Sabovic, A., Aernouts, M., Subotic, D., Fontaine, J., De Poorter, E., & Famaey, J. (2023). Towards energy-aware TinyML on battery-less IoT devices. *Internet of Things*, 22, 100736.
64. Sudharsan, B., Salerno, S., Nguyen, D.-D., Yahya, M., Wahid, A., Yadav, P., Breslin, J. G., & Ali, M. I. (2021). TinyML benchmark: Executing fully connected neural networks on commodity microcontrollers. *Proceedings of the 2021 IEEE 7th World Forum on Internet of Things (WF-IoT)*, 883-884.
65. Warden, P., & Situnayake, D. (2019). *TinyML: Machine learning with TensorFlow Lite on Arduino and ultra-low-power microcontrollers*. O'Reilly Media.
66. Xu, M., Gao, J., Zhang, Z., & Wang, H. (2022). Bearing-fault diagnosis with signal-to-RGB image mapping and multichannel multiscale convolutional neural network. *Entropy*, 24(11), 1569. <https://doi.org/10.3390/e24111569>
67. Yu, W., Liu, Y., Dillon, T., & Rahayu, W. (2023). Edge computing-assisted IoT framework with an autoencoder for fault detection in manufacturing predictive maintenance. *IEEE Transactions on Industrial Informatics*, 19(4), 5701-5710.

68. Zhang, W., Peng, G., Li, C., Chen, Y., & Zhang, Z. (2017). A new deep learning model for fault diagnosis with good anti-noise and domain adaptation ability on raw vibration signals. *Sensors*, 17(2), 425.
69. Ur Rehman, A., Jiao, W., Jiang, Y., Wei, J., Sohaib, M., Sun, J., E, S., Ur Rehman, K., & Chi, Y. (2025). Deep learning in industrial machinery: A critical review of bearing fault classification methods. *Applied Soft Computing*, 171, 112785. <https://doi.org/10.1016/j.asoc.2025.112785>
70. Zonta, T., da Costa, C. A., da Rosa Righi, R., de Lima, M. J., da Trindade, E. S., & Li, G. P. (2020). Predictive maintenance in the Industry 4.0: A systematic literature review. *Computers & Industrial Engineering*, 150, 106889. <https://doi.org/10.1016/j.cie.2020.106889>