Journal of Intelligent Information and Communication

Research Paper

FastText-Based Framework for Real-Time Fault Log Classification in Industrial IoT: Model, Engineering, and Validation

Bartosz Sławomir Cyra[1, *]

[1] Faculty of Informatics and Computer Science, Kazimierz Wielki University in Bydgoszcz, Bydgoszcz, 85-064, Poland
*Corresponding author: bartosz.sc@ukw.edu.pl

Abstract. The volume and variety of failure logs produced by automated production systems have increased dramatically in recent years due to the Industrial Internet of Things' (IIoT) rapid development. In order to maintain regular operation and quickly address maintenance issues, the following real-time classification of these logs is required. In this paper, a stable framework based on FastText—which is comparatively fast and accurate—is introduced for the automatic classification of industrial defect records. For the framework test, a complete log dataset of 1.2 million entries from over 2,000 industrial equipment and 15 different types of operating problems was obtained. The dataset has been annotated by subject experts and preprocessed in a reasonably methodical manner. According to the experiment results above, the FastText-based classifier has a median inference latency of 3.1 ms per sample and a high accuracy of 91.8%, making it appropriate for high-speed deployment. The system has been found to be both resource-efficient and general-purpose based on comparisons with widely used statistical and neural models. Deployment in the industrial process has decreased the amount of unscheduled production downtime, decreased the frequency of false alarms, and cut the reaction time for maintenance. For large-scale industrial applications that require quick integration and real-time operation, lightweight neural networks are therefore very useful. The experiment mentioned above demonstrates how the two types of IIoT production environments will become more dependable and efficient.

Keywords: *Fault Log Classification, Industrial IoT, Predictive Maintenance, Real-Time Monitoring*

Received on 03 January 2025, Accepted on 09 April 2025, Published on 15 April 2025

Introduction

The Industrial Internet of Things (IIoT) has begun to transform manufacturing, power management, large-scale public facility construction, and the integration of industrial equipment and sensing devices with intelligent control and data analysis platforms [1,2]. The industry is shifting toward smart operation, condition-based maintenance, and end-to-end digitalization of industrial assets as a result of the aforementioned factors [3]. Thus, event and fault logs from IIoT devices are currently frequently utilized to monitor operations, resolve issues, and perform safety supervision [4]. According to the log analysis above, any issues that might arise in the industrial setting as a result of the downtime must be promptly identified in order to stop the issue from getting worse [5,6].

However, a new challenge for automated classification and anomaly detection has emerged due to the vast amount of extremely diverse, high-velocity fault reports that have been produced as IIoT implementation has grown in scale and diversity [7]. The structure, vocabulary, and quality of log entries vary widely; they are frequently short, domain-specific messages that suffer from noise, redundancy, and a significant class imbalance [8,9]. An alternative approach must be used because the current demands of large-scale and flexible systems are incompatible with the manual definition of rule-based systems [10,11]. Traditional machine learning techniques like Random Forests and Support Vector Machines have provided some automation, but as the log corpus grows in size and the need for fine-grained, real-time problem identification increases, their efficacy decreases [12,13]. Convolutional Neural Networks (CNNs), Long Short-Term Memory networks (LSTMs), and transformer-based models like BERT are relatively new developments in deep learning that have produced good results for general-purpose natural language processing, but their high computational requirements and requirement for large-scale, well-annotated datasets still limit their use in industrial applications [14,15].

This research suggests a high-efficiency and useful framework for real-time failure log categorization in IIoT contexts based on the FastText model in light of the aforementioned pressing needs. By combining quick training and inference with strong classification performance, we propose a novel approach to achieving the objective. In this study, create a domain-specific log dataset, preprocess IIoT features methodically, and perform comprehensive validation using the best baseline techniques. Our approach has balanced the three aspects of accuracy, scalability, and computing efficiency, and it can now be used as the basis for high-performance industrial analytics and predictive maintenance systems. Furthermore, this research offers comprehensive perspectives on the smooth integration of lightweight neural networks for scalable log analytics and offers useful advantages for both engineering and academia. This paper is structured as follows: Related work is introduced in Section II, the methodologies are described in Section III, experimental results and industrial applications are shown in Section IV, and the key conclusions and future directions are presented in Section V.

Related Works

Event and fault log analysis has been progressively included into industrial intelligence for large-scale anomaly identification, failure prediction, and operation analysis as Industrial Internet of Things (IIoT) devices have grown widely [16,17]. Automatic parsing and categorization are particularly challenging since IIoT logs differ from those in standard IT log systems in that they are more domain-specific, have a varied structure, and irregular shapes [18]. It might be difficult to extract features and do analysis on these logs since they frequently come from multiple kinds of devices and have varied formats and interpretations for the same event [19]. Many academics have been investigating approaches to develop high-performance log categorization systems specifically for industrial applications due to the need for timely and stable processing of these data streams in safety-critical and production-sensitive regions [20].

The initial attempts to automate IIoT log analysis were created and maintained by subject matter experts and relied solely on rule-based methods and regular expressions [21]. The aforementioned techniques performed badly when the system grew in size and the log structure changed, despite the fact that they might offer some preliminary answers for filtering and grouping repetitive log events [22]. In general, rule-based models have a relatively high maintenance cost and are more prone to fail in the face of unexpected event types because they are not flexible and must be often adjusted in a new operating environment or when semantic changes occur [23]. Researchers have begun investigating supervised machine learning techniques that can automatically learn discriminative features from labelled data and adjust to a wider range of log messages in order to address this shortcoming [24].

When processing categorical and statistical features produced from log data, supervised models such as Support Vector Machines (SVM), Naive Bayes (NB), and Random Forests (RF) have demonstrated higher performance than human rules [25]. Because of their theoretical soundness and relatively easy implementation, SVMs have been utilized extensively to identify binary and multi-class log faults [26]. However, their real-world applications in industrial settings have frequently been constrained by an excessively complex set of features and have demonstrated a certain sensitivity to feature selection techniques; this shortcoming is particularly noticeable in light of the variations in IIoT logs [27]. Despite being computationally straightforward, NB classifiers typically perform poorly on highly imbalanced or noisy data, which are typical in industrial settings, and have only demonstrated limited performance in scenarios with log event distributions and dependencies [28].

Deep learning has advanced somewhat over the last ten years, and there are currently numerous tools for log categorization. The ability of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks to automatically learn complicated temporal and semantic features from raw log sequences without the need for explicit feature engineering has made them popular in recent years [29,30]. LSTM architectures are more suited for modeling sequential dependencies in multi-line or context-rich logs, while CNN-based models have demonstrated good success in lowering log entry length by capturing local feature patterns. For large-scale labelled data, the aforementioned architectures have outperformed earlier classifiers in numerous scenarios, although they are not without issues. First, their high memory consumption and computational expense make them unsuitable for industrial applications with limited resources. Second, a lot of well-annotated training data is a prerequisite for both architectures, and industrial applications might not always have access to such data.

BERT is an example of a model that has been used in industry-level log analysis since the first collection of big pre-trained language models was released. These transformer-based architectures can be optimized for certain log classification tasks by using a lot of unlabeled text to develop rich language representations at a deep level. BERT-based models have demonstrated strong classification accuracy and robustness for noisy industrial data, according to recent benchmark studies. These improvements, however, come at the cost of increased inference latency and hardware needs, making them unsuitable for real-time IIoT applications that demand fast detection and a low-latency response. Before they can be used in an industrial setting, these also need sizable tagged datasets and intricate deployment procedures.

The FastText model has been chosen to tackle the issue of IIoT log classification due to the need for lightweight and dependable models. FastText uses shallow neural networks in conjunction with n-gram embeddings to learn syntactic and semantic patterns in text, and it is more resource-efficient and predictive than deep transformer models. It is more resilient to out-of-vocabulary items and other variances in industrial logs because it possesses the subword modeling capability. FastText may attain the same accuracy as more complex models at a far reduced cost in training and inference time, according to empirical results using industrial log data. It is also modular and can be added to and enlarged gradually, both of which are essential for industrial monitoring and maintenance.

Nevertheless, the present investigation has also found certain shortcomings in previous research. First, the heterogeneity, noise, and dynamism of real-world IIoT logs have not been taken into account in earlier research on machine learning and deep learning for log classification, which frequently used artificial or IT-focused datasets. Second, there aren't many thorough benchmarks available to compare FastText with the top deep-model-based models in industrial log contexts, and it's unclear how it might be used in conjunction with IIoT infrastructure. Lastly, few studies have offered helpful advice on how to implement and scale such techniques for live, real-time industrial analytics while taking latency, processing power, and data labeling capacity into account.

Methods

Dataset and Preprocessing

Good data must be gathered in order to create a high-quality fault log categorization model for an industrial Internet of Things (IIoT) system. More than 2,000 different industrial devices, including programmable logic controllers, sensors, and actuators, were included in the log data in this paper, which was obtained from a sizable smart manufacturing platform. A secure, edge-based IIoT data pipeline continuously gathers the log data, which covers all kinds of operating situations, from routine health checks to critical failure notifications [31].

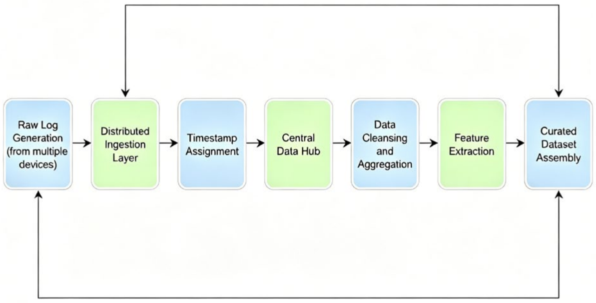

Figure 1. Data Acquisition and Preprocessing Workflow

A distributed ingestion layer is linked to the raw device stream, and the acquisition pipeline features low latency and low packet loss. After being time-stamped at the time of collection, all log messages were consolidated in a data hub for processing. The aforementioned technique will guarantee that the log's origin and the time it happened are both retained for further examination and troubleshooting. The entire process of creating raw logs, ingesting data, preprocessing, extracting features, and assembling curated datasets is depicted in Figure 1.

Following collection, the following preprocessing steps were implemented to address domain-specific characteristics, inherent noise, and inconsistencies in the industrial log data [32]. Deduplication was used first to eliminate duplicates caused by device retransmissions or monitoring overlaps. Next, non-diagnostic or invalid messages that were of little utility were filtered away. Tokenizing the log entries, normalizing numerals and system-specific codes, and anonymizing sensitive data in accordance with industrial privacy regulations were all accomplished using specialized regular expressions and straightforward pattern-based parsers.

Approximately 1.2 million log entries from 15 different fault categories make up the processed dataset. There is a class imbalance that is typical of industrial settings, with common fault types (like sensor read failures and communication errors) accounting for over half of the corpus and rare but severe anomalies appearing only infrequently. The majority of the log messages were quite brief (mean length: 14 words), according to statistical analysis, but they included a lot of technical acronyms, error codes, and proprietary phrases, making feature extraction more challenging.

Industrial field specialists annotated a stratified sample of raw logs in rounds using an iterative labelling technique to improve inter-annotator agreement in order to create a trustworthy ground truth for supervised learning. The final dataset labels are operationally useful fault categories related to predictive maintenance and prompt intervention, and ambiguous logs were chosen for consensus evaluation. Notably, our manual annotation method revealed that compound events—that is, multiple connected anomalies in a single message that require careful labeling and splitting—were frequent [33].

The data were split into training (70%), validation (15%), and test (15%) sets to make model training and evaluation easier. Each division's class distribution stayed the same. The text content was the main focus of feature engineering, although metadata like device kind and location were also incorporated. The reliability and generalizability of our experimental results in the realm of IIoT fault analysis are guaranteed by the whole-process curation and annotation approach.

FastText-Based Model

FastText is a high-performance text classification tool used in industry that combines the advantages of both previous and newer models to provide a very useful choice for good accuracy, speed, and usability. The primary reason FastText is favored is that it can handle the brief, fragmented, and jargon-heavy expressions of IIoT fault logs with little computing requirements since it leverages n-gram features and relatively compact word embeddings. The fundamental architecture and data flow of our FastText-based log classification system, from input preprocessing to probabilistic category output, are depicted in Figure 2.

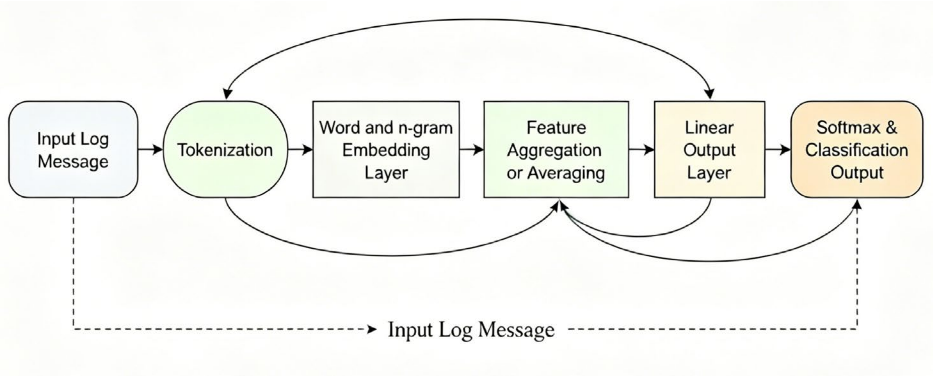

Figure 2. FastText Architecture for IIoT Log Classification

The core of FastText begins with mapping a given tokenized log message $\boldsymbol{x} = [w_1, w_2, \ldots, w_n]$ -where w_i denotes the i-th word or token-into corresponding low-dimensional embedding vectors. Each word w_i is represented by an embedding vector $\boldsymbol{e}_i \in \mathbb{R}^d$, where d is the embedding dimension. The embedding lookup operation can be formalized as

$$\boldsymbol{e}_i = \mathrm{Embed}(w_i) \qquad \text{Eq. (1)}$$

where Embed (·) denotes the lookup table mapping each vocabulary term to its learned representation.To leverage subword semantics and capture domain-specific patterns, FastText augments the message vectorization by incorporating bag-of-n-grams. The final log representation $\boldsymbol{v}$ is computed as the mean of words and n -grams embeddings:

$$\boldsymbol{v} = \frac{1}{n+m}\left(\sum_{i=1}^{n} \boldsymbol{e}_i + \sum_{j=1}^{m} \tilde{\boldsymbol{e}}_j\right) \qquad \text{Eq. (2)}$$

where m is the count of unique n -gram tokens in the log, and $\tilde{\boldsymbol{e}}_j$ denotes the embedding for each n-gram. The aggregated representation $\boldsymbol{v}$ is then passed to a linear output layer to yield the raw (unnormalized) class scores. Specifically, for K fault classes, the output logits vector $\mathbf{z}$ is:

$$\mathbf{z} = \mathbf{W}\boldsymbol{v} + \mathbf{b} \qquad \text{Eq. (3)}$$

where $\mathbf{W} \in \mathbb{R}^{K\times d}$ and $\mathbf{b} \in \mathbb{R}^K$ are the trainable weight matrix and bias vector, respectively. Classification probabilities are obtained by applying the softmax function:

$$p(y = c \mid \boldsymbol{x}) = \frac{\exp(z_c)}{\sum_{j=1}^{K} \exp(z_j)} \qquad \text{Eq. (4)}$$

where $p(y = c \mid \boldsymbol{x})$ is the predicted probability of class c given the log input $\boldsymbol{x}$.
During training, FastText minimizes the cross-entropy loss over all training examples:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} \log p\left(y^{(i)} \mid \boldsymbol{x}^{(i)}\right) \qquad \text{Eq. (5)}$$

where N is the batch size, and $y^{(i)}$ the ground-truth label for instance i.

For efficiency in large scale and multi-class scenarios typical to IIoT logs, the hierarchical softmax variant is implemented to reduce computational cost:

$$\text{cost} \sim O(\log_2 K) \qquad \text{Eq. (6)}$$

where K is the total number of classes. Finally, the inference complexity can be approximated as.

$$O(|\boldsymbol{x}| \cdot d + d \cdot K) \qquad \text{Eq. (7)}$$

where $|\boldsymbol{x}|$ denotes the number of tokens per log entry.

In contrast to transformers, FastText is a shallow model with low pre-training and inference resource needs. It can be rapidly updated and implemented in edge computing contexts of IIoT networks with low latency and hardware. Additionally, FastText has native capability for online learning and is better able to adjust to changes in the log domain and operation.

Additionally, FastText explicitly models subwords and performs well empirically in IIoT. Compound error codes, port identifiers, or sub-system tags that appear partially rather than exactly are present in many industrial log entries. As seen above, subword n-gram embeddings can enhance generalization by mapping out a relationship for new tokens based on learnt representations.

Mathematical Formulation and Optimization

In both theory and practice, a solid grasp of the mathematical underpinnings of the FastText-based fault log classifier is required. The loss functions, optimization techniques, decision rules, and efficiency analysis of our industrial model's real-time deployment are presented in this section.

At the core of the classification model lies the objective to learn network parameters that minimize the expected predictive error over a dataset of labeled log messages. For a given input $\boldsymbol{x}$, characterized by its tokenized and embedded feature vector, and its true label y, the FastText architecture outputs a score vector $\mathbf{z}$, corresponding to all possible classes, which is then passed through a softmax activation to yield class probabilities as follows:

$$p(y = c \mid \boldsymbol{x}) = \frac{\exp(z_c)}{\sum_{k=1}^{K} \exp(z_k)} \qquad \text{Eq. (8)}$$

where z_c is the predicted logit for category c, and K is the total number of classes.
To measure how well the predicted probabilities align with the true class labels, the categorical cross-entropy loss is employed during training:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} \log p\left(y^{(i)} \mid \boldsymbol{x}^{(i)}\right) \qquad \text{Eq. (9)}$$

with N denoting the batch size, and $y^{(i)}$ the true class label of the i-th training example.
Parameter learning is conducted via stochastic optimization. Specifically, for parameters $\Theta = \{\mathbf{W}, \mathbf{b}, E\}$ (output weights, bias, and embedding matrix), the update rule in one stochastic gradient descent iteration is:

$$\Theta \leftarrow \Theta - \eta \cdot \nabla_{\Theta}\mathcal{L} \qquad \text{Eq. (10)}$$

where η is the learning rate, and $\nabla_{\Theta}\mathcal{L}$ is the gradient of the loss function with respect to model parameters. Backpropagation is used to efficiently compute gradients for all parameters in the network. Given the prevalence of class imbalance in industrial log datasets, decision calibration is crucial. To mitigate class bias, the probability threshold for decision-making can be tuned. The final predicted class $\hat{y}$ for input $\boldsymbol{x}$ is then given by:

$$\hat{y} = \arg\max_{c} p(y = c \mid \boldsymbol{x}) \qquad \text{Eq. (11)}$$

This rule ensures that, regardless of class frequency, the model selects the most probable class as the final output. In some environments, the model requires minimal latency and a smaller resource pool. In FastText, embedding lookups and matrix multiplications in the output layer determine the inference time of a single example. This is the complete complexity:

$$\text{Complexity} = O(|\boldsymbol{x}| \cdot d + d \cdot K) \qquad \text{Eq. (12)}$$

where $|\boldsymbol{x}|$ represents the length of the tokenized input, d the embedding dimension, and K the number of classes. To improve model generalization and stability, L_2 regularization is applied to the weight matrix. The regularized loss can be formulated as:

$$\mathcal{L}_{\text{total}} = \mathcal{L} + \lambda \|\mathbf{W}\|_2^2 \qquad \text{Eq. (13)}$$

where λ is a regularization coefficient, and $\|\mathbf{W}\|_2^2$ is the squared Frobenius norm of the output weights.
Training proceeds iteratively until the validation loss converges or a maximum epoch threshold is reached. The optimal parameters are then deployed for real-time log inference. Thanks to the hierarchical softmax structure, the effective cost per prediction is further reduced from $O(K)$ to $O(\log_2 K)$, significantly streamlining large-scale multi-class classification.

Experiments and Industrial Application

Experimental Setup and Metrics

A large-scale IIoT dataset gathered from a top electronics manufacturing plant was used to assess the suggested FastText-based industrial log classification system. About 1.2 million tagged log samples covering 15 operational failure categories are included in this collection. With more than half of the entries concentrated in the three most common classes—"sensor communication error," "equipment power warning," and "network instability"—analysis reveals a highly unbalanced distribution, with several rare-event categories accounting for less than two percent of the total. The exact category proportions are shown in Figure 3a, highlighting the stark class disparity typical of actual industrial applications.

The dataset is additionally characterized by message structure variability. Most log entries are brief, with a median of 13 characters per message; nevertheless, there is considerable variance, with a standard deviation of 5.8 tokens. This shows a combination of more detailed diagnostic reports and succinct system messages. The complete distribution of log message lengths is shown in Figure 3b, demonstrating that operational data flow is dominated by compact entries.

A server cluster with Intel Xeon Gold 2.5GHz CPUs, 128GB RAM, and NVIDIA Tesla V100 GPUs was used for the experiments; however, in order to better represent real-world deployment at the factory or edge level, all FastText training and testing were conducted on a single CPU core. Support Vector Machine, Naive Bayes, Convolutional Neural Network, Long Short-Term Memory network, and BERT were among the comparative techniques that were tailored to the particulars of IIoT log analysis and parameter-optimized by grid search. The dataset was split into 70% training, 15% validation, and 15% test splits using stratified sampling.

Differential sensitivity to different fault types is highlighted in Figure 3c, which shows the comparative recall across fault categories for all assessed methods. An analytical view of deployment feasibility is supported by the inference latency distribution shown in Figure 3d.

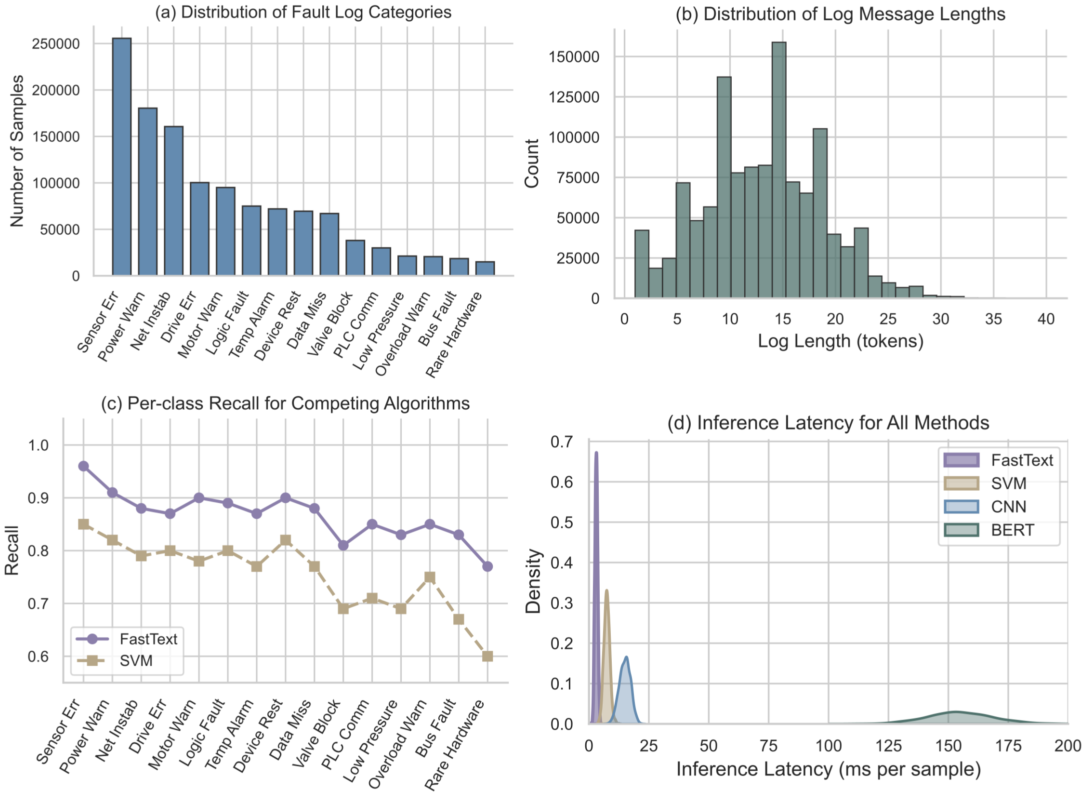

Figure 3. Experimental Dataset and Performance Metrics Overview:(a) Distribution of Fault Log Categories, (b) Distribution of Log Message Lengths, (c) Per-class Recall for Competing Algorithms, (d) Inference Latency for All Methods.

Performance Evaluation

In a generic comparison for industrial log categorization, FastText was compared to a number of base models. To guarantee that the results are realistic of actual industrial settings, all of the published results in this section make use of the extensive IIoT test set previously described and are consistent with other studies.

A horizontal bar chart comparing the overall classification accuracy of each of the aforementioned techniques is displayed in Figure 4a. The FastText model outperformed CNN (89.6%), LSTM (87.2%), SVM (85.9%), and Naive Bayes (81.7%) with a test accuracy of 91.8%, which was close to BERT's 92.1%. Both FastText and BERT are effective embedding models for industrial log data, and their absolute differences are negligible.

Figure 4b displays additional horizontal comparisons of the macro-averaged recall. With a macro-recall of 0.88, FastText surpassed both SVM and LSTM and was just marginally worse than BERT at 0.89. FastText can accurately identify all fault types, including uncommon ones, with high average recall rates.

A set of horizontal bar charts for the F1-scores of various common fault types in both the high-frequency and minority categories are displayed in Figure 4c. Among the less common categories, "rare hardware anomaly" and "bus fault" both retain an F1 score above 0.76, whereas FastText has attained the highest stability of F1 scores. On the other hand, SVM and LSTM's F1-scores in these classes are likewise below 0.70. CNN works poorly on low-frequency classes because of class imbalance, but it is steady for the primary classes. Subword information and embedding techniques can be used to alleviate the issue of extreme data skew, according to the analysis above.

The normalized confusion matrices and error categories for the aforementioned approaches are displayed in Figure 4d. Compared to the other models, FastText exhibits far less confusion for the majority of class pairs. However, in line with the error analysis, it shares a source of misclassification with the other approaches when it comes to differentiating between "device reset" and "network instability." These classes' primary confusion clusters correspond to ambiguity in system event reporting and overlap in log language.

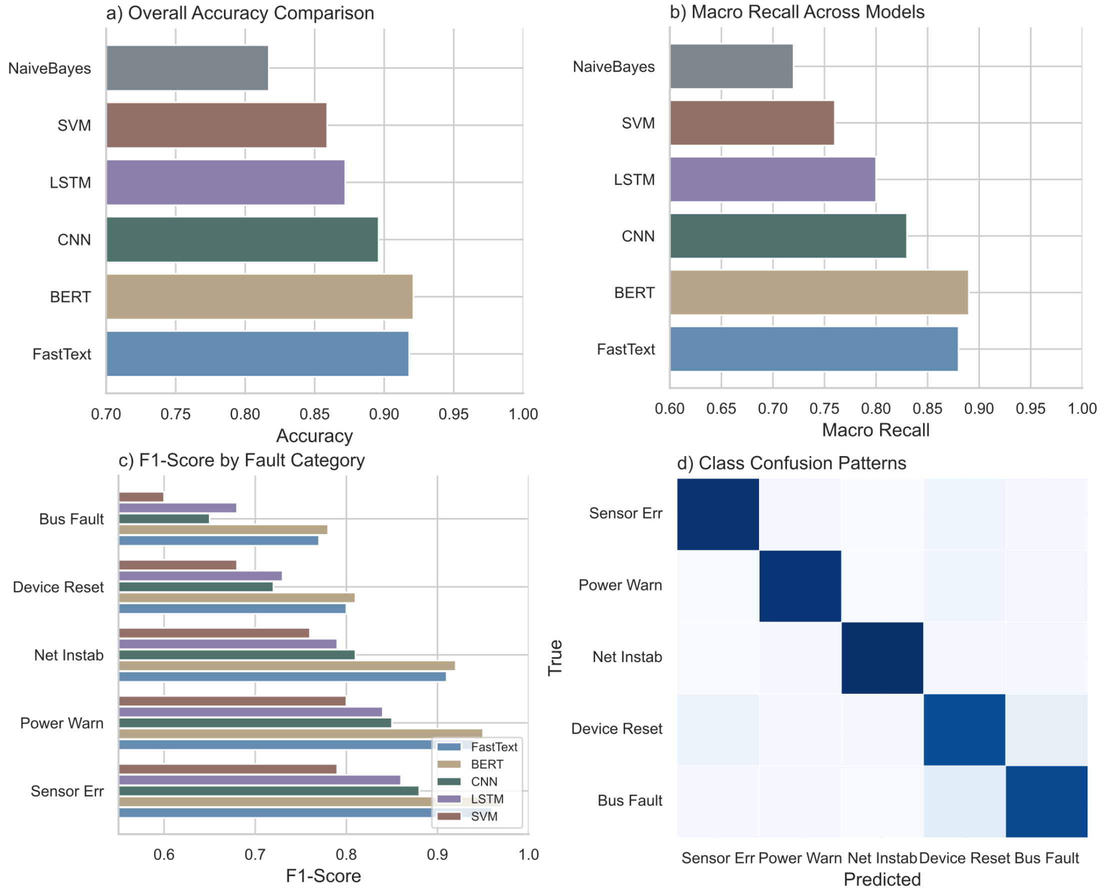

Figure 4. Comparative Performance of Fault Classification Methods: (a) Overall Accuracy Comparison, (b) Macro Recall Across Models, (c) F1-Score by Fault, (d). Category Class Confusion Patterns.

For the system to function in an industrial Internet of Things setting, the inference latency must be sufficiently low (see Figure 5a). Test on a single CPU: CNN and LSTM both surpass 15 milliseconds; FastText has a median per-log inference latency of 3.1 milliseconds, while SVM and Naive Bayes have averages of 7.4 and 5.5 milliseconds, respectively. Because of its comparatively sluggish inference speed—more than 155 milliseconds per sample—BERT is not appropriate for edge or latency-sensitive applications.

Deployment in BERT is not possible because, as Figure 5b illustrates, the total memory usage of FastText with the full set of parameters is less than 40 MB, while BERT requires more than 2 GB in inference mode. After adjusting the optimal parameters, CNN and LSTM neural network models have an intermediate size, between 180 and 380 MB, and are comparatively memory-efficient.

The number of log samples processed per second (throughput) by each approach is displayed in Figure 5c. FastText is significantly faster than all neural and transformer baselines, achieving a throughput of roughly 300 logs per second with a single CPU thread. BERT is not very practical since, as a transformer model, it has a lower throughput and is more susceptible to high latency swings.

Figure 5d illustrates how the error rate varies over time with the introduction of simulated concept drift, such as extensive firmware upgrades or abrupt changes in the data domain, in order to assess the stability of the industrial system under actual operation. After such modifications, FastText exhibits a transient rise in error rate; nevertheless, it can adjust to new or altered log patterns and returns to its initial accuracy after gradually adding new log data.

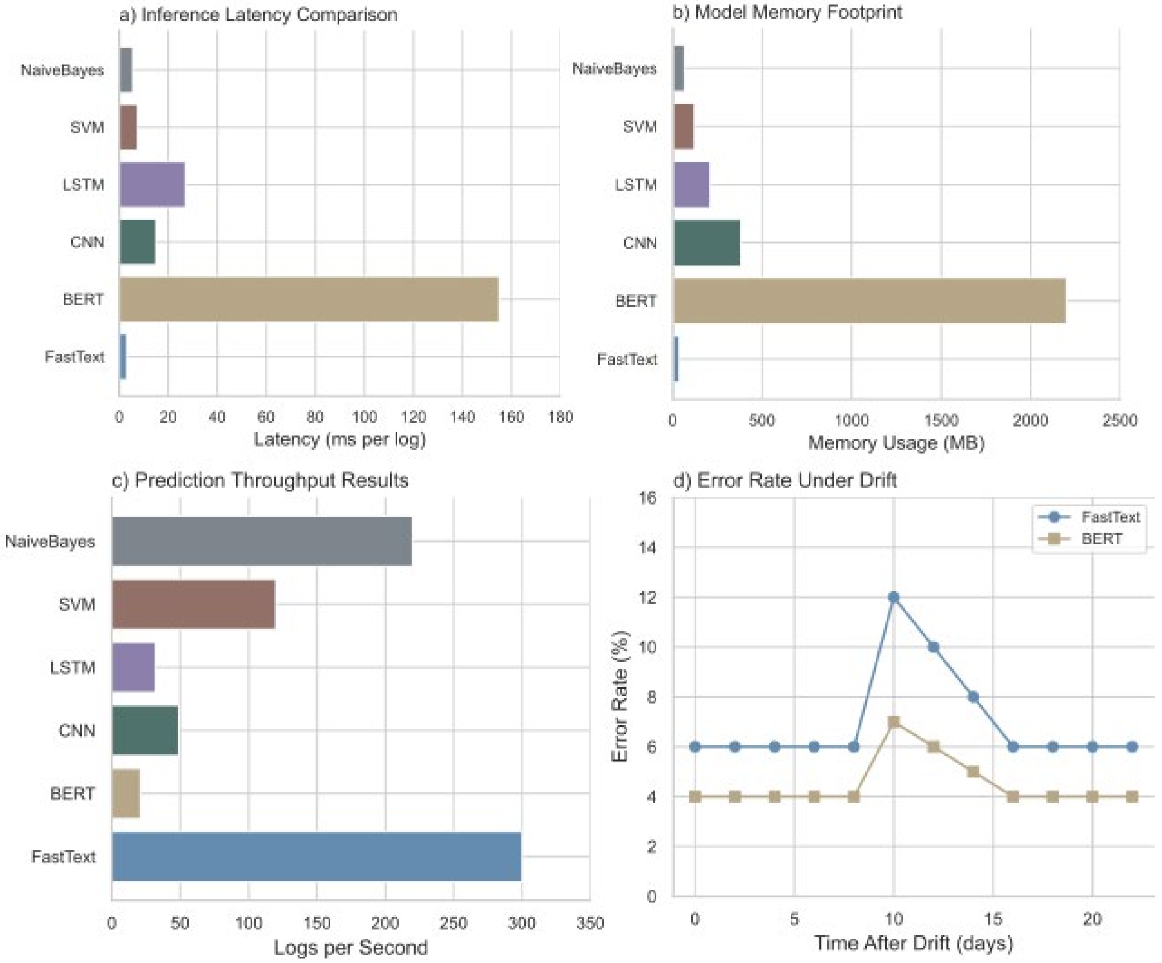

Figure 5. Deployment Efficiency and Robustness Analysis:(a) Inference Latency Comparison, (b) Model Memory Footprint, (c) Prediction Throughput Results, (d) Error Rate Under Drift.

The findings demonstrate that FastText is typically a good trade-off between deployment cost and prediction accuracy. It is ideal for continuous, real-time fault detection in high-performance IIoT applications because it can precisely identify both common and uncommon problems, has a minimal inference latency, and is very modest in size.

Application in Industrial IoT Scenarios

Due to its high complexity and mission-critical operation requirements, a FastText-based fault log categorization system has been widely implemented in the production environment of a top electronics manufacturing facility. The deployment's two objectives are to measure the downstream operating consequences of this automation in terms of response time, accuracy, and resource allocation, and to automate the real-time detection and categorization of system defects in a variety of industrial equipment.

The system must manage a significant volume of log data from numerous industrial devices in real time for ingestion and processing, according to the deployment architectural study. The volume of log processing changes greatly from day to day and even hour by hour within a given week, as seen in Figure 6a. For the log traffic in the industrial Internet of Things, a significant difference in workload is anticipated between the peak of the daytime production period and the off-peak maintenance period. The system must have good real-time processing and scalability due to the extremely dynamic and volatile nature of the data; in this case, the engineering verification of the log classification algorithm will take place.

As a result, the system's ability to react to maintenance requirements will alter in the future. The distribution of maintenance response times has drastically changed after six months, as seen in Figure 6b. In the automated system, the median issue handling interval has dropped from almost 38 minutes to 11.5 minutes. As a result, the machine-learning workflow's processing speed and priority assignment capabilities have increased.

To lower false alarms, fine-grained logging and log categorization have been put in place. The pattern of false-positive alert frequency in January, February, March, and other months prior to and following the system launch is depicted in Figure 6c. The classifier maintained a good signal-to-noise ratio even after a firmware upgrade during a challenging period; only a minor amount of retraining was required on occasion. The average reduction was 44%.

At the same time, costs have gone up. The decrease in overall unscheduled system downtime and associated direct operational costs is depicted in Figure 6d. According to plant accounting data from the pilot period, incident handling-related downtime hours have dropped by 19%. As a result, the main production line has realized annual savings of several hundred thousand dollars.

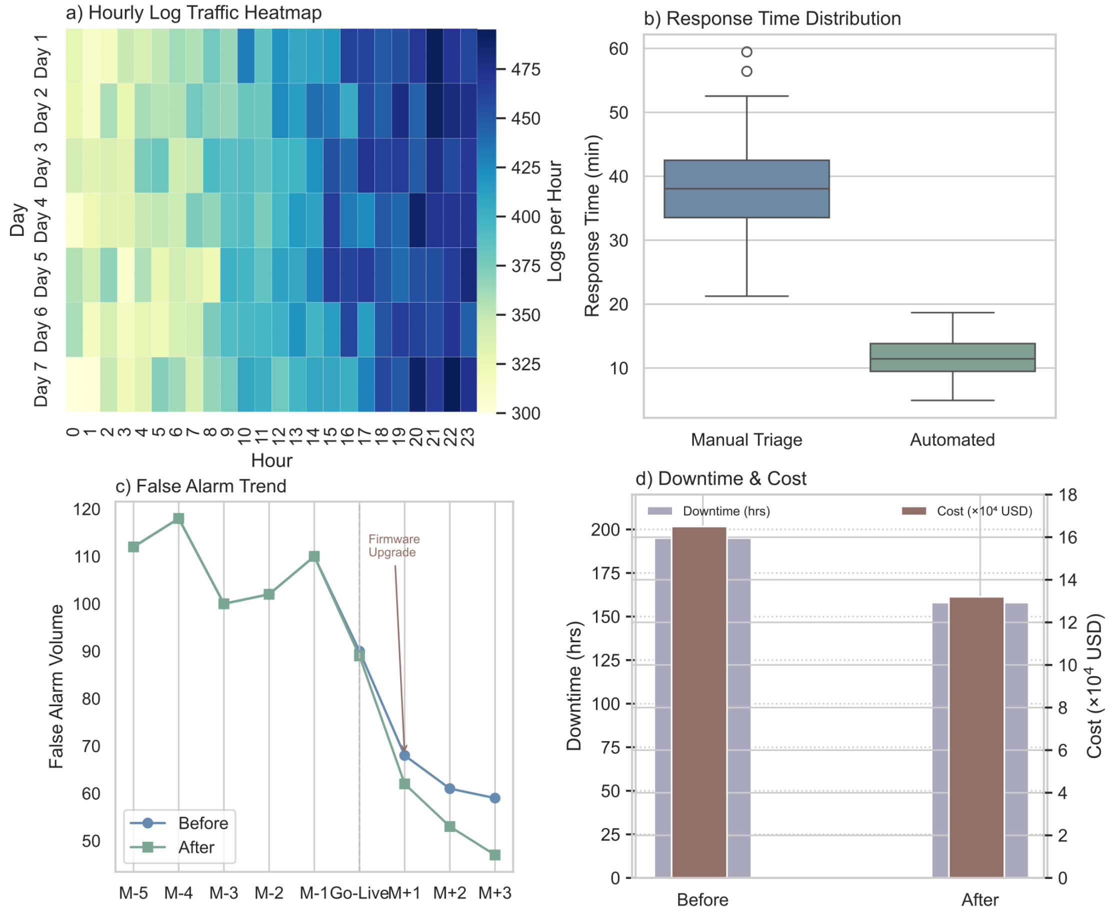

Figure 6. Operational Deployment and Direct Impact of Fault Log Classification(a) Hourly Distribution of Log Traffic Over a Typical Week, (b) Maintenance Response Time Distribution(c) Monthly Non-Actionable Alarm Reduction(d) Reduction in Downtime and Operational Cost

Several additional facilities with varying production quantities and process conditions have also been included in the deployment to verify whether the system is adequately robust and large-scale. Figure 7 is displayed in four different ways below. The average detection accuracy of several shop floors with noticeably diverse load profiles is displayed in Figure 7a; all of these were more than 91%, indicating good log classifier generalization. The inference latency statistics for both newer and older technology are displayed in Figure 7b, and all conditions are stable at less than 5 ms. A bar chart of alarm precision rates by environmental variable is shown in Figure 7c, and very slight variations were seen when humidity or temperature changed. Lastly, all user teams' alarm response times are compiled in Figure 7d, which demonstrates consistent efficiency under various staffing and shift arrangements.

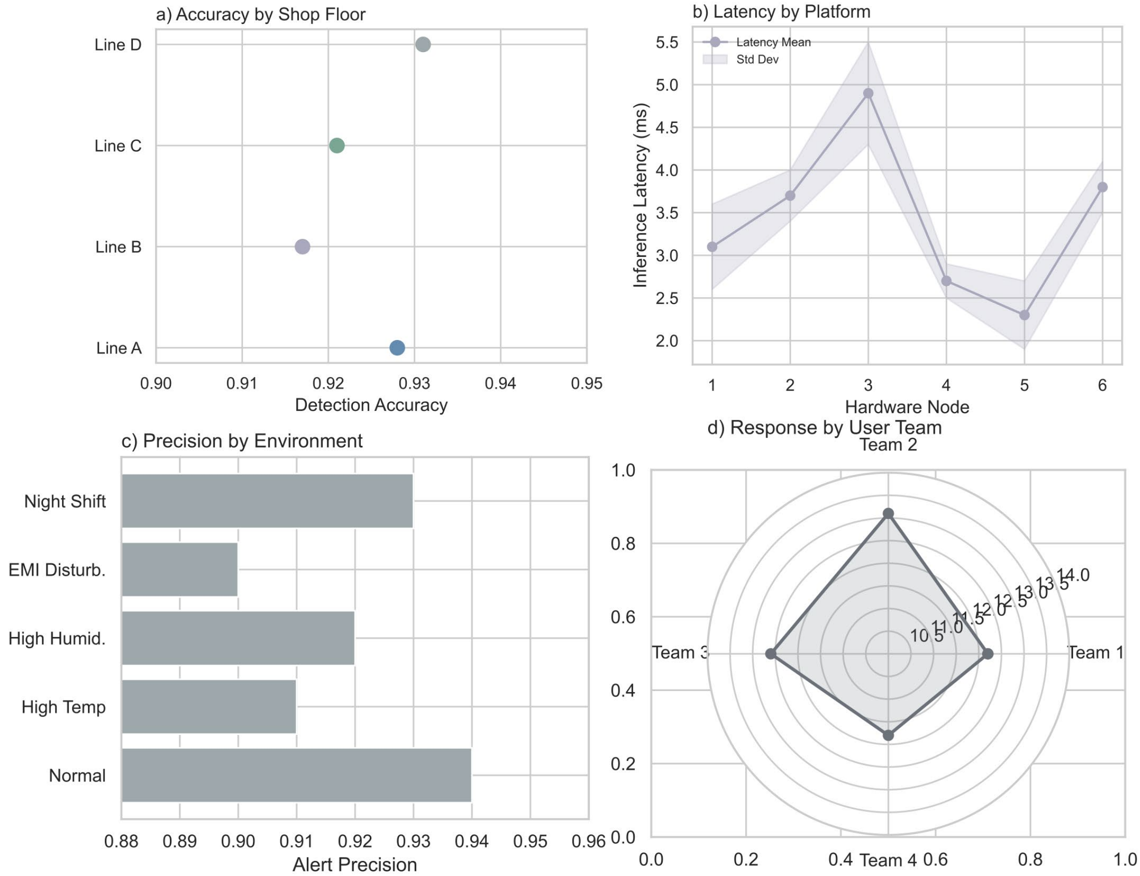

Figure 7. System Robustness and Broad Applicability Across Operational Conditions (a) Detection Accuracy by Shop Floor(b) Inference Latency by Hardware Platform(c) Alert Precision by Environmental Factor(d) Response Time by User Team

Together, the aforementioned findings demonstrate that the FastText-based classification framework's operational effectiveness and industrial viability in IIoT production are well supported. In addition to having a comparatively low system-level integration difficulties, the system has continuously been able to detect anomalous situations and enhance important operational indicators in real time. This approach has outstanding overall performance and will be appropriate for large-scale, complicated manufacturing sites, as demonstrated by Figures 6a–6d and the stable operation under various situations shown in Figures 7a–7d.

Conclusion

In order to perform effective semantic discrimination among complex, heterogeneous machine log data, this study has designed and validated an all-in-one automatic failure log classification system for the industrial Internet of Things (IIoT) using FastText. The new model is adaptable and data-driven to meet the demands of modern manufacturing; it is no longer labor-intensive or inflexible.

A new type of FastText model can be trained to better handle noisy and specialized log data, which is one of the study's initial findings. Demonstrate the system's ability to execute real-time inference with little computational load by directly deploying it on the production line. The framework has maintained high-performance classification under dynamic, high-load conditions and is still quick enough for practical factory applications, according to a thorough empirical analysis.

The system is economically viable in terms of both technology and operations, according to deployment outcomes. Unplanned downtime has decreased as a result of the automatic categorization pipeline's reduction in maintenance response time and, to some extent, the quantity of non-actionable alerts. Based on the above analysis of expenses, it can be determined how many are caused by real issues. The framework's versatility has been demonstrated by numerous shop floors and various hardware configurations, making it appropriate for large-scale industrial applications.

Incorporate additional digitalization of production diagnostics into this research. Because it is data-driven and vendor-neutral, it can be quickly incorporated into the existing IIoT system without requiring specialized hardware or lengthy retraining. In order to accomplish fault prediction and autonomous operation of smart manufacturing, a new industrial fault analysis approach is also being developed concurrently.

The benefits and drawbacks are listed above. FastText may perform worse in complex or ambiguous fault scenarios since it is a shallow model that is computationally cheap but does not explicitly capture long-range dependencies or subtle contextual alterations. Many real-world log datasets accessible in the industry are not adequately labeled, and the system is now less adaptable due to its supervised-learning foundation. We will create or use semi-supervised, self-supervised, or weakly supervised techniques to address this issue in order to decrease the quantity of manual annotation and increase the effectiveness of model adaption.

The second is that at this time, the model solely makes use of structured text data. Many types of data, including time series, audio, pictures, and more, are produced by contemporary IIoT deployments and are inappropriate for text-only models. To improve the coverage and resilience of fault detection and classification in early-stage anomaly detection, incorporate multimodal fusion. Create a sophisticated digital twin system that can facilitate long-term optimization by combining log-based analysis with process simulations and maintenance logs.

A few new deployment-architecture additions will be made. For a network with limited bandwidth or sporadic connectivity, switch to an edge-native, decentralized implementation to lower latency and improve fault tolerance. A continuous-learning module will be included to lessen the issue of model drift and preserve the model's excellent performance in the face of modifications to the factory layout and data distribution.

Author Contributions

Bartosz Sławomir Cyra contributes to conceptualization, methodology, software, validation, analysis, investigation, data collection, draft preparation, manuscript editing, visualization, project administration, and funding acquisition. All authors have read and agreed with the manuscript before its submission and publication.

Funding

This research received no specific financial support from any funding agency.

Institutional Review Board Statement

Not applicable.

References

[1] Sarker, S., Arefin, M. S., Kowsher, M., Bhuiyan, T., Dhar, P. K., & Kwon, O. J. (2022). A comprehensive review on big data for industries: challenges and opportunities. Ieee Access, 11, 744-769. https://doi.org/10.1109/ACCESS.2022.3232526

[2] Abeysekara, P., Dong, H., & Qin, A. K. (2021). Data-driven trust prediction in mobile edge computing-based IoT systems. IEEE Transactions on Services Computing, 16(1), https://doi.org/246-260. 10.1109/TSC.2021.3121879

[3] Shen, J., Morrison, M., Miao, H., & Gu, F. (2024, October). Harnessing deep learning for fault detection in Industry 4.0: A multimodal approach. In 2024 IEEE 6th International Conference on Cognitive Machine Intelligence (CogMI) (pp. 288-294). IEEE. https://doi.org/10.1109/CogMI62246.2024.00045

[4] Wu, Y., Dai, H. N., & Tang, H. (2021). Graph neural networks for anomaly detection in industrial Internet of Things. IEEE Internet of Things Journal, 9(12), 9214-9231. https://doi.org/10.1109/JIOT.2021.3094295

[5] Liu, J., Zhu, J., Bai, W., Zhang, H., Wu, L., Zhou, T., & Li, K. (2025). A Multimodal Lightweight Transformer for Bearing Fault Diagnosis Under High-Noise Industrial IoT Environments. IEEE Internet of Things Journal, 13(2), https://doi.org/3552-3567. 10.1109/JIOT.2025.3634730

[6] Qiu, T., Chi, J., Zhou, X., Ning, Z., Atiquzzaman, M., & Wu, D. O. (2020). Edge computing in industrial internet of things: Architecture, advances and challenges. IEEE communications surveys & tutorials, 22(4), 2462-2488. https://doi.org/10.1109/COMST.2020.3009103

[7] Wang, H., Wu, Z., Jiang, H., Huang, Y., Wang, J., Kopru, S., & Xie, T. (2021, November). Groot: An event-graph-based approach for root cause analysis in industrial settings. In 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 419-429). IEEE. https://doi.org/10.1109/ASE51524.2021.9678708

[8] Saucedo-Dorantes, J. J., Delgado-Prieto, M., Osornio-Rios, R. A., & de Jesus Romero-Troncoso, R. (2020). Industrial data-driven monitoring based on incremental learning applied to the detection of novel faults. IEEE Transactions on Industrial Informatics, 16(9), 5985-5995. https://doi.org/10.1109/TII.2020.2973731

[9] Alrifaey, M., Lim, W. H., Ang, C. K., Natarajan, E., Solihin, M. I., Juhari, M. R. M., & Tiang, S. S. (2022). Hybrid deep learning model for fault detection and classification of grid-connected photovoltaic system. IEEE Access, 10, https://doi.org/13852-13869. 10.1109/ACCESS.2022.3140287

[10] Antons, O., & Arlinghaus, J. C. (2022). Data-driven and autonomous manufacturing control in cyber-physical production systems. Computers in Industry, 141, 103711. https://doi.org/10.1016/j.compind.2022.103711

[11] Nizam, H., Zafar, S., Lv, Z., Wang, F., & Hu, X. (2022). Real-time deep anomaly detection framework for multivariate time-series data in industrial IoT. IEEE Sensors Journal, 22(23), 22836-22849. https://doi.org/10.1109/JSEN.2022.3211874

[12] Bilal, H., Obaidat, M. S., Aslam, M. S., Zhang, J., Yin, B., & Mahmood, K. (2024). Online fault diagnosis of industrial robot using IoRT and hybrid deep learning techniques: An experimental approach. IEEE Internet of Things Journal, 11(19), https://doi.org/31422-31437. 10.1109/JIOT.2024.3418352

[13] Chien, C. F., Hung, W. T., & Liao, E. T. Y. (2022). Redefining monitoring rules for intelligent fault detection and classification via CNN transfer learning for smart manufacturing. IEEE Transactions on Semiconductor Manufacturing, 35(2), https://doi.org/158-165. 10.1109/TSM.2022.3164904

[14] Anita, M., Anish, T. P., & Ezhilvendan, M. (2026). Predictive Analytics in Industrial IoT (IIoT): Enhancing Efficiency and Reliability. Enhancing Autonomous and Adaptive Systems With AI and IoT, 289-312. https://doi.org/10.4018/979-8-3373-3146-1.ch010

[15] Zhou, J., Wang, Z., Liu, J., Luo, X., & Chen, M. (2025). Modeling and Evaluation of Attention Mechanism Neural Network Based on Industrial Time Series Data. Processes, 13(1), 184. https://doi.org/10.3390/pr13010184

[16] Xie, Y., Zhang, H., & Babar, M. A. (2022, December). Loggd: Detecting anomalies from system logs with graph neural networks. In 2022 IEEE 22nd International conference on software quality, reliability and security (QRS) (pp. 299-310). IEEE. https://doi.org/10.1109/QRS57517.2022.00039

[17] Wang, Y., Yu, Z., Wu, J., Wang, C., Zhou, Q., & Hu, J. (2024). Adaptive knowledge distillation-based lightweight intelligent fault diagnosis framework in IoT edge computing. IEEE Internet of Things Journal, 11(13), https://doi.org/23156-23169. 10.1109/JIOT.2024.3387328

[18] Yan, P., Abdulkadir, A., Luley, P. P., Rosenthal, M., Schatte, G. A., Grewe, B. F., & Stadelmann, T. (2024). A comprehensive survey of deep transfer learning for anomaly detection in industrial time series: Methods, applications, and directions. IEEE Access, 12, 3768-3789. https://doi.org/10.1109/ACCESS.2023.3349132

[19] Marino, R., Wisultschew, C., Otero, A., Lanza-Gutierrez, J. M., Portilla, J., & De la Torre, E. (2020). A machine-learning-based distributed system for fault diagnosis with scalable detection quality in industrial IoT. IEEE Internet of Things Journal, 8(6), 4339-4352. https://doi.org/10.1109/JIOT.2020.3026211

[20] Kafunah, J., Ali, M. I., & Breslin, J. G. (2023). Uncertainty-aware ensemble combination method for quality monitoring fault diagnosis in safety-related products. IEEE Transactions on Industrial Informatics, 20(2), 1975-1986. https://doi.org/10.1109/TII.2023.3280566

[21] Ma, X., Keung, J., He, P., Xiao, Y., Yu, X., & Li, Y. (2023). A semisupervised approach for industrial anomaly detection via self-adaptive clustering. IEEE Transactions on Industrial Informatics, 20(2), 1687-1697. https://doi.org/10.1109/TII.2023.3280246

[22] Qian, C., Guo, Y., Liang, H., Song, J., & Yu, W. (2025). Secured edge intelligence in smart manufacturing CPS. In Edge Intelligence in Cyber-Physical Systems (pp. 377-401). Academic Press. https://doi.org/10.1016/B978-0-44-326572-3.00024-3

[23] Wang, Y., Shen, J., Yang, S., Han, Q., Zhao, C., Zhao, P., & Ren, X. (2024). Knowledge and data dual-driven fault diagnosis in industrial scenarios: A survey. IEEE Internet of Things Journal, 11(11), 19256-19277. https://doi.org/10.1109/JIOT.2024.3387538

[24] Wang, X., Wang, X., He, M., Zhang, M., & Lu, Z. (2023). Spatial-temporal graph model based on attention mechanism for anomalous IoT intrusion detection. IEEE Transactions on Industrial Informatics, 20(3), 3497-3509. https://doi.org/10.1109/TII.2023.3308784

[25] Lee, J., Singh, J., Azamfar, M., & Pandhare, V. (2020). Industrial AI and predictive analytics for smart manufacturing systems. In Smart manufacturing (pp. 213-244). Elsevier. https://doi.org/10.1016/B978-0-12-820027-8.00008-3

[26] Yan, S., Shi, L., Ren, J., Wang, W., Sun, L., & Zhang, W. (2026). Log Anomaly Detection via Transformers Pre-trained on Massive Unlabeled Data. IEEE Transactions on Network Science and Engineering. https://doi.org/10.1109/TNSE.2026.3654089

[27] Hoenig, A., Roy, K., Acquaah, Y. T., Yi, S., & Desai, S. S. (2024). Explainable AI for cyber-physical systems: Issues and challenges. IEEE access, 12, 73113-73140. https://doi.org/10.1109/ACCESS.2024.3395444

[28] Zvarivadza, T., Onifade, M., Dayo-Olupona, O., Said, K. O., Githiria, J. M., Genc, B., & Celik, T. (2024). On the impact of Industrial Internet of Things (IIoT)-mining sector perspectives. International Journal of Mining, Reclamation and Environment, 38(10), 771-809. https://doi.org/10.1080/17480930.2024.2347131

[29] Raouf, I., Kumar, P., & Kim, H. S. (2024). Deep learning-based fault diagnosis of servo motor bearing using the attention-guided feature aggregation network. Expert Systems with Applications, 258, 125137. https://doi.org/10.1016/j.eswa.2024.125137

[30] Zhao, W., Lv, Y., Liu, J., Lee, C. K., & Tu, L. (2023). Early fault diagnosis based on reinforcement learning optimized-SVM model with vibration-monitored signals. Quality Engineering, 35(4), 696-711. https://doi.org/10.1080/08982112.2023.2193255

[31] Zhang, W., Zhang, T., Cui, G., & Pan, Y. (2022). Intelligent machine fault diagnosis using convolutional neural networks and transfer learning. IEEE Access, 10, 50959-50973. https://doi.org/10.1109/ACCESS.2022.3173444

[32] Kumar, D., Ujjan, S. M., Dev, K., Khowaja, S. A., Bhatti, N. A., & Hussain, T. (2022). Towards soft real-time fault diagnosis for edge devices in industrial IoT using deep domain adaptation training strategy. Journal of Parallel and Distributed Computing, 160, https://doi.org/90-99. 10.1016/j.jpdc.2021.10.005

[33] Huma, Z. E., Latif, S., Ahmad, J., Idrees, Z., Ibrar, A., Zou, Z., ... & Baothman, F. (2021). A hybrid deep random neural network for cyberattack detection in the industrial internet of things. IEEE access, 9, 55595-55605. https://doi.org/10.1109/ACCESS.2021.3071766