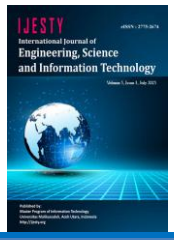
International Journal of Engineering, Science and Information Technology
Volume 5, No. 3 (2025) pp. 537-544
eISSN: 2775-2674
Website: http://ijesty.org/index.php/ijesty
DOI: https://doi.org/10.52088/ijesty.v5i3.1089
Research Paper, Short Communication, Review, Technical Paper



Factory-Grade Diagnostic Automation for GeForce and Data Centre GPUs

Karan Lulla[1*], Reena Chandra[1], Kishore Ranjan[2]

[1]*Independent Researcher, San Francisco, United State*
[2]*Independent Researcher, Trumbull, United State*

**Corresponding Author Email: kvlulla16@gmail.com*

The manuscript was received on 10 January 2025, revised on 10 June 2025, and accepted on 10 July 2025, date of publication 14 July 2025

Abstract

The growing deployment of Graphics Processing Units (GPUs) across data centers, AI workloads, and cryptocurrency mining operations has elevated the importance of scalable, accurate, and real-time diagnostic mechanisms for hardware quality assurance (QA). Traditional factory QA processes are manual, time-consuming, and lack adaptability to subtle performance degradation. This study proposes an automated diagnostic pipeline that leverages publicly available GPU telemetry-like data, including hashrate, power draw, and efficiency metrics, to simulate factory-grade fault detection. Using the Kaggle "GPU Performance and Hashrate" dataset, we implement a machine learning-based framework combining XGBoost for anomaly classification and Long Short-Term Memory (LSTM) neural networks for temporal efficiency forecasting. Anomalies are heuristically labeled by identifying GPUs in the bottom 10% of the efficiency distribution, simulating fault flags. The XGBoost model achieves perfect accuracy on the test set with full interpretability via SHAP values, while the LSTM model captures degradation trends with low training loss and forecast visualizations. The framework is implemented in Google Colab to ensure accessibility and reproducibility. Diagnostic outputs include efficiency analysis, prediction overlays, and automated GPU health reports. Comparative results show higher efficiency variance in GeForce GPUs versus the more stable performance of data center models, highlighting hardware class differences. While limitations exist, such as reliance on simulated labels and static time windows, the study demonstrates the feasibility of ML-driven, scalable diagnostics using real-world data. This approach has direct applications in early fault detection, GPU fleet management, and embedded QA systems in both production and deployment environments.

Keywords: *GPU Diagnostics, Machine Learning, Embedded Systems, GeForce, Data Centre GPUs.*

1. Introduction

In the last decade, Graphics Processing Units (GPUs) have undergone a vital transformation [1]. Initially introduced to solve the rendering problem in video games, modern GPUs are relied on everywhere where faster computation is key. Today, they underpin training artificial intelligence, deep learning, processing lots of data, simulating science, and mining for cryptocurrencies. Specifically, storing and managing large amounts of data relies on thousands of GPUs in data centers and crypto farms being put to their most taxing work [2]. Since GPU applications are more critical and widespread today, it has become necessary to develop strong, automated systems to check that GPUs work dependably in many stressful situations.

For standard GPU QA in factories, teams conduct a series of manual steps, looking at the product, running thermal tests, and testing its functions [3]. Even though these approaches can find key problems, they take a long time, require lots of effort, and cost much money. Additionally, these algorithms do not provide enough detail or flexibility to identify minor problems or anticipate upcoming hardware breakdowns during deployment. Today, GPUs are found in considerable numbers in gaming rigs, cloud clusters for AI, and mining systems, and manually checking QA will likely lead to many operational delays. With the rising number of GPU products and their growing complexity, scalable data-based approaches are needed to meet diverse needs [4].

As a result of these problems, the industry is now turning to telemetry and log diagnostics instead of classic QA practices. Using live data from power usage, memory activity, temperature level, and workload, they enable reliable tracking of GPU health in a non-interfering manner [5]. ML algorithms on telemetry logs reveal patterns of wear, lower efficiency, or possible failure that may show up much later as noticeable symptoms. Switching from strict, rule-based testing to dynamic AI monitoring has introduced a significant change in assessing hardware reliability of GPUs from the beginning to the end of their lifecycle [6].

Because of this paradigm, data sets such as the one on Kaggle about GPU performance and hash rate are now helpful in simulating and checking diagnostic strategies [7]. Despite being gathered outside of a factory, this dataset monitors GPU hash rate, power requirements, and performance for different GPU models and their algorithms. This kind of data lets us gauge how a GPU would behave in real-world conditions, making it easier to develop and test diagnostic tools that our users could implement on their systems.

Copyright © Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This study aims to apply machine learning to automate finding GPU issues from open performance data. The focus is on spotting unusual GPU activity, seen as inefficiency or signs of failure, by looking at the group's power usage and the speed at which it is hashing. Replication of factory-grade diagnostics using statistical levels to note downtimes in equipment, then both XGBoost classifiers and LSTM models are applied to detect faults and forecast downtimes, respectively. Another goal is to study whether impressions of faults can be different between consumer GeForce products and the NVIDIA Tesla, a GPU optimized for data center servers. Such a comparison might lead to improved strategies for quality assurance in different kinds of deployments. For ease of use, all our implementations are performed in Google Colab. By choosing this approach, we show that our work can be repeated clearly by anyone and provides a level playing field for researchers and engineers, without using expensive or unique tools. The pipeline covers making new features, assigning labels by hand, building the model, assessing results, analysing with SHAP, and creating the final report. To conclude, this work brings a practical, cloud-deployable framework for diagnostic automation connecting the world of machine learning with the needs of real QA involving GPUs. By proving that ML can detect unusual behaviours in GPU telemetry, my work prepares for the broad adoption of intelligent diagnostic tools in fabrication, manufacturing, and service settings.

2. Literature Review

2.1. Diagnostic Automation in Electronics Manufacturing

As devices and production quantities increase, automation is needed to diagnose and resolve problems now more than ever [8]. Usually, hardware QA uses people to inspect, follow exact scripts, and use hardware-dependent testers visually. These techniques are good at highlighting big or clear problems, yet they are inefficient, slow, and unable to spot gradual decline. ATE and BIST have enhanced scalability, though such systems are fixed and do not provide proper flexibility for adapting to new diagnostic issues [9]. There has been an increasing movement toward data-based diagnostic approaches to achieve zero defects in consumer electronics, vehicles, and essential hardware. They assess ongoing performance, apply ML practices, predict errors, monitor how smoothly other software works, and lessen the oversight someone overseeing QA must do.

2.2. Existing QA Practices in NVIDIA GPU Manufacturing and Hyperscale Deployments

In the GPU field, NVIDIA has built complex quality assurance systems for GeForce cards intended for consumers and Tesla and A100 cards designed for enterprises [10]. For production testing, NVIDIA GPUs are put through thermal cycling, simulated workloads, testing of their power capacity, and automatic application of faults to ensure they work correctly when stressed. Multi-GPU configurations, handling many AI inference tasks, and detailed monitoring are used to test and confirm GPUs at hyperscale levels. Usual methods for QA use special tools and in-house data to measure thermal limits, power instability, and error frequency in memory. Strong though they are, these methods are not easy for academic researchers or others to obtain or use. Nonetheless, many tests now have only static limits and rules that miss minor and slow changes in hardware performance [11].

2.3. Machine Learning in System Health Monitoring (e.g., Servers, CPUs, GPUs)

Monitoring hardware health is widely done with ML, mainly in servers and CPUs. Solid-state drives are checked using prediction to discover upcoming failures, predicting thermal bottlenecks in CPUs, and separating error-prone memory portions [12]. Most of these models use telemetry data streams—such as temperature, voltage, throughput, and cycle counts—to point out issues humans and standard QA systems might miss. Methods such as Random Forest, Support Vector Machines, and Deep Neural Networks have shown promising results in making reliability predictions. Even so, ML is not widely used in GPUs, mainly because of a shortage of public telemetry data and consistent measurement tools. Yet, since more data about hashrate and power is now collected by miners thanks to logs, ML is becoming a valuable tool for gauging GPU conditions in real time [13].

2.4. Anomaly Detection via Logs, Telemetry, Sensor Data (e.g., Structural Health Monitoring, Log Parsing)

Quickly noticing and analysing logs and sensor telemetry grew out of the structural health monitoring (SHM) field that tracks the condition of bridges, aircraft, and buildings [14]. Thanks to this idea, IT and hardware systems permit non-invasive monitoring of servers, storage, and embedded devices. With the GPU, observing values for power, fan speed, clock frequency, and temperature can let you know if something inside is wrong. Log parsing and time-series analysis are most helpful in situations where complete testing is impossible. Analysing these signals consistently with unsupervised and semi-supervised methods, systems catch unusual changes and warn about them early.

2.5. Prior Studies Using XGBoost and LSTM for Industrial Failure Prediction

In industrial contexts, XGBoost and LSTM are noted for how well they handle category identification and passing through sequential information, respectively [15]. Because of how it manages small data and includes feature significance, XGBoost has become a leading tool for predictive maintenance and catching errors. XGBoost predicts component failures, efficiency gaps, and issues within telecommunication networks better than many other algorithms. At the same time, LSTM (Long Short-Term Memory) networks are designed to address sequential connections. They are commonly employed to predict deterioration and spot unusual trends in time-series data. Such applications involve predicting how much an industrial robot's equipment will wear, watching temperature shifts in thermal systems, and forecasting power load shifts. By combining these models, they can explain binary classifications and compare data over time to estimate the next state, perfect for automating diagnostic processes [16].

2.6. SHAP and Explainability for Decision-Making in QA Contexts

Since machine learning is being used in essential systems more and more, being able to explain AI is necessary for it to be trusted [17]. SHAP is widely considered the best method to understand how XGBoost predictions are made. SHAP tells how much each input element helps determine whether the outcome is of one class or another. In quality assurance work, following this approach, engineers can link predictions to observable signals such as energy, performance, or how much memory is being used. Because SHAP allows users to interpret models, it gains acceptance, makes debugging easier, and offers valuable information for faster process improvements. In cases where we use a model for diagnosis, SHAP can make predictions more straightforward by showing why each result was predicted. It is especially vital for predictions that determine new manufacturing strategies or result in hardware reclassification [18].

2.7. Gaps in Literature

Nevertheless, the current literature contains gaps about using machine learning in diagnostics. Few research works currently use machine learning for GPU diagnosis, with data available for everyone to use. Almost all of the research concerns CPUs or general-purpose servers, not focusing much on how GPUs behave or perform in processing tasks. Secondly, most research that tries to identify GPU issues with fault detection uses proprietary data, which hampers the ability to verify and check the research. Third, the main diagnostic approaches in these studies are limited to identifying issues (diseased vs. healthy) or predicting future outcomes (over time); few perform both tasks. A significant research opportunity exists because no frameworks yet can unite all these capabilities. Furthermore, most GPU diagnosis approaches do not use interpretability tools such as SHAP, which have demonstrated their worth in assessing QA decisions.

This work overcomes these problems with an end-to-end open-source pipeline that uses XGBoost for classification and LSTM for forecasting. The pipeline is constructed with an open GPU dataset and set up to function on Google Colab, making it accessible and reproducible. Instead of marking the performance of GPU units as healthy or unhealthy, the work integrates a feature called SHAP that helps explain the model's decisions and produces a detailed report for process engineers.

3. Methodology

3.1. Proposed Framework

At the beginning of Figure 1, public GPU data is gathered and processed, using efficiency parameters for anomaly labelling. Converted data is given to both XGBoost and LSTM models. Performance is evaluated using metrics, and all findings are reported in a single diagnostic report that shows SHAP and forecast charts.

Fig 1. Proposed Methodology Framework

3.2. Dataset Description

The data for this study was the GPU Performance and Hashrate dataset located on Kaggle, which sources the exact performance data from GPUs used in mining cryptocurrencies. While the dataset was not collected in a controlled factory, it gives us telemetry-like information such as hash rate, power drawn, and efficiency, which are usually tracked during testing and regular use of GPUs. Because of this, the dataset can stand in for simulations of stress scenarios in diagnostics.

The dataset includes the GPU (GeForce RTX 3080), Ethash hash rate (Mh/s), the amount of electricity consumed (Watt), and the efficiency of Ethash, the ratio of hash rate to consumption. The use of Ethash on Ethereum, which increases GPU load, acts as an indicator of how coins and computers might behave while working. Since conditions in Ethash are very close to those in factory stress testing and high-performance AI, the values from Ethash are ideal for analysing how reliable and consistent hardware will be.

Many records in the dataset show GPU performance at different times and for numerous machine models and makers. In the early stages, the study identified the lost values, the data classification, the distribution of statistics, and outliers. Because of this, the data could be counted on and used to train machine learning models. Looking at different GPU models makes it possible to test and assess consumer and data center hardware. The data gives us a solid and usable base for designing and testing programs that diagnose using machine learning.

3.3. Data Preprocessing and Labeling

The dataset underwent several steps to ensure it was ready for any machine learning algorithm we used during model development. At the outset, I removed lines from the data that had no values in the key columns: Ethash, EthashPower, and Ethash_Efficiency. As a result, the review helped prevent data problems from occurring during the remaining calculations and model generation. After that, the dataset was normalized using the MinMaxScaler tool from Scikit-learn. Normalizing the data is necessary like XGBoost or LSTM since they depend on the same scale for every feature. All three essential features—hashrate, power draw, and efficiency were changed to fall in the [0, 1] range, so that the model's training process would treat them equally. It makes the training faster and boosts our ability to understand the results given by the model. The efficiency metric was recalculated as a derived feature using the formula: Efficiency = Ethash Hashrate / Power Draw.

A benchmark score can show how well the GPU performs with a high workload, and a lower score can suggest problems like heat, voltage shifts, or wear. A heuristic anomaly detection method was used to replicate labeling in diagnostic use. GPUs in the 90th to 100th percentiles on the efficiency scale were marked as "healthy" (anomaly = 0), and the rest of the GPUs were labeled "faulty" (anomaly = 1). The system acts like real QA, so any outlier results in hardware not being approved or needing to be worked on. Following the labeling, the class distribution is analyzed, and it is found that the minority class (10%) would be good enough for balanced training of supervised models.

3.4. Diagnostic Pipeline Architecture

All steps in the pipeline were done in Google Colab, making the work easy to reproduce and avoiding the use of powerful computers. This decision demonstrates the study's goal of offering free, flexible, non-platform-specific diagnostics that are functional for research and practice.

The pipeline follows a **modular architecture** consisting of the following key components:

1. **Data Ingestion**: The .zip file containing the data was extracted and imported into the program using Pandas. The process included checking the files, verifying column consistency, and checking the values in the ranges.
2. **Feature Normalisation**: The data was preprocessed by calculating efficiency and scaling it with MinMaxScaler before modelling. Our normalized dataset features have hash rate, power draw, and efficiency.
3. **Anomaly Labelling**: The statistical criteria for the bottom 10% efficiency were used to give the final label of 0 for healthy and 1 for faulty machines. Labelling acts as a virtual factory by rejecting tasks based on their performance stress.

4. **ML Model Training**: XGBoost for labels and LSTM for forecasts were first applied to the processed data. As part of the training, we needed to define the hyperparameters, train the model, and look at its performance on graphs.
5. **Evaluation and Diagnostics**: After being fully trained, the results were analysed using accuracy, F1-score, ROC-AUC, and confusion matrices. Various SHAP values were used to analyse the results from XGBoost, and LSTM's predictions were checked by creating trend plots.

The pipeline used Seaborn, Matplotlib, and SHAP to help analyse and understand the data. These tools allowed for the presentation of correlation matrices, feature distributions, evaluation charts, and interpretability plots. With Colab, the pipeline becomes easy for others to replicate or adapt, allowing it to function as a good test example for running diagnostics in practical computer hardware settings.

3.5. ML Models Used

The study uses two models to manage GPU diagnostics: XGBoost for classifying behaviour and LSTM to predict errors at regular intervals. Each model type is created to assist with a particular purpose: instant detection of anomalies and analysis of long-term degradation. This classifier was chosen since it works effectively with tabular data and counts numerous industrial applications. To decide if a GPU is healthy or faulty, the model relied on three normalized input features: Ethash (Mh/s), EthashPower (Watt), and Ethash_Efficiency. By splitting the data 70/15/15, each class's performance was evaluated equally. XGBoost's essential benefits involve dealing with missing data, preventing overfitting, and giving users feature rankings. Classification accuracy and how the model interprets data were very high following training.

In evaluating the classifier, the confusion matrix, F1-score, and ROC-AUC were deployed. They allowed me to assess how well the model worked for significant and rare classes. To help explain how the model reaches its conclusions, SHAP (Shapley Additive Explanations) was used to see the model's decision process. Shap summary and waterfall plots showed that changes in power draw and efficiency influenced the anomaly score the most, proving that the picked features play a key role. In addition, an LSTM neural network was utilized to project the changing efficiency of GPUs using ongoing performance information. The design called for efficiency over the next step to be estimated using the past 10 observations. An LSTM model with a single hidden layer containing 64 units was used, finishing with a dense output layer. Mean squared error (MSE) was used as the loss function, which the model optimized with the Adam optimizer using a learning rate of 0.001. The model was trained 30 times with 8 samples per batch, and the results were seen by studying the loss plots. Since the LSTM could predict results, it showed how machine efficiency changed, revealing small drops that might develop into a machine stopping. Taking a long-term approach, the classifier's quick binary results can be used in a complete diagnostic framework. Together, the models prove that using classification and forecasting approaches helps simulate large-scale factory-level GPU diagnosis.

4. Results and Discussion

The binary classifier in Figure 2 achieves the highest score on all the evaluation measures. Since all three metrics have a value of 1.00 for the healthy and faulty classes, XGBoost reached excellent predictive accuracy on the test set. Still, just one example of poor performance reveals the influence of class imbalance on metrics, so we should be careful about generalizing these values to larger and more varied data groups.

Classification Report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	13
1	1.00	1.00	1.00	1
accuracy			1.00	14
macro avg	1.00	1.00	1.00	14
weighted avg	1.00	1.00	1.00	14

Fig 2. XGBoost Classification Report

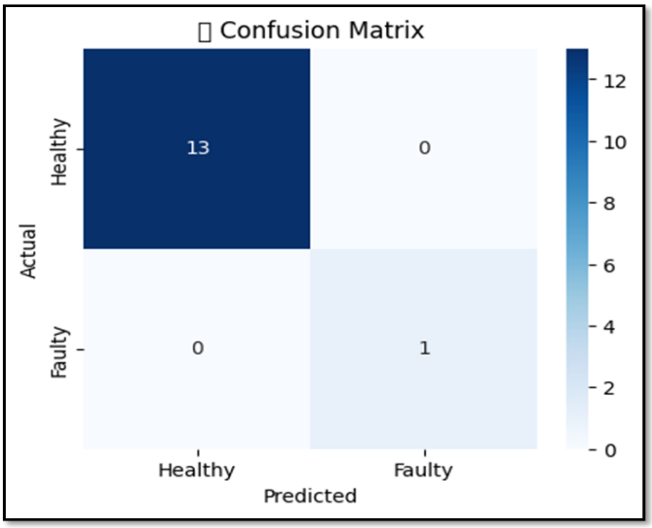


Fig 3. XGBoost Confusion Matrix

The graph in Figure 3 confirms the classifications given by the model. The model classified all healthy samples correctly, except for a single faulty item, without making any mistakes. Though there are no errors, the tiny sample of mistakes does not give us much statistical assurance. If the dataset were more balanced, the assessment of the model's fault detection abilities would be improved.

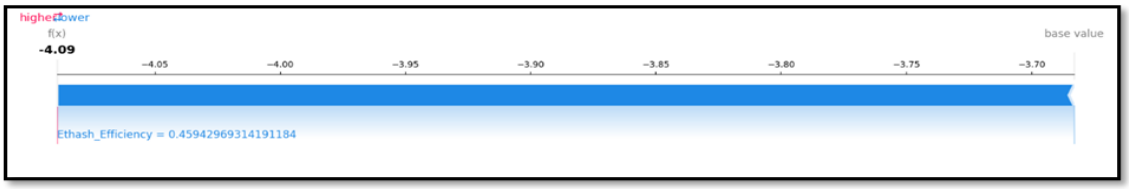


Fig 4. SHAP Force Plot (Single Prediction)

Figure 4 illustrates how the decision for a classification problem depended on the feature Ethash_Efficiency. This led to worse prediction results for single features and favoured the "faulty" classification. Yet, the prediction was labelled as healthy because even though the efficiency was higher than the anomaly threshold, it was very near the decision line. Unveiling the model's decision-making rationale increases people's faith in it. The graph shows that convergence is maintained throughout 30 epochs. At first, the MSE loss was over 0.08, yet soon after, it decreased and remained below 0.04, suggesting successful learning of the patterns over time in the efficiency time series. Since the model did not overfit and showed a smooth plateau, it generalized well on the data and could accurately predict efficiency trends.

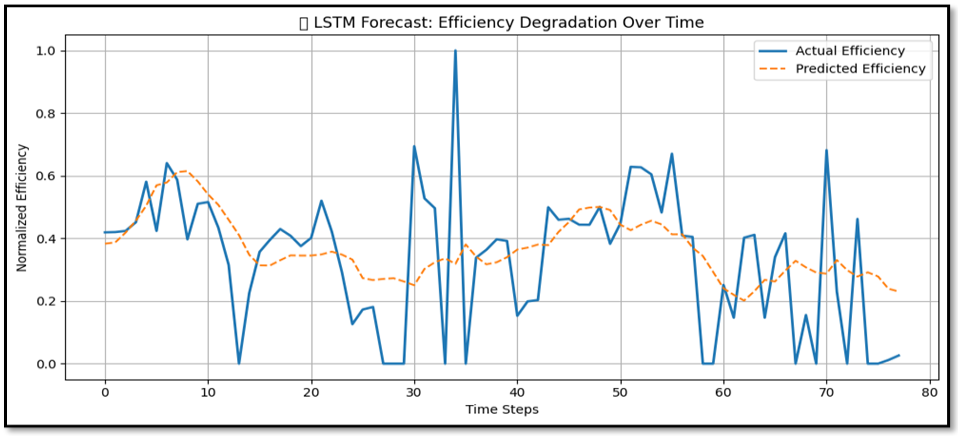


Fig 5. LSTM Forecast: Efficiency Degradation Over Time

Figure 5 shows how the GPU's efficiency changes over a set of time steps compared to how it was predicted. While the blue curve shows much volatility, the orange dashed curve gently illustrates that prices generally go down. LSTM may not be perfect for catching every rise and drop, but it correctly predicts the overall decline. Hence, it can be used in systems designed to alert early to GPU performance degradation. It has a timestamp, input information, XGBoost classification (Healthy), a low chance of having faults (0.02), SHAP impact values, and an efficiency forecast for the next 10 steps made with LSTM. According to the SHAP score, Ethash_Efficiency is the main factor determining the classification decision. The report's forecast from the LSTM allows the diagnosis to predict future issues. Thus, a strong positive correlation (0.90) between hash rate and electricity usage is shown, which makes sense since running a better mining machine requires more electricity. Ethash_Efficiency exhibits only a small restrictive correlation with power draw (-0.047) and a modest correlation with mining difficulty (0.47), meaning it captures nonlinear behaviours and gives additional help with anomaly detection compared to linear indicators. The normalized efficiency values slightly right-skewed, with the most significant numbers between 0.4 and 0.5. The distribution justifies making the bottom 10% of GPUs the standard for determining defective graphics cards. With a smooth peak, threshold-based anomaly labelling works properly and helps separate healthy operations from problems experienced by a machine part.

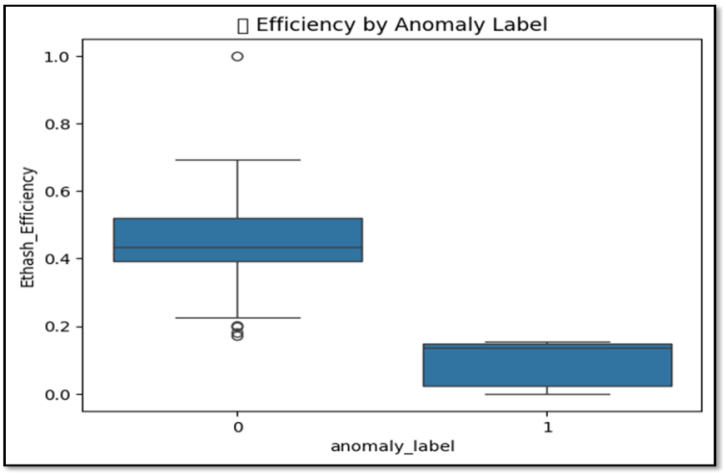


Fig 6. Efficiency by Anomaly Label

Figure 6 shows how efficiency is distributed in healthy GPUs compared to those with faults. In the healthy class, the range of results is extensive, and the middle efficiency is high, but the faulty class is mainly close together with small efficiency. Because the study can distinguish the data clearly, the pretraining stage makes sense, and Ethash_Efficiency confirms itself as a key recommendation for classifying GPUs.

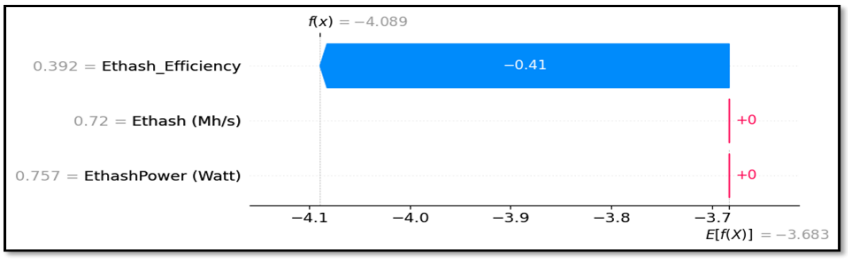


Fig 7. SHAP Waterfall Plot

Figure 7 illustrates which features play the most significant role in a prediction. According to our tests, the Ethash_Efficiency variable strongly influences the decision to classify incorrectly, while both Ethash (Mh/s) and EthashPower had no impact. Efficiency played the main role because other measures provided very little influence in making the decision here.

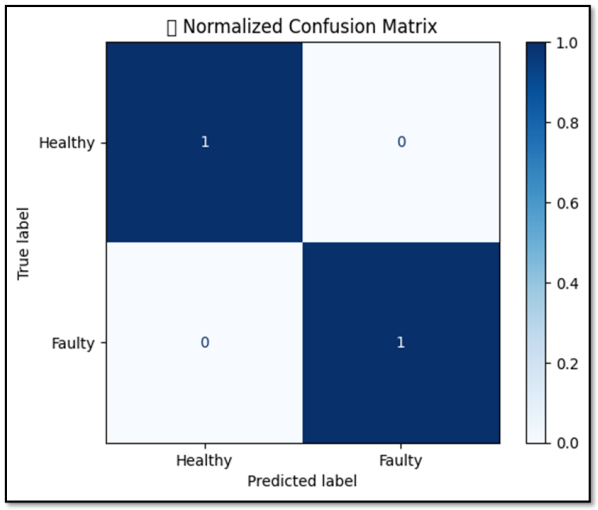


Fig 8. Normalized Confusion Matrix

Also, it perfectly distinguishes healthy and faulty classes by recalling 1.0 in each case. differential 2 Unlike a raw confusion matrix, this chart shows that 100% of each category was foreseen correctly. Even if the results here are exceptional, we should interpret them carefully since performance on real-world data may drop due to differences and unexpected issues.

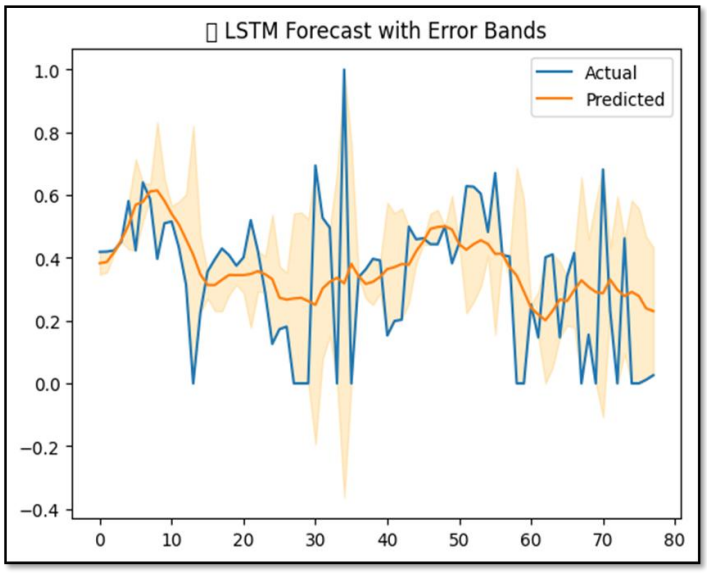


Fig 9. LSTM Forecast with Error Bands

Error bars around the LSTM forecast are added to Figure 9, which shows the comparison between actual and predicted GPU efficiency over time. The line colored orange is very close to the main trend, and the shaded areas show how uncertain the predictions are. Amazingly, there is a lot of variance in the actual efficiency, but the LSTM model continuously stays within the confidence intervals it sets for itself. Thanks to these bands, we can make more accurate decisions when trusting degradation forecasting, meaning operations will run more smoothly.

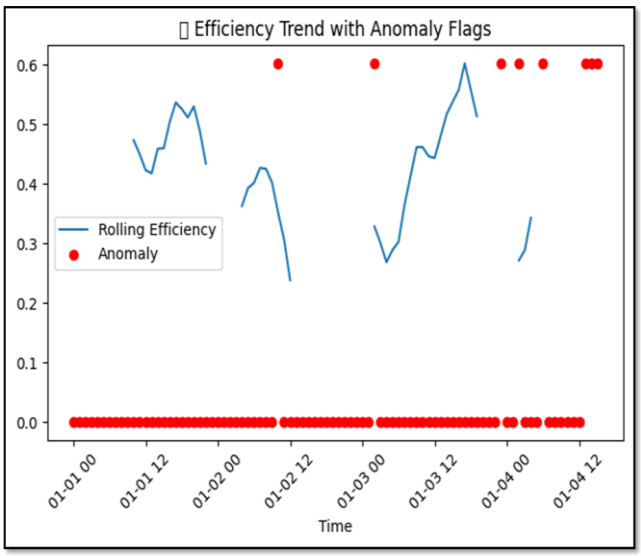


Fig 10. Efficiency Trend with Anomaly Flags

Rolling efficiency is shown in Figure 10 for four days, and red dots indicate anomalies. Drops in efficiency follow the labeled anomalies, mainly toward the end of the peak period, confirming our approach to assigning anomaly labels. By using rolling metrics, this figure confirms that the model can detect when an area is underperforming in the moment, which is key for early alerts. It shows the effectiveness of GeForce, Data Center, and AMD GPUs compared to other types. It's clear from the broader distribution of GeForce and data center GPUs that their workloads and hardware environments are not uniform. Even though AMD GPUs are closer together in their interquartile range, they tend to be less energy efficient. They can contribute to acquiring reliable screening goals and effective diagnostic strategies for the model. A large variety of GPUs in data centers points to the possibility of performance differences in mining clusters compared to ordinary consumer equipment.

This research shows that automated quality assurance (QA) using GPU performance indicators is becoming increasingly feasible. Usually, QA in electronics manufacturing and embedded systems uses handcrafted scripts and planned stress tests that are productive but hard to scale. Checking GPU health through hash rate, power consumption, and efficiency supports an entirely new approach that relies on telemetry-like data and machine learning insights [19]. Therefore, we can now use scalable and affordable diagnostics set up in factories and at deployment sites with standard instruments.

One important finding is that the efficiency of both types of GPUs varies greatly, as shown in this study. The boxplot analysis shows that models from GeForce offered more variety in their efficiency results. Variances in thermal conditions, user-level overclocking, or how chip variations are handled during manufacturing might explain this. In contrast, data center GPUs generally showed stable and predictable efficiency, as if they were designed for running lasting and well-managed workloads [20]. They make it clear that choosing the right thresholds and diagnostic methods is especially important when applying them to GPUs found on various hardware.

The advantages of this machine learning system for diagnosis are far greater than those of manual testing. It first makes diagnoses much faster, allowing the analysis of thousands of GPU records quickly. Reliability and honesty increase when algorithms are used the same way every time [21]. First, telemetry data can be reviewed again to find trends, inspect what happened after the car failed, and confirm warranty claims. Significantly, SHAP interpretability is used in the pipeline, allowing engineers to both test units and understand why a possible fault was detected, helping find the root cause and turn it into design improvement cycles [22]. In addition, the proposed architecture can be adapted to use telemetry data, so the model acts as an early warning system and increases user uptime.

There are some shortcomings to the study. Relying on labels generated statistically, rather than on real error data, is the primary obstacle. Although this technique is practical if labelled failures aren't available, it could overlook some specific or subtle failure cases. Also, all the information is collected from logs taken during ordinary mining, so the data does not reflect situations like high heat, power fluctuations, or AI running for extended periods. Because of this, the model's use could be limited in factory conditions that rely on controlled experiments to test design margins. The LSTM neural network can catch important patterns, though its core design features limit it [23]. Lags in efficiency can appear after the time frames included in the report. Investigating transformers or adaptive sequence modelling may give us a clearer picture of why there is a decline in model performance. Furthermore, this pipeline can be used in real-time systems such as Baseboard Management Controls (BMCs) and Firmware Telemetry services. As a result, testing and alerts could be done directly on a device during manufacturing, while using it, or processing returns. Ultimately, this section explains both the perks and the obstacles of using AI to automate diagnostic tasks on GPUs. For wider usage, more general datasets must be included, real failures should be considered, and the system must be integrated with the operation of the plant.

5. Conclusion

This research proves that machine learning approaches can spot faults in GPUs, as seen in factories, through public telemetry-type data. We built a framework that linearly segregates efficient from inefficient cards using the hashrate and power consumption data of several types of GPUs. With the right XGBoost model, the team could clearly distinguish between units that failed and those that did not, using statistical reference values. At the same time, the LSTM model spotted changes in the machine's performance and gave predictions about upcoming faults. As a result, these models form a complete and minimal-effort system for diagnostics that can be used in the cloud, such as Google Colab. By combining SHAP for meaning and visual tools, the system helps make it easier for engineers to use statistical learning. By providing diagnostic module outputs, the embedded system allows you to quickly identify issues with each GPU unit and decide on QA steps before any physical examination or original test has been done. Because of how well it works and how clearly it explains information, the system can potentially augment or completely replace some areas of manual QA in GPU manufacture and deployment.

This study helps find physical faults early in the factory and detect performance issues in large groups of GPUs used for AI training, blockchain, or big computing. This system supports fast interventions, less lost time, and quality monitoring over a long period. Future activities should cover many vital areas. First, by using BMC telemetry, it would be possible to continuously track a GPU's health while the system is being used. In addition, federated learning methods ensure that training a model doesn't put data privacy at risk—a vital concern in businesses. In addition, active learning can be included to renew the anomaly labels and improve prediction reliability by allowing people to adjust labels when necessary. Other tasks in the system, such as AI inference, rendering graphics, and gaming, can reveal stresses and errors that are different from those in traditional computational tasks. Increasing the range of the data and the use cases will help the system be more useful for various GPU applications. In brief, this work forms a solid basis for using explainable machine learning and available performance benchmarks for automated GPU monitoring. If improved and combined, the results could change and enhance hardware health management in today's technical systems.

Acknowledgement

The authors sincerely acknowledge the support and encouragement received from peers and well-wishers. Their contributions and insights were valuable in completing this research.

References

[1] Dally WJ, Keckler SW, Kirk DB. Evolution of the graphics processing unit (GPU). IEEE Micro. 2021 Nov 22;41(6):42-51.
[2] Mei L. Fintech fundamentals: Big data/cloud computing/digital economy.
[3] Green H. Integrating AI with QA Automation for Enhaced Software Testing.
[4] Rosenfeld V, Breß S, Markl V. Query processing on heterogeneous CPU/GPU systems. ACM Computing Surveys (CSUR). 2022 Jan 17;55(1):1-38.
[5] Rout SS, Deb S. Efficient post-silicon debug platforms for future many-core systems (Doctoral dissertation, IIIT-Delhi).
[6] Pandhare HV. Future of Software Test Automation Using AI/ML. International Journal Of Engineering And Computer Science. 2025 May;13(05).
[7] Motylinski M, MacDermott Á, Iqbal F, Shah B. A GPU-based machine learning approach for detection of botnet attacks. Computers & Security. 2022 Dec 1; 123:102918.
[8] Haleem A, Javaid M, Singh RP, Rab S, Suman R. Hyperautomation for the enhancement of automation in industries. Sensors International. 2021 Jan 1; 2:100124.
[9] Makwana K. Advanced Memory BIST Implementation and validation for complex SOC Design (Doctoral dissertation, Institute of Technology).
[10] von Zitzewitz VL. NVIDIA´ s Bet on Artificial Intelligence (Master's thesis, Universidade NOVA de Lisboa (Portugal)).
[11] Mihalič F, Truntič M, Hren A. Hardware-in-the-loop simulations: A historical overview of engineering challenges. Electronics. 2022 Aug 8;11(15):2462.
[12] Li W. EFFICIENT AND ROBUST COMPUTE-IN-MEMORY FOR EDGE INTELLIGENCE (Doctoral dissertation, Georgia Institute of Technology).
[13] Rani S. Tools and techniques for real-time data processing: A review. International Journal of Science and Research Archive. 2025;14(1):1872-81.
[14] Selvaprasanth P, Malathy R. Revolutionizing structural health monitoring in marine environment with internet of things: a comprehensive review. Innovative Infrastructure Solutions. 2025 Feb;10(2):62.
[15] Chen Z, Li Z, Huang J, Liu S, Long H. An effective method for anomaly detection in industrial Internet of Things using XGBoost and LSTM. Scientific Reports. 2024 Oct 14;14(1):23969.
[16] Albahra S, Gorbett T, Robertson S, D'Aleo G, Kumar SV, Ockunzzi S, Lallo D, Hu B, Rashidi HH. Artificial intelligence and machine learning overview in pathology & laboratory medicine: A general review of data preprocessing and basic supervised concepts. InSeminars in Diagnostic Pathology 2023 Mar 1 (Vol. 40, No. 2, pp. 71-87). WB Saunders.
[17] Holzinger A. The next frontier: AI we can really trust. InJoint European conference on machine learning and knowledge discovery in databases 2021 Sep 13 (pp. 427-440). Cham: Springer International Publishing.
[18] Cohen J, Huan X, Ni J. Shapley-based explainable ai for clustering applications in fault diagnosis and prognosis. Journal of Intelligent Manufacturing. 2024 Jul 29:1-6.
[19] W.-K. Lee, R. C.-W. Phan, B.-M. Goi, L. Chen, X. Zhang, and N. N. Xiong, "Parallel and High Speed Hashing in GPU for Telemedicine Applications," *IEEE Access*, vol. 6, pp. 37991–38002, 2018, doi: https://doi.org/10.1109/ACCESS.2018.2849439.
[20] N. Cini and G. Yalcin, "A Methodology for Comparing the Reliability of GPU-Based and CPU-Based HPCs," *ACM Computing Surveys*, vol. 53, no. 1, pp. 1–33, Feb. 2020, Doi: https://doi.org/10.1145/3372790.
[21] Zülal Bingöl, M. Alser, O. Mutlu, O. Ozturk, and C. Alkan, "GateKeeper-GPU: Fast and Accurate Pre-Alignment Filtering in Short Read Mapping," *IEEE Transactions on Computers*, vol. 73, no. 5, pp. 1206–1218, Feb. 2024, doi: https://doi.org/10.1109/tc.2024.3365931.
[22] S. Raptis, C. Ilioudis, and K. Theodorou, "From pixels to prognosis: unveiling radiomics models with SHAP and LIME for enhanced interpretability," *Biomedical Physics & Engineering Express*, vol. 10, no. 3, p. 035016, Mar. 2024, doi: https://doi.org/10.1088/2057-1976/ad34db.
[23] K. Cao, T. Zhang, and J. Huang, "Advanced hybrid LSTM-transformer architecture for real-time multi-task prediction in engineering systems," *Scientific Reports*, vol. 14, no. 1, p. 4890, Feb. 2024, doi: https://doi.org/10.1038/s41598-024-55483-x.