IoT-Based CNC Coolant Quality Detection using Photodiode and Gas Sensors with Incremental Learning Algorithms

Ardani Cesario Zuhri[1], Mario Ardhany[1,*], Agus Widodo[1], Ratna Mayasari[2], Albertus Rianto Suryaningrat[1], Galang Ilman Islami[1], Ahmad Musthofa[1], Nasril[1], Danny Mokhammad Gandana[1], Cecep Sujana[1]

[1]Research Center for Manufacturing Technology of Production Machinery, National Research and Innovation Agency, Jakarta 10340, Indonesia
[2]Research Center for Sustainable Industrial and Manufacturing Systems, National Research and Innovation Agency, Jakarta 10340, Indonesia

*Author to whom correspondence should be addressed:
E-mail: mari026@brin.go.id

(Received May 27, 2025; Revised July 04, 2025; Accepted December 22, 2025)

Abstract: Maintaining optimal coolant quality in CNC machining is critical because degraded coolant can harm the environment and machining efficiency. The current reliance on manual inspection by operators is subjective, inconsistent, and prone to inaccuracies, necessitating an automated and precise detection system. This study aims to develop an IoT-based sensor system for real-time, nondestructive assessment of CNC coolant quality, leveraging machine learning to objectively and adaptively classify coolant conditions as new data become available. A Raspberry Pi-based system integrates photodiode and gas sensors to monitor coolant quality, with data processed via cloud-based machine learning. The system categorizes coolants into three classes (very bad, bad, good) based on pH and microbiological analysis. Incremental learning algorithms, including Stochastic Gradient Descent-Logistic Regression (SGD-LR), SGD-SVM, Gaussian Naive Bayes, Passive-Aggressive (PA), and Perceptron, are employed to handle evolving data streams efficiently. The experimental results demonstrate high classification accuracy, with SGD-LR achieving the highest average accuracy (96.88%), closely followed by SGD-SVM (96.68%) and PA (96.33%). Gaussian Naive Bayes (95.07%) and Perceptron (94.88%) also performed robustly, proving the system's reliability in distinguishing coolant degradation levels. The proposed system eliminates subjective human judgment by providing consistent, real-time coolant quality detection. Its incremental learning capability ensures adaptability to new data, offering a scalable solution for industrial CNC maintenance. This approach can replace traditional manual inspections, reduce environmental risks, and improve machining efficiency.

Keywords: classification; cnc; coolant quality; incremental learning; machine learning

1. Introduction

Competition in an increasingly advanced global industrial world has raised the bar for modern manufacturing in recent years. Computer numerical control (CNC) machines are capable of making highly complex and precise workpieces in a short period of time by transforming a metal workpiece into a product using milling[1)], turning[2)], and boring[3)] operations, driven by a special machine program language. Lubrication and cooling systems, in this case, water-soluble coolants, are important factors in the machining operation process to improve machining smoothness, quality, and capacity, reduce tool wear, dissipate frictional heat, and extend the machine life[4–6)]. Sludge accumulation, such as debris, falling abrasive grains, and floating oil, can occur while the coolant circulates internally for a long period. The adverse effect of coolant circulation on human health is the oil mist sprayed into the air during the operation process, which contains biological factors, for example, bacteria, toxins, and machining deposits, posing a threat to the breathing of humans exposed around the engine area[7,8)]. Several case studies of workers' health effects during work with coolant reported 227 outcomes, of which 26 were related to cancer (including leukemia, liver cancer, stomach cancer, lung cancer, and laryngeal cancer), 58 to respiratory effects

Cite: A. Zuhri et al., "IoT-Based CNC Coolant Quality Detection using Photodiode and Gas Sensors with Incremental Learning Algorithms". Evergreen, 13 (01) 57-75 (2026). https://doi.org/10.5109/7405128.

(including asthma and hypersensitivity pneumonitis), 32 to skin effects (including dermatitis and skin allergies), 45 to microbial contamination (including skin irritation), and 76 to exposure measurements[9].

A coolant with a considerable amount of sludge accumulation causes scratches on the workpiece blockage of the circulation pump and microbial growth, which can damage the machine, lead to hazardous liquid waste, and worsen the environment[10–13]. The coolant is replaced multiple times within a specified period to prevent these issues. However, there are still many unpredictable technical and non-technical issues in the field. Currently, many machine operators identify the quality and feasibility of coolant by conventional means through direct contact with the coolant liquid, namely visual inspection with the sense of the eye, odor assessment with the sense of the nose, and liquid assessment by hand touch. These methods are difficult to quantify due to their unstable, error-prone measurements and the need for machine operator reliability and experience.

Water-soluble coolants usually have several parameters including the pH, debris, and microbial growth, that must be maintained to maintain the smooth operation of engines. The standard pH value of the coolant is usually in the range of 8 - 9.5. The fluid offers superior ferrous corrosion protection when the pH level is high, but it can cause issues related to skin softness and the prevention of non-ferrous corrosion. A lower pH will benefit mild and non-ferrous corrosion control although it could create issues with rancidity control and ferrous corrosion protection. Typically, pH below 8.5 is caused by bacterial action. The instability of the mix, ferrous corrosion control, and microbial control can be impacted. Chemicals known as additives can be used to increase the pH of the mixture. A pH value exceeding 9.5 is typically indicative of alkaline contamination, which can impact the liquid's lightness[14]. Machining debris can contaminate the cutting fluid circulation system, thereby affecting the precision of the machine cut; therefore, it is essential to keep a close eye on the cutting fluid to prevent excessive debris in the fluid's total volume[15]. The presence of organic compounds in water-soluble coolants affects microbial growth. Different types of fungi and bacteria can thrive in this environment. The machining process and duration affect the type and number of microorganisms in the coolant[11]. The growth of microorganisms, including reduced viscosity, decreased heat resistance, and decreased pH, is one of the factors that cause damage to the physical and chemical properties of coolants. Therefore, it is important to identify the microbial growth of both fungi and bacteria in coolants as it is to maintain the quality of coolant, lubrication effectiveness in machining processes, and the health protection of machine operators[16].

Some scientific methods for detecting coolant quality and viability include potential of hydrogen (pH) measurement[17], total dissolved solids (TDS), total suspended solids (TSS)[18], microscopy[19], and microbiological analysis[8]. However, some of these methods have specific difficulties, such as complex processes, long processing times, expensive tools and services, and sufficient experience and expertise. The Internet of Things (IoT) technology[20] and artificial intelligence enable more practical, economical, and accessible methods in the decision-making process[21]. This study proposes the realization IoT-based gas sensors and photodiodes to detect coolant quality in CNC machines using machine learning (ML). The implementation of ML algorithms on sensor data facilitates the detection of anomalous data and improves decision-making[22]. The combination of ML and IoT technologies in smart manufacturing provides a higher level of reliability and effectiveness in continuous, precise, efficient, and adaptive data collection and processing with minimal operational disruption in industrial settings[23–25].

Some previous research on coolant quality detection has focused on using a combination of sensors and measuring instruments called internal sensors, consisting of pH meters, Brix% meters, and sludge accumulation to investigate the influence of coolant quality on energy consumption[17]. In another study, the coolant was monitored in real time within a certain period using a probe containing pH and proximity sensors with an ESP32 microcontroller for data transmission[26]. Although traditional sensors are still used, great opportunities exist for detecting coolant quality for further optimization through the integration of sensors and ML.

Despite these advancements, existing methods for detecting coolant quality face critical limitations. First, traditional sensor systems (e.g., pH meters, TDS probes) require manual intervention and offline analysis, leading to maintenance decisions being delayed[17,26]. Second, batch-learning-based AI models[27,28] depend on static datasets, necessitating full retraining when new data arrive—a process that is computationally expensive and impractical for real-time industrial environments. Third, current approaches lack adaptability to dynamic coolant degradation patterns because microbial growth and chemical changes occur unpredictably over time[11,16]. Finally, multisensor fusion techniques in related domains (e.g., robotics, traffic management) struggle with high-dimensional or heterogeneous data[29,30], whereas coolant quality detection demands lightweight, temporally rich sensor data analysis tailored to industrial constraints.

Coolant testing is usually performed based on gradually increasing testing data. Hence, this study uses incremental learning (IL) algorithms, which can create a data model and modify it later when new data are available. The proposed approach can upgrade existing models rather than rebuilding a new model from scratch[31]. Hence, the prior knowledge gained from the data can be expanded for

future prediction[32].

Unlike batch-learning methods, the proposed IoT-based system leverages incremental learning algorithms to address these challenges. First, IL enables real-time model updates as new sensor data streams are introduced, eliminating the need for retraining from scratch and reducing computational overhead[31,32]. This is critical for dynamic industrial settings where the coolant conditions continuously evolve. Second, our gas sensor and photodiode array generate temporally rich data optimized for lightweight IL algorithms (e.g., SGD-LR, SGD-SVM), ensuring efficient processing even on edge devices like Raspberry Pi. Third, the system automates data transmission and decision-making by integrating IoT connectivity, removing subjective human inspections and enabling remote monitoring[20,23]. Our work is the first to apply IL to coolant quality detection, bridging the gap between ML and industrial maintenance needs.

Recent studies in stochastic modeling, such as novel mixture distributions, demonstrate advanced probabilistic frameworks for capturing complex data variability in engineering systems[33]. Although their approach offers theoretical robustness for multi-modal data, their application focuses on static datasets and lacks integration with real-time sensor networks or incremental learning, which are key requirements for dynamic industrial environments like CNC coolant monitoring. Similarly, the analysis of stochastic airfoil responses under stall flutter regimes highlights the importance of modeling temporal uncertainties in mechanical systems[34]. However, their study focuses more on aerodynamic instability prediction than on adaptive sensor-data classification, leaving a gap in scalable, IoT-driven solutions for real-time industrial condition monitoring.

Moreover, this IL approach has been effectively used in a variety of fields, such as intelligent robotics, unmanned aerial vehicles andautonomous driving[29], image processing[35,36], and traffic management[30]. This framework was also developed for an IoT-based smart water quality classification system to predict the suitability of water for different applications[37]. In addition to the genetic algorithm, it has been used for optimization and dynamic aeration control in wastewater treatment plants[38]. A review of the state-of-the-art IL methods concluded that IL is still a hot research area and will continue to be so for a long period[29]. Nevertheless, the use of IL algorithms for coolant quality detection has not been previously reported to the best of our knowledge..

Despite having common challenges with other domains, such as the need for real-time adaptation to dynamic environments, there are differences between our study and previous related works. The robotics dataset comprises multimodal data and high-dimensional data, whereas our dataset consists of time-series data specific to coolant characteristics. Similarly, image data are spatially rich, whereas our sensor data are temporally rich. Traffic management data deals with multi-modal data (time-series from sensors, images from cameras, text from social media), while our application is more focused on detecting specific physical properties (coolant quality). Compared with water suitability and wastewater treatment, which can be categorized as environmental applications, our coolant quality detection is more specific to industrial applications.

Table 1: Comparison of existing methods for detecting coolant quality and the proposed approach.

No	Method	Description	Drawbacks	Proposed Solution
1	Batch - Learning AI Models[27,28]	Trained on static datasets; requires full retraining for new data.	High computational cost. Delayed adaptation to dynamic coolant conditions. Impractical for industrial use in real-time.	Adapts Incremental Learning (IL) algorithms to update models in real time using IoT sensor streams.
2	Traditional Sensor Systems[17,26]	Manual pH/TDS probes with offline analysis.	Subjective inspections. Delayed maintenance decisions. No real-time monitoring.	IoT-enabled gas sensors / photodiodes automate data collection; IL processes streaming data in real time.
3	Multi-Sensor Fusion[29,30]	Combines heterogeneous sensor data (e.g., robotics, traffic).	Struggles with high-dimensional data. Complex integration for industrial constraints.	Lightweight IL algorithms optimize the temporal coolant-specific data.
4	Microbiological Analysis[8,19]	Lab-based microbial growth detection.	Time-consuming. Expensive equipment/expertise required. No real-time feedback..	Photodiode/gas sensors detect microbial proxies (e.g., pH shifts); IL dynamically classifies trends.
5	Static Probabilistic Models[33,34]	Theoretical frameworks for multi-modal data.	Designed for static datasets. No IoT integration or incremental updates.	IL adapts to real-time sensor streams in CNC environments

Most existing AI-driven sensor systems in industrial applications rely on batch learning, where models are trained on a complete, static dataset and require full retraining to incorporate new data[27,28]. These methods are computationally expensive and lack adaptability to evolving sensor data. This study proposes IL, which IL allows the model to be continuously updated as new sensor data become available without retraining the entire model from scratch. This allows real-time adaptation to changing coolant conditions while reducing computational costs and improving scalability.

Table 1 summarizes the limitations of existing coolant quality detection methods and compares them with the IL approach for coolant monitoring. First, batch-learning models lack real-time adaptability, requiring full retraining to incorporate new data. However, our IL approach enables continuous updates with minimal computational overhead[31,32]. Second, some industrial settings still rely on manual coolant checks, such as using smell, color, and touch, which are subjective, operator-dependent[9], and insensitive to critical degradation markers (e.g., early microbial growth[16]). Our IoT-integrated gas sensors and photodiodes automate data collection, thereby reducing subjectivity and operational downtime. Third, multisensor fusion techniques struggle with high-dimensional industrial data[29,30], but our feature selection and PCA methods may optimize dimensionality for coolant-specific time-series patterns. Fourth, microbiological analysis[8,19] is laboratory dependent and slow (often requiring days for microbial culturing), yet our system may dynamically classify trends. Finally, static probabilistic models[33] do not consider IoT integration, whereas the proposed framework adapts to real-time sensor streams in CNC environments.

The motivation for this study includes the following:

1) Develop and validate an AI-driven coolant monitoring system combining multisensor data, IoT connectivity, and incremental learning algorithms;
2) Identifying critical features from sensor data;
3) Practical industrial adoption of automated quality prediction through IoT and cloud-based interface.

This study introduces novelties that may advance the field of industrial condition monitoring, namely: (1) the integration of multisensor AI with IL for coolant quality, and (2) the end-to-end practical implementation. First, while prior research has used sensors or AI for coolant monitoring, this work is the first to combine photodiodes, gas sensors, IoT, and IL into a unified system. Unlike traditional batch-processing models, our approach continuously updates the model with new data. This study addresses a critical gap in adapting to real-time sensor and concept drift in industrial environments. Second, we built a functional web prototype that bridges AI and industrial usability beyond theoretical metrics. This system uniquely supports both real-time IoT data and offline uploads, with IL for lab-validated samples, a feature that was previously absent in coolant monitoring tools.

Therefore, the main contributions of our study include the following:

- A unified AI system combining multisensor (photodiodes, gas sensors), IoT, and incremental learning for real-time coolant monitoring.
- Identification of critical features from raw sensor data using dimensional reduction techniques (feature selection and PCA).
- Industrial deployment of an IoT-cloud system that eliminates manual inspections by enabling real-time remote monitoring and automated quality alerts.

The remainder of this paper is organized as follows. Section 2 describes the methods used in designing gas sensors and photodiode arrays, conducting coolant quality tests, IoT connectivity, preparing the dataset, preparing dimension reduction and classification techniques, and parameter optimization. Section 3 describes the experimental results and discusses the gas sensor and photodiode measurements, selected and extracted features, and classifier performance and testing results. Finally, Section 4 presents the conclusions of the research, practical implications, and suggestions for future research.

2. Methods

The steps to perform the experiment include: (1) configuring the sensors, (2) collecting coolant samples and setting their class reference, (3) measuring the coolant condition, (4) analyzing the feature importance, and (5) comparing the classification performances. Additionally, an IoT connection is designed to facilitate the coolant measurement step until the performance comparison step.

2.1. Design and configuration of sensors

The main components in the design of the sensor electronics system are smell and vision. Eight gas sensors from Winsen Electronics Technology and one photodiode from Thorlabs are used in these sensors. The gas and photodiode sensors detect the aroma emitted from the coolant and the presence of particles and contaminants in the coolant through changes in light intensity. The gas sensor was chosen to detect the unpleasant odor emitted by the coolant due to the growth of microorganisms such as fungi and causing pH changes in the coolant, which indirectly releases certain gases[8]. A photodiode sensor was chosen to detect turbidity and color in the coolant due to contamination or degradation of the coolant during the machining process[39–41]. Table 2 lists the sensors used in this study. Various gas sensors with range measurements from 10 to 10,000 ppm were used. The gas sensor converts the aroma produced into an electrical signal through internal changes in the sensor's internal resistance value, and the photodiode sensor converts changes in light intensity into an electrical signal with an infrared LED light source. The photodiode sensor has a wavelength of

900-2600 nm which is useful for detecting contaminants in the coolant. The photodiode sensor uses three LED light sources with peak wavelengths of 1300, 1450, and 1600 nm. All these sensors have a fast response time (<1s). This multisensor fusion approach combines gas and photodiode data, thereby avoiding false detection when using nonspecific gas sensors. In this study, each sensor's contribution was investigated. PCA and feature selection were used to identify which sensor has a great contribution. All sensors generate analog data that are connected to the Raspberry Pi through an analog-to-digital converter called (ADC) ADS1115. A multiplexer with three ADCs was used to accommodate eight gas sensors and one photodiode. The Raspberry Pi processes the data using Python. Then, it sends the data to the IoT-based cloud for further detection using ML. The electronic sensor system was used to identify the coolant samples using two sample boxes connected to a Raspberry Pi and the coolant placed inside. The first box has eight gas sensors on top and has a size of 6 inches (152 mm:250 mm) with a volume of 4.5 L. The second box has one photodiode sensor on top and has a size of 3.5 inches (90 mm:63 mm) with a volume of 0.3 L. The experimental setup is illustrated in Figure 1.

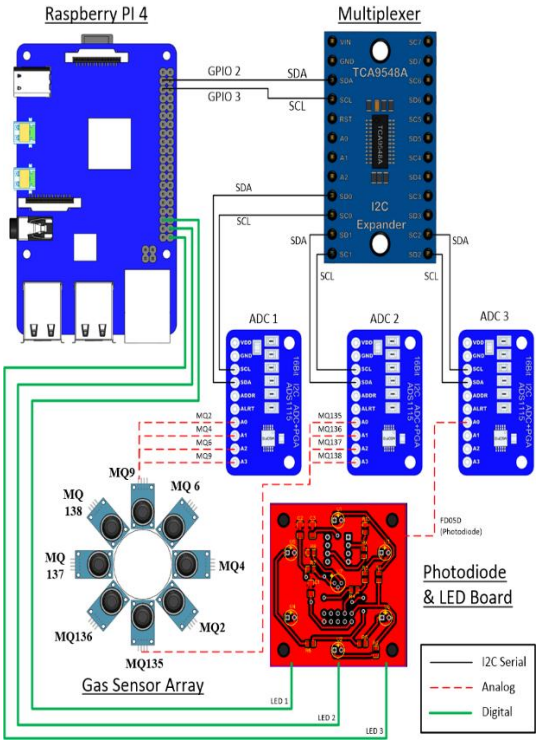


Fig. 1: Schematic of the electronic sensor

Table 2: List of the sensor

Initials	Sensors	Detection
M1	MQ2	detect hydrogen, LPG, and methane
M2	MQ4	detect natural gas and methane (CH4)
M3	MQ6	detect isobutane, propane, and LPG
M4	MQ9	detect CO, propane, and methane
M5	MQ135	detect CO2, benzene, NH3, and NOx
M6	MQ136	detect H2S, hydrogen, CO, and methane
M7	MQ137	detect ammonia, hydrogen, and ethanol
M8	MQ138	detect aromatic and other organic solvents
P1	FD05D	detect infrared light (wavelength 1600 nm)
P2	FD05D	detect infrared light (wavelength 1450 nm)
P3	FD05D	detect infrared light (wavelength 1350 nm)

The sensors used are normalized according to the datasheet specifications. Each gas sensor measured the clean air environment for 2h before the experiments. The measured resistance was saved as a reference for the initial resistance[42]. Meanwhile, the photodiode sensor was normalized using a Thorlabs LED Driver DC4100 by supplying the reference voltage value to drive the LED used in this research according to its typical value[43]. The measured voltage from the photodiode calibration was saved and later used as a reference in the experiment.

2.2. Coolant quality testing

Data validity is required as reference data to verify coolant quality by conducting pH testing and microbiological analysis. This test was conducted at the Biotechnology Laboratory, Serpong. The implementation of microbiological analysis testing refers to the Indonesian National Standard (SNI) ISO 4833-1:2015, while pH testing refers to SNI 06-6989:2004. Coolant samples were selected from several different CNC machines to ensure data completeness, representing various operational conditions and variations in coolant types in the production process, including the frequency of use, different material types, and varying coolant ages, as shown in Figure 2.

The pH test was performed using a Beckman pHI 50 pH meter, and microbiological analysis was performed using the pour plate method to obtain the total plate count (TPC). This method calculates the proliferation of microbial

Fig. 2: Coolant samples

Table 3: The coolant quality test results

Coolants	Value of pH	TPC (Cfu / mL)
C1	6.93	2.7×10^4
C2	6.12	1.3×10^6
C3	7.86	1.9×10^6
C4	6.76	1.0×10^6
C5	7.98	2.1×10^2
C6	8.6	1.9×10^1
C7	8.89	2.5×10^7

Fig. 3: Front view of the pH meter and the microbiological analysis

Table 4: The coolant quality test results

Category	Class	Definition
good	0	coolant with pH=7.5 - 9.5 and TPC<10^5
bad	1	coolant with pH=7.5 - 9.5 and TPC>10^5 or coolant with pH<7.5 or pH>9.5 and TPC<10^5
very bad	2	coolant with pH<7.5 or pH>9.5 and TPC>10^5

colonies in CFU/mL to assess microbial contamination of the coolant. In this study, the pH testing method was based on the SNI 06-6989: 2004 procedure. The sample size of coolant used is 10 mL, one testing frequency until the reading is stable, and environmental conditions during data collection with temperature 26.6°C and humidity 65%. The microbiological analysis method refers to the SNI ISO 4833-1: 2015 procedure. The sample size of the coolant used was 25 g, one testing frequency with dilutions of 10-1 to 10-7 until bacterial colonies could be counted and recorded, and environmental conditions during data collection with temperature 30 ± 1°C, and humidity 65%. Both tests met the coolant quality standards. Figure 3 displays the physical appearance of the pH meter and the microbiological analysis test performed using the TPC method. Table 3 shows the lower pH levels of the coolant were observed at C2 and C4, whereas the highest pH levels were observed at C6 and C7. The coolant with the highest TPC was observed at C7, whereas the coolant with the lowest TPC was observed at C1. Table 4 shows the coolant quality assessment outcomes, which were categorized into three groups: very bad, bad, and good.

The electronic sensor system designed with the gas sensor circuit and photodiode must be able to detect and identify the coolant quality correctly and precisely from the coolant that belongs to the very bad, bad, and good categories.

2.3. Measurement setup

Of the seven coolant samples tested for pH and microbiological analysis, three (C2, C3, and C4) were categorized as "very bad", two (C1 and C7) as "bad", and two (C5 and C6) as "good". The test time for each coolant sample was 1,110 s; the first 180 s was the chamber cleaning time, 30 s was the zero resistance setting time, and 900 s was the data acquisition time. Data sampling was performed at one data point per second or at a data acquisition frequency of 1 Hz. Box cleaning was performed every time one coolant sample was tested until no residual odor was left in the box. Each coolant sample was tested using eight gas sensors and one photodiode containing 11 × 900 × 1 = 9,900 data points.

Each coolant sample weighed 750 g in the first box containing the gas sensor and 250 g in the second box containing the photodiode sensor. The coolant samples were weighed using an electronic scale to ensure appropriate and accurate measurements. In the data collection test, the sensor was heated for 120 min and clean air data were collected for 3 min. The coolant sample was placed in the sample box, and data were collected for 930 s, including a 30 s buffer time for zero resistance settings and 900 s for actual ML data acquisition. The sample box was then cleaned before the next experiment.

Each sensor in the coolant sample contains 900 records. Therefore, for the seven coolant samples, each sensor has 900 x 7 = 6,300 records. Of the seven coolant samples, two are labeled as ‘very bad’, two others are labeled ‘bad’, and the three remaining samples are labeled as ‘good’. Therefore, the percentages of class distribution are 28.6%, 28.6%, and 42.8% for each class.

After preparing the dataset, we performed dimensionality reduction either by feature selection or feature extraction (using PCA). The last step is to classify the data according to class. The classification data were split into training and testing data at a percentage of 80%-20% to evaluate the performance of each classifier. To ensure deterministic partitions, we set the random_state parameter to a fixed integer value (e.g., random_state=101) in the Python’s ‘train_test_split’ function.

2.4. IoT connection

IoT platforms can remotely manage, collect, and monitor data coming from sensors in real time. IoT involves the expansion of internet connectivity for communication and interaction. The IoT architecture used in this study is shown in Figure4. The IoT system has three layers: perception, network, and application layers. In the perception layer, data from sensors were gathered using Raspberry Pi. The collected data are then transferred to the cloud server through the router on the network layer using REST API. Sensor data are sent to the cloud layer via HTTP requests, a protocol that transfers hypertext markup language (HTML) from a web server to a local browser with a request and response pattern that uses the HTTP GET and POST methods to simultaneously access the server, send requests, and store data in the application layer. Through HTTP, the server sends data through a universal resource identifier (URI), and the client receives data through the same URI[44].

The following parameters were sent in JSON format: dataset name, detection time (timestamp when the sensor retrieved the data), photodiode sensor data (FD05D_1300, FD05D_1450, FD05D_1600), and gas sensor data (MQ

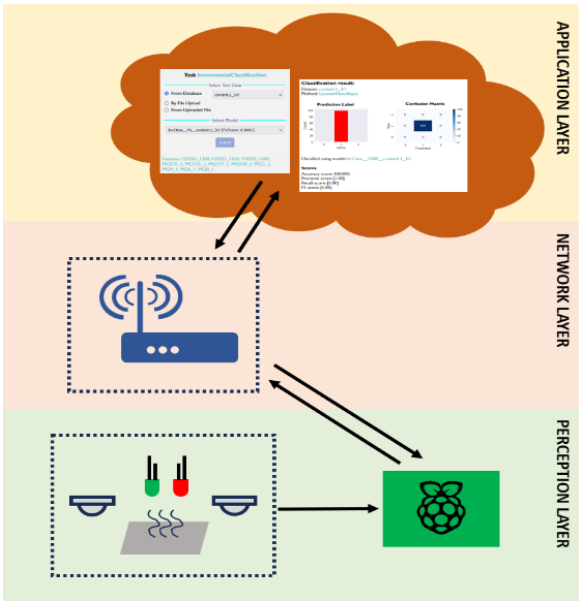


Fig. 4: Architecture of the IoT framework

135, MQ136, MQ137, MQ138, MQ2, MQ4, MQ6, and MQ8). Using the REST API, the Raspberry Pi communicates directly with the website server to ensure secure and efficient data transmission, which supports real-time monitoring and data analysis, which are critical to the effectiveness of the system. The website can be accessed through a mobile app, providing a user-friendly interface for quick coolant quality detection. This interface can be accessed from any device with internet connectivity, allowing industry managers to remotely view the coolant quality detection status without having to be physically present at the tunnel site. The cloud server keeps a continuous record of each part's status, creating a detailed database that industry managers and machine operators can access remotely.

To ensure security in our web application, we implemented a role-based access control system. Administrators have elevated privileges, allowing them to manage users, create machine learning models, upload training data, and fit classifiers. Regular users, on the other hand, are restricted to submitting test samples (inference data). For authentication, passwords are encrypted using MD5 hashing. Additionally, we enforce HTTPS to secure all HTTP transmissions, protecting data integrity and confidentiality during communication.

Furthermore, the proposed system exhibits varying latency characteristics across different operational stages. Initial model training requires approximately one minute per classifier to process the foundational dataset, representing a one-time computational investment to establish robust baseline performance. Subsequent incremental updates demonstrate significantly reduced latency, with partial fitting and accuracy evaluation operations completing in under one minute when processing user-submitted test samples. The most substantial latency occurs in the data acquisition phase, where IoT sensor data collection and transmission typically require 5-10 minutes due to inherent constraints in the sensing systems, network variability, and data volume. Hence, this multi-stage latency structure prioritizes computational thoroughness during initial model establishment while maintaining responsiveness for incremental learning operations.

2.5. Data preprocessing

In this study, data standardization and outlier detection were employed for the data pre-processing step. To ensure the optimal performance of scale-sensitive classifiers, feature standardization was applied using the StandardScaler from scikit-learn, which transforms the data to have zero mean and unit variance. The scaling is applied as:

$$X_{scaled} = \frac{X-\mu}{\sigma} \quad (1)$$

where μ is the mean and σ is the standard deviation of the feature values. This method is essential for algorithms whose learning processes are influenced by the magnitude of feature values. In particular, classifiers such as Stochastic Gradient Descent (SGD), Perceptron, and Passive-Aggressive rely on gradient-based or margin-based updates that are sensitive to input scale. Features with larger numerical ranges could dominate the learning process without standardization, leading to suboptimal convergence and performance. While Gaussian Naive Bayes (GNB) is not inherently sensitive to feature scale—because it models each feature independently as a normal distribution—standardization can still help improve numerical stability when features vary significantly in magnitude.

Outliers were identified using a two-step detection approach. First, the standard deviation (σ) method flagged values beyond $\pm 3\sigma$ from the mean[45]). Second, the interquartile range (IQR) method robustly detected extremes in skewed data. These methods were combined to capture both global deviations (σ) and localized anomalies (IQR), ensuring comprehensive outlier identification. IQR measures the spread of the middle 50% of data (outlier-resistant) as follows:

$$IQR = Q_3 - Q_1 \quad (2)$$

where $Q1$ is the 25th percentile and $Q3$ is the 75th percentile. The outlier threshold is typically defined as values outside $[Q_1 - k.IQR, Q_3 + k.IQR]$, where k's default value is 1.5[46]).

Outliers were defined as the union of indices flagged by either method for each feature, minimizing false negatives while maintaining interpretability.

Outliers were treated via per-feature Winsorization, which capped extreme values at the 5th and 95th percentiles of

the original data distribution[47]. This approach preserved the dataset's size and structure while reducing the influence of outliers. The treatment was applied only to pre-detected outliers, leaving non-outlying data points unchanged.

2.6. Feature selection

Feature selection is used to select a subset of features that can be used to differentiate samples into different classes[48]. These features improve learning execution in terms of higher precision, lower computational costs, and finer model interpretability. Feature selection generally involves three main methods: filter, wrapper, and embedded feature selection[49]. The filter method observes the relationship between features, whereas the wrapper method selects features using a classifier. The embedded method selects features during model optimization[50].

In this study, the filter method was selected because the classifiers designated for IL have no embedded feature selection capability, and the wrapper method is more computationally costly. Python's GenericUnivariateSelect function was used to compute the F-value of ANOVA (Analysis of Variance) for each feature relative to the target variable using the f_classif scoring function.

2.7. PCA

Feature extraction acquires information from the feature set to construct a new feature subspace[51], which reduces the number of dimensions or features. In this study, the feature extraction used is principal component analysis (PCA). PCA is used to extract the most relevant information by compressing the dataset into a lower-dimensional feature subspace[52]. PCA determines a set of comprehensive lists, and the primary variables are transformed into a different dimensional form through an orthogonal transformation that preserves the maximum variance[53]. In addition, this method is widely used to reduce data by retaining as much important information as possible[54].

2.8. Incremental learning algorithms

An incremental learning algorithm continuously updates its knowledge base by incorporating new patterns from fresh data without forgetting previously acquired information[55]. This type of learning is suitable for cool conditions, where data arrival is gradual. As shown in Figure 5, the model is updated with the previous data when labeled partial data arrive, incorporating the knowledge learned from all the previously observed data. The continuously updated model is then used to predict the incoming testing data class.

The proposed IL framework operates in three phases: (1) initial training, (2) incremental updates, and (3) evaluation. During initial training, the model is trained on the first batch of labeled data (initial data) using standard

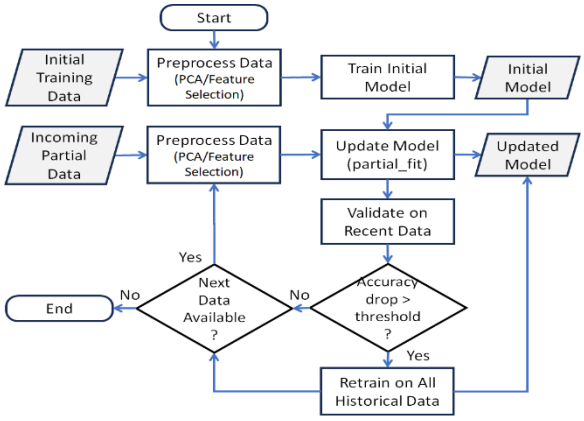


Fig. 5: Incremental learning algorithm

supervised learning. Feature selection/PCA is applied to reduce dimensionality (if enabled). As new partial data arrive, the model is incrementally updated via partial_fit (mini-batch learning), ensuring that prior knowledge is retained. After each update, the model's performance is validated on a held-out test set to monitor drift or degradation.

The IL algorithms used in this study include stochastic gradient descent (SGD), passive-aggressive (PA), gaussian naive bayes (GNB), and perceptron. These classifiers were chosen for their native support of 'partial_fit' in scikit-learn, computational efficiency in streaming settings, and complementary strengths: SGD/PA for linear separability, GNB for probabilistic robustness, and Perceptron for low-latency updates. Together, they cover diverse learning paradigms suitable for gradual data arrival.

The IL pseudocode is as follows:

Algorithm 1 Pseudocode of IL

Input: The dataset has been divided into initial and partial data.
Output: the accuracy of each method
For each method **do**
  Read the initial data
  If feature selection is True **then**
    Get the ranking of selected features
  Else-If PCA is True **then**
    Get the Principal Components (PCs)
Fit the model on the training data
  Get prediction on the testing data for all or selected or extracted features
For each partial data **do**
  Partial-fit the partial data on previous model for all or selected or extracted features
  Get prediction using partial data for all or selected or extracted features

The SGD Classifier is a linear classifier optimized by the SGD using stochastic or random probability such that one sample is chosen for model training in each cycle[56]. The proposed SGD optimizes the loss functions associated with certain classifiers. The loss function 'hinge' can be selected to emulate a linear SVM and another loss function 'log_loss' for Logistic Regression. The SGD's learning

rate is decayed over time to ensure convergence, while the hinge loss aggressively penalizes misclassified samples, emulating SVM behavior.

Likewise, the passive-aggressive classifier is a part of online learning algorithms, where the input data are expected to come in sequential order, and the model is updated incrementally. The 'passive' term refers to keeping the model unchanged if the prediction is correct, while 'aggressive' term relates to updating the model if the prediction is incorrect[57]. The PA classifier's C parameter is tuned to limit model updates when confidence is high, preventing overfitting to noisy batches.

The gaussian naive bayes classifier presents a probabilistic approach to classification that is a variation of the naive bayes classifier, specifically designed for use with continuous data. This classifier assumes that the features are independent of each other for a given class label. The Bayes theorem is used to calculate the probability for each new data point belonging to each class. This formula is derived from the multiplication of the likelihood and prior probabilities and then dividing by the available evidence[58]. GNB incrementally updates the class-conditional mean and variance statistics for each feature, leveraging Bayes' theorem without retraining from scratch.

Lastly, the perceptron is a single-ANN suitable for online learning due to its straightforward structure and updating rules, which makes it easy to adapt to incoming new data points. The perceptron's simplicity and efficient updating rule make it a more attractive choice for IL scenarios than multilayer perceptron[59]. The Perceptron's weights are adjusted via error-driven updates per sample, avoiding the computational overhead of backpropagation in deeper networks.

In addition, when feature selection/PCA is enabled, the same feature subspace (selected features/PCs) is propagated to partial data batches to ensure dimensional consistency. New features are projected onto the original PCA axes or filtered via precomputed rankings.

The evaluation metrics are tracked over time as the incoming partial is partially fitted to the existing model. To calculate the accuracy of each classifier, we measured the proportion of correctly predicted instances out of the total number of instances[57,58]. In addition, we compute the precision to measure the number of predicted positives that are truly positive, the recall to measure how many actual positives are captured, and the F1-Score as a harmonic mean of precision and recall. The precision, recall, and F1-score are all weighted by class size to ensure that larger classes influence the final metric more. Hence, they are robust to class imbalance. These metrics can be written as follows:

$$Accuracy = \sum_{i \in C} \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \quad (3)$$

$$Precision_{weighted} = \sum_{i \in C} \left(\frac{n_i}{n} \cdot \frac{TP_i}{TP_i + FP_i} \right) \quad (4)$$

$$Recall_{weighted} = \sum_{i \in C} \left(\frac{n_i}{n} \cdot \frac{TP_i}{TP_i + FN_i} \right) \quad (5)$$

$$F1_{weighted} = \sum_{i \in C} \left(\frac{n_i}{n} . 2. \frac{Precision_i . Recall_i}{Precision_i . Recall_i} \right) \quad (6)$$

where TP_i, TN_i, FP_i, and FN_i denote true positive, true negative, false positive, and false negative, respectively, for each class *i*.

To handle the possibility of concept drift (shifts in data distribution over time), our framework incorporates performance monitoring by tracking accuracy decay over successive batches, where a significant drop would trigger a drift alert. Upon detecting an accuracy drop that exceeds a threshold, such as 10-15%[60,61], the model is retrained on all accumulated data to capture the new concept. Afterward, incremental updates (using python's 'partial_fit') resume. Based on empirical guidelines for performance-based drift detection, we set the drift threshold to 10% accuracy decay[62,63]. This balances the sensitivity to significant drift while avoiding overfitting to noise.

Similarly, to detect potential sensor drift, change detection algorithms such as ADWIN (Adaptive Windowing) and Page-Hinkley are employed[64]. These methods are designed to monitor real-time data streams and identify any significant changes in the underlying distribution of sensor readings. ADWIN works by dynamically adjusting the size of its sliding window to detect statistically significant differences between sub-windows, making it suitable for environments with gradual or sudden drifts. Page-Hinkley, on the other hand, is particularly effective at detecting abrupt changes by tracking the cumulative deviation from the mean. Together, these techniques provide robust mechanisms for identifying and responding to shifts in sensor behavior, which is critical for maintaining the reliability and accuracy of data-driven systems.

3. Results and Discussion

Based on the experimental setup outlined in the previous section, this section describes the results of the sensor measurements, feature selection and extraction, and classification performances, as well as the use of a coolant monitoring system prototype.

3.1. Measurement of sensors

The sensor measurement patterns are shown in Figure 6 where the upper parts of the Figure are the photodiodes and the bottom parts are the gas sensors. Each of these measurement readings is labeled according to the specified categories in the previous section: class 0 is good category (blue highlight), class 1 is bad category (red highlight), and class 2 is very bad category (green highlight). Each legend

on the right section of the Figure signifies the sensor type and series.

The measurement unit for photodiodes is voltage (v), whereas the measurement unit for gas is Rs/Ro, which is the ratio between the initial and measured resistance. The photodiode measurements have means of approximately 0.29 V, standard deviations of approximately 0.0217 V, minimum value of 0.237, and maximum value of 0.345 V. The gas sensors have means ranging from 0.828 to 0.975 Rs/Ro, standard deviations between 0.0311 and 0.0689 Rs/Ro, minimum value of 0.237 Rs/Ro and maximum value of 1.101 Rs/Ro.

Initial observations of sensor features by class label indicate that Class 0 contains lower readings in both photodiode (FD05D) and MQ gas sensors, Class 1 shows distinctly higher values across all sensors, particularly in the MQ series (e.g., MQ135, MQ138), and Class 2 exhibits intermediate values, often overlapping with Class 0 in MQ sensors but trending toward Class 1 in FD05D sensors. Notably, FD05D sensors demonstrate minimal variance across classes, with slight shifts in the median. MQ sensors (e.g., MQ135, MQ138, MQ2-8) could be strong discriminators, showing clear class separation.

Boxplot diagram in Figure 7 highlights the distribution of each feature's readings, with photodiodes (e.g., FD05D sensors) and gas sensors (e.g., MQ series) exhibiting distinct value ranges. While FD05D sensors show tightly clustered values, gas sensors (e.g., MQ135) span wider ranges across classes. For MQ sensors, there is clear vertical separation between classes, with Class 1 boxes being distinctly higher. Conversely, FD05D sensors exhibit flatter boxes, emphasizing their weaker discriminative power compared to MQ sensors.

Figure 8 indicates that significant outliers across multiple sensor channels were detected, with the most prominent anomalies occurring in the FD05D_1300, FD05D_1450, and FD05D_1600 sensors, each containing 600+ outlier indices (e.g., indices 3239–6299). These outliers indicate systematic deviations, possibly due to sensor drift or environmental interference during prolonged operation. The MQ8 and MQ138 sensors exhibited fewer but clustered outliers (e.g., indices 913–946 and 910–946, respectively), indicating transient disturbances. Notably, sensors like MQ135 showed no outliers, confirming stable operation. The detection leveraged a hybrid standard deviation and IQR approach to capture both global deviations and localized anomalies.

ADWIN analysis confirmed the absence of abrupt drift across all sensors in sensor drift detection. However, the Page-Hinkley test identified gradual drift patterns in several sensors: MQ135 (3 drift points), MQ136 (2 points), MQ138 (2 points), MQ2 (1 point), MQ4 (2 points), and MQ6 (1 point). We applied a moving average filter to mitigate these gradual drifts, which stabilizes sensor readings without requiring model recalibration.

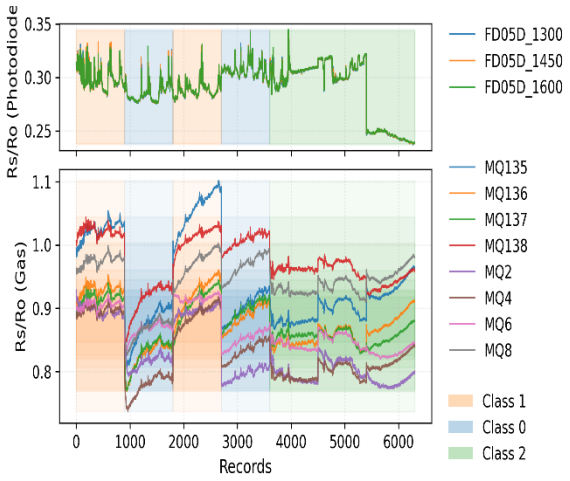


Fig. 6: Sensor measurement patterns

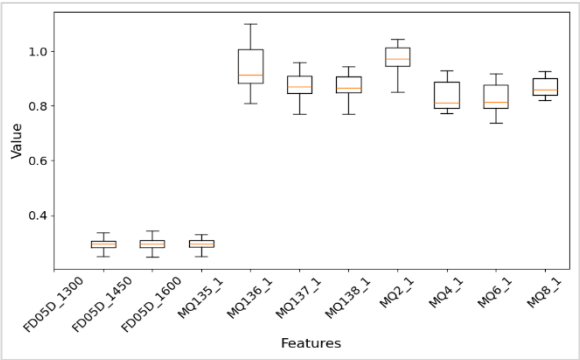


Fig. 7: Boxplot diagram of each feature

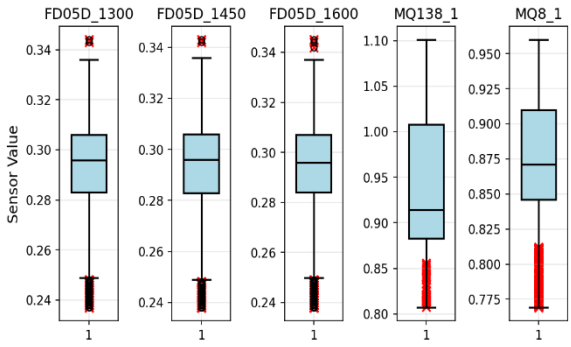


Fig. 8: Original distributions with outliers highlighted

Moreover, to detect the existence of concept drift, we observed that the performance decreases across the five partial datasets were all below 10% (ranging from 0.7% to 4.2%), with no single drop exceeding the commonly adopted drift threshold of 10-15% used in IL literature[60,61]. Concept drift is not statistically or practically significant for the current model and data stream.

3.2. Feature selection results

The feature selection results were obtained by computing the F-value scores for the best k features, where k is 1 to the total number of features, using the list of features shown in Figure 9. Each selected feature was run against

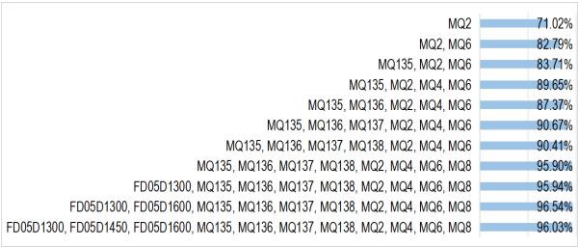


Fig. 9: Performance of the combined features

all considered methods to validate the selection result. The percentage of performance accuracy indicates that all features contributed to the classification process because all features achieved the highest performance average. However, the combination of eight gas sensors achieved 95.90% accuracy, whereas the three features of photodiode sensor only added small improvement of 0.13%. These eight features were selected during the comparison of classification performance.

3.3. PCA results

As with feature selection, the more Principal Components (PCs) considered, the better the classification performance. Figure 10 indicates that, generally, the accuracy values increase until the number of PCs reaches four, at which point the average performance reaches 90.79%. From this point onwards, performance improvement decreases. The variance threshold for PC retention is about 70- 85% to ensure that PCs can retain most of the information of the original variables[65]. Therefore, four PCs are selected for the classification performance comparison.

The optimal number of principal components (PCs) in was selected through a two-step process, namely: the variance retention threshold and classification accuracy plateau. PCs were retained until achieving 70–85% cumulative explained variance, a widely adopted threshold for balancing dimensionality reduction and information preservation[66]. In this study, the first two PCs captured 87.11% of the total variance, exceeding the minimum threshold. The fourth PC (97.54% variance) was included to ensure near-complete coverage of the original data's variability, aligning with industrial monitoring requirements where minor signal losses could impact coolant quality interpretation.

Furthermore, Figure 10 demonstrates that classification accuracy peaks at 90.79% with four PCs, beyond which marginal gains (<0.5%) do not justify added complexity. This aligns with the "elbow method" principle, where the optimal PC count lies at the point of diminishing returns[67]. Figure 11 shows the data groupings based on the number of PC, which shows two PCs where all data in the same class are clustered. The variances explained for the first four PCs are 60%, 87.11%, 96.59%, and 97.54%. Accordingly, two PCs explained the total variance in the original dataset by as much as 87.11%. Similarly, for the four PCs, 97.54% of the explained variance almost covers

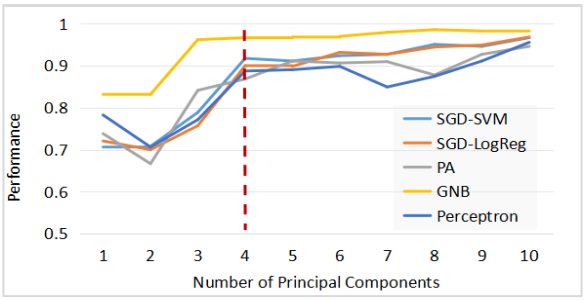


Fig. 10: Optimal PC number

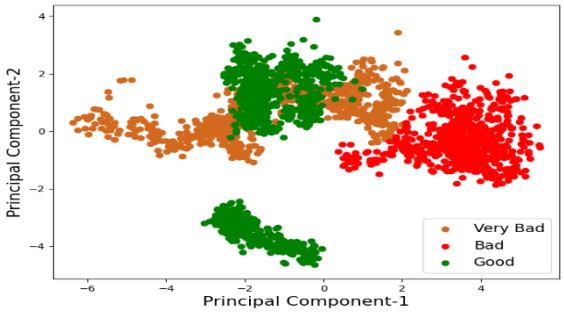


Fig. 11: Data grouping by two PCs

the total variance of the original data.

3.4. Classification

The IL classifiers employed during the experiment, namely, stochastic gradient descent (SGD), passive-aggressive (PA), gaussian naive bayes (GNB), and perceptron, were run against 11 features of olfactory and photodiode sensors. To assess the performance stability across different data splits, we run the algorithm 5 times with fixed random seeds (such as arbitrary integers of 101-105) for train-test partitioning. This ensures reproducibility while allowing variance measurement in the results.

Before the performance assessment, several parameters need to be fine-tuned. Parameter selection was performed using Python's RandomizedSearchCV run on 5-fold cross validation and 20 optimized parameters. The optimal number of optimized parameters (n_iter) was chosen based on several trials of n_iter values, ranging from 10 to 50. Figure 12 shows the average validation score plateaued beyond 'n_iter=20'. Further iterations yielded negligible improvement (<0.1%), suggesting that the hyperparameter space was sufficiently explored.

The parameters were chosen based on their documented impact on model performance, with initial ranges reflecting common practices in the literature[68,69]. For example, alpha values were selected to cover a range of regularization strengths, while learning_rate options included both fixed and adaptive strategies. Table 5 presents the optimized values, along with their initial ranges and descriptions. Notably, the absence of regularization (penalty: None) in the SGD classifiers indicates that the models benefited from the minimal constraints on the weight updates.

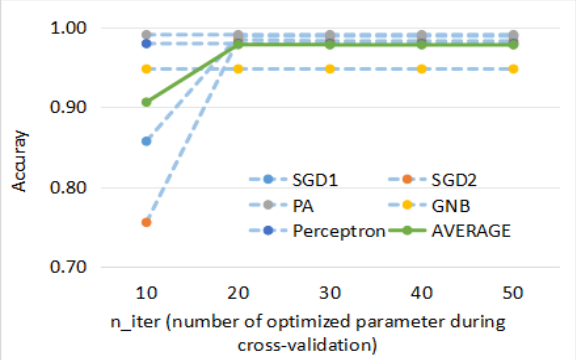


Fig. 12: Optimal number of optimized parameters for RandomizedSearchCV

Table 6 shows the classification accuracy of each classifier on the 11 original features, 8 selected features, and 4 extracted features for the initial and partial data. In this experiment, the initial data is randomly selected as half of the total dataset. The other half is evenly divided into 5 parts which would simulate the partial data.

The average accuracy values obtained by the classifiers were 96.68% for SGD-SVM, 96.88% for SGD-LogReg, 96.33% for PA, 95.07% for GNB, and 94.88% for perceptron. This average is determined by weighting the initial data at 50% and five partial data points at 10% each based on their respective sample sizes.. Hence, all algorithms performed quite well, and all algorithms scored accuracies greater than 90%. Impressive performances are shown by the classifiers on selected features, namely the eight olfactory sensors, which achieve 100% accuracy for SGD-SVM, SGD-LogReg, PA, and perceptron.

For comparison, we conducted a quantitative evaluation against traditional manual methods (visual, tactile, and olfactory inspection) performed by six experienced operators on seven coolant samples. The operators' average accuracy was 36%, whereas our system achieved over 95% detection accuracy. These results strongly support the system's potential to replace human manual inspection.

Table 5: Optimized parameters of each classifier

Classifier	Parameters	Initial Values	Optimized Values
SGD1	penalty	'l2', 'l1', None	None
	learning_rate	'constant', 'optimal', 'adaptive'	adaptive
	eta0	0, 0.01, 0.1	0.1
	alpha	0, 0.0001, 0.001, 0.01	0.001
SGD2	penalty	'l2', 'l1', None	None
	learning_rate	'constant', 'optimal', 'adaptive'	adaptive
	eta0	0, 0.01, 0.1	0.1
	alpha	0, 0.0001, 0.001, 0.01	0.001
PA	loss	'hinge', 'squared_hinge'	squared_hinge
	C	0.01, 0.1, 1	0.1
GNB	var_smoothing	1e-10, 1e-9, 1e-8	1,00E-10
Perceptron	penalty	'l2', 'l1'	l1
	alpha	0.00001, 0.0001, 0.001	1,00E-05

Table 6: Performance accuracy of the algorithms on the initial and the incoming data based on the original, selected, and extracted features

Feature	Data	SGD-SVM	SGD-LogReg	Passive-Aggr	Naïve Bayes	Perceptron
Original Features	Initial	0.9743	0.9749	0.9759	0.9460	0.9549
	Part-1	0.9444	0.9571	0.9460	0.9571	0.9127
	Part-2	0.9508	0.9587	0.9508	0.9571	0.9476
	Part-3	0.9508	0.9556	0.9476	0.9460	0.9270
	Part-4	0.9556	0.9651	0.9444	0.9635	0.9444
	Part-5	0.9508	0.9603	0.9492	0.9508	0.9413
Feature Selection	Initial	1.0000	0.9994	0.9994	0.9562	1.0000
	Part-1	0.9619	0.9651	0.9571	0.9429	0.9095
	Part-2	0.9651	0.9683	0.9540	0.9540	0.9476
	Part-3	0.9635	0.9683	0.9619	0.9270	0.9270
	Part-4	0.9698	0.9730	0.9635	0.9444	0.9365
	Part-5	0.9730	0.9794	0.9571	0.9381	0.9619
PCA	Initial	0.9756	0.9746	0.9724	0.9717	0.9619
	Part-1	0.9365	0.9349	0.9222	0.9127	0.8984
	Part-2	0.9222	0.9286	0.9317	0.9286	0.9143
	Part-3	0.9349	0.9302	0.9206	0.9540	0.9016
	Part-4	0.9413	0.9381	0.9254	0.9635	0.9079
	Part-5	0.9333	0.9381	0.9286	0.9127	0.9032

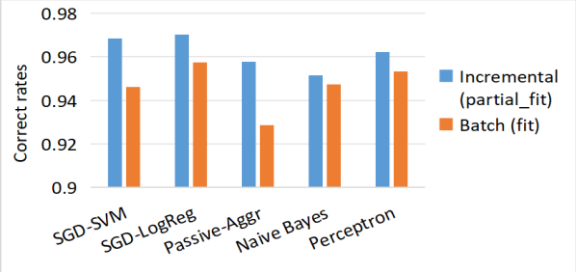


Fig. 13: Comparison of the accuracy of the algorithms using incremental and non-incremental updates of the previous model

Table 7: Performance accuracies among classifiers across different metrics

Classifier	Precision (Mean ± SD)	Recall (Mean ± SD)	F1-Score (Mean ± SD)	Accuracy (Mean ± SD)
SGD-SVM	0.9625 ± 0.0100	0.9624 ± 0.0099	0.9623 ± 0.0100	0.9624 ± 0.0099
SGD-LogReg	0.9672 ± 0.0074	0.9671 ± 0.0073	0.9671 ± 0.0073	0.9671 ± 0.0073
PA	0.9623 ± 0.0108	0.9617 ± 0.0108	0.9615 ± 0.0109	0.9617 ± 0.0108
GNB	0.9508 ± 0.0084	0.9505 ± 0.0084	0.9505 ± 0.0084	0.9505 ± 0.0084
Perceptron	0.9459 ± 0.0167	0.9448 ± 0.0172	0.9444 ± 0.0176	0.9448 ± 0.0172

Likewise, on average, PCA helps algorithms yield good performance for the initial dataset, which is above 96% for SGD-SVM, SGD-LogReg, and PA. The algorithms performed slightly worse on 4 PCs compared to the 8 selected features. However, when the number of PCs was increased to 8, their performance was on par (97.64% for 8 PCs vs. 97.29% for 8 selected features).

Performance generally drops slightly with partial data in IL, a consequence of avoiding full data training. Nonetheless, these decreases, which range from 0.7% to 4.2%, are acceptable[60,61] and do not signify concept drift, thereby negating the need for full data retraining.

As shown in Figure 13, IL allows the model to adapt to new data, achieving an accuracy up to 1.55% higher than that obtained by simply using the initial, static model over time. IL enables continuous model adaptation by updating parameters with new data batches via scikit-learn's 'partial_fit' function. This differs from the traditional batch mode, where the 'fit' function trains the model once on a static dataset. Therefore, IL is effective, especially when the quality of the incoming labeled partial data consistently exceeds that of the initial data.

Using various metrics, Table 7 indicates that across runs on the original features, SGD-LogReg achieved the highest mean F1-score (0.9671 ± 0.0073) across runs on the original features with the lowest variability (standard deviation/SD) among all classifiers, suggesting robust reproducibility. Passive Aggressive (PA) and SGD-SVM

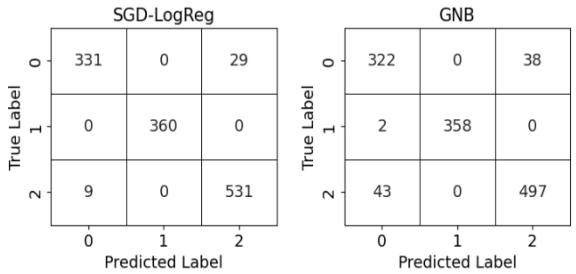


Fig. 14: Example of confusion matrices

Table 8: Performance accuracy among classes

Class	Precision	Recall	F1-score	Support
0	0.9392	0.9110	0.9244	360
1	0.9989	0.9991	0.9990	360
2	0.9426	0.9603	0.9511	540

showed comparable performance (F1-score of ~0.962), though PA's marginally higher standard deviation (±0.0109) indicates slightly less stability. Gaussian Naïve Bayes (GNB) underperformed (F1-score of 0.9505 ± 0.0084), consistent with its sensitivity to feature dependencies. The Perceptron's larger variability (F1's SD of ±0.0176) highlights its known limitations in convergence stability[70].

Figure 14 shows the confusion matrices obtained from the two classifiers. Based on the number of true and predicted labels from both Figures, we can compute their precision, recall, F1-score, and accuracy. With a precision of 0.9702, SGD-LogReg makes fewer false positives (better at avoiding incorrect predictions) than GNB whose precision is 0.9345. Similarly, SGD-LogReg with a recall of 0.9698 misses fewer true positives (better at capturing all relevant cases) as opposed to GNB with a precision of 0.9341. SGD-LogReg also balances precision/recall more effectively (harmonic mean) compared to GNB as their F1-scores are 0.9697 and 0.9343, respectively. In addition, the accuracies of SGD and GNB (0.9698 and 0.9341, respectively) align with the F1-score, suggesting that both models perform similarly relative to class distribution.

Moreover, Class 1 has nearly perfect precision and recall, which indicates that its features could well separate it from other classes (Table 8). Meanwhile, Class 0, which has the lowest precision and recall, tends to avoid false positives but misses true positives. Lastly, Class 2 has higher recall (96%) but lower precision (94.3%), indicating that the model overpredict Class 2 (for instance, labels Class 0/1 samples as Class 2). In addition, despite Class 2's larger test sample size (540 as opposed to 360 for other Classes), we mitigate imbalance effects using weighted precision/recall, which scales metrics by class support.

Figure 15 compares the F1-scores (mean ± standard deviation) of five classifiers (SGD-SVM, SGD-LogReg, PA, GNB, and Perceptron) across three data treatments: original, outlier-treated (using winsorization), and drift-

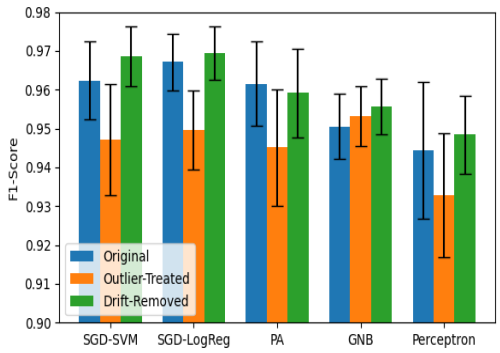


Fig. 15: Classifier performance across data treatments

removed (using moving average). The grouped bar chart highlights that the drift correction consistently improved the performance of most classifiers, with SGD-LogReg achieving the highest F1-score (0.9695 ± 0.0069). However, outlier treatment led to slight degradation compared to the original data, except for GNB, which saw marginal gains. Error bars (standard deviations) indicate that drift removal also reduced variability, indicating enhanced robustness. The Perceptron, while the least performant, benefited the most from drift correction, narrowing its performance gap with other models. Recent studies[71,72] have demonstrated that outlier treatment does not always improve accuracy and can degrade performance if extreme values contain valuable signals. Our results are consistent with these findings, as the original data yielded better metrics.

3.5. Testing results

To simulate a client bringing a coolant sample to test its quality, we developed a web interface to read the coolant condition using our sensors (Figure 16). The sensor data can be directly stored in and read from the system's database or stored in offline media and read by file upload. The testing data are run against the previously compiled model. If the data contain no label, the class of the tested coolant can be given, either 'very bad', 'bad', or 'good' without testing accuracy.If the label of the testing data can be obtained by submitting it to relevant laboratories, it can be used as a partial dataset for IL.

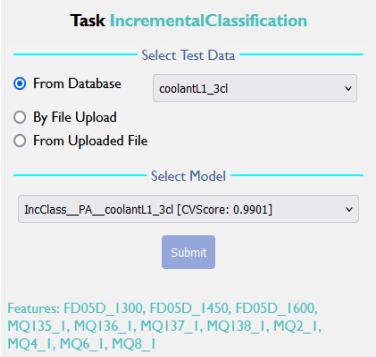
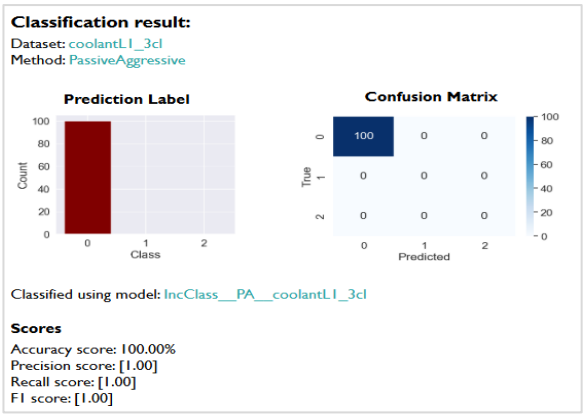
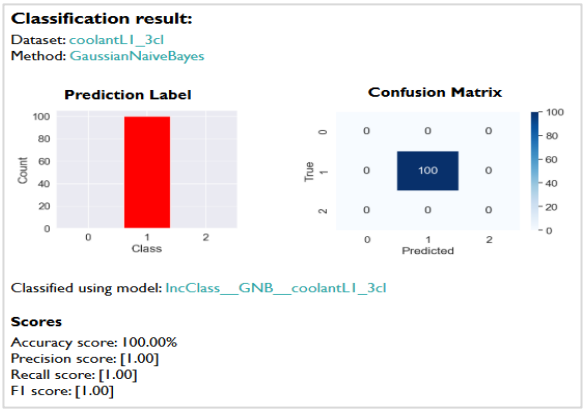


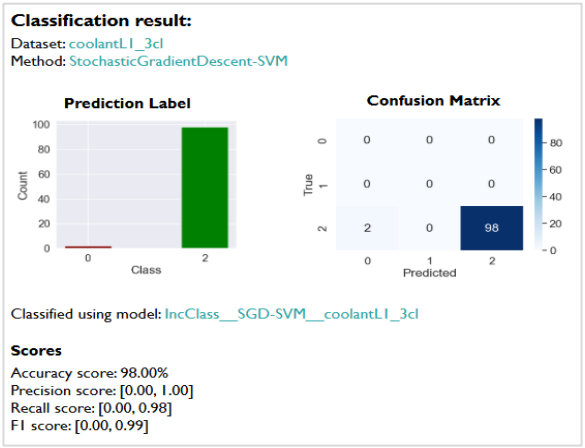
Fig. 16: Input for testing data and the previously built coolant quality model



(a). Prediction result for class '0' (very bad)



(b). Prediction result for class '1' (bad)



(c). Prediction result for class '2' (good)

Fig. 17: Classification results of three labeled testing data

The prediction results of different sets of testing data along with its confusion matrix when the data contain labels are shown in Figure 17. The first testing data is 100 records of "very bad" coolant, which is accurately predicted by the chosen model. Likewise, the 100 records testing data were accurately predicted as 'bad' coolant. Lastly, the 100 records of the third testing data are 98% accurately predicted as 'good' and 2% inaccurately predicted as 'very bad' coolant. Therefore, the proposed interface demonstrates the possibility of its implementation in real

circumstances where the coolant quality can be predicted quickly with relatively high accuracy.

In terms of real-time processing capabilities, our proposed system can achieve fast detection through lightweight incremental algorithms and efficient feature handling. The system processes sensor data in near real-time by employing IL classifiers. Unlike batch processing, IL via 'partial_fit' (Figure 13) enables continuous updates without reprocessing historical data, which critical for rapid decision-making.

The web prototype's prediction (Figure 14) confirms operational readiness for real-time use. Similarly, using only 8 dominant gas sensors (selected via F-value scoring) minimizes computational overhead while maintaining accuracy, enabling faster inference.

For industrial deployment, the proposed system may scale across multiple CNC machines using a distributed IoT framework and resource-efficient learning algorithms. Each machine's sensors could act as independent nodes transmitting data to a central hub, which aggregates information for incremental model updates, avoiding single-point bottlenecks. Additionally, ML algorithms like passive-aggressive (PA) and SGD, require minimal memory, allowing concurrent operation on multiple machines without server overload[72].

3.6. Study limitation

Although this study demonstrates promising results, several limitations must be acknowledged. First, the dataset, although substantial with 6,300 records from seven coolant samples, may not fully represent the diverse conditions encountered in real-world industrial settings. Manufacturing environments vary significantly in terms of coolant types, machine usage patterns, and contamination levels, which could influence sensor readings and model performance. For instance, the study's samples were collected under controlled conditions, potentially omitting edge cases such as highly contaminated coolants or mixtures with unconventional additives. Thus, although the model performs well on the available data, its generalizability to untested scenarios remains uncertain. Further validation across a wider range of operational contexts, including different CNC machines, coolant brands, usage time, and environmental conditions, would strengthen the system's reliability for broader industrial adoption.

The second limitation is the long-term robustness of the sensor system. Although drift mitigation techniques (using moving average filters) were effective in stabilizing short-term sensor fluctuations, the study did not explicitly evaluate prolonged sensor degradation or extreme environmental interference. For example, temperature fluctuations, humidity, or exposure to chemical vapors in industrial settings could gradually alter sensor calibration, leading to performance decay over time. The impact of sensor aging, such as reduced sensitivity in gas sensors after extended use, was not addressed. These factors are critical for real-world deployment, where maintenance cycles and recalibration requirements must be clearly defined. Future work should incorporate accelerated aging tests and environmental stress experiments to assess the system's durability and refine maintenance protocols accordingly.

4. Conclusions

This study introduced an adaptive monitoring framework for automated CNC coolant quality detection, which combines photodiode and gas sensors, IoT-enabled data collection, and IL algorithms. The system adheres to standardized metrics (SNI 06-6989:2004 and SNI/ISO 4833-1:2015) to classify coolant into three quality levels: very bad, bad, and good. This research advances both theoretical and practical domains of industrial AI, offering scalable solutions for real-time monitoring and predictive maintenance.

Our work makes significant theoretical contributions by demonstrating the effectiveness of IL algorithms (such as SGD-SVM, SGD-LogReg) in handling dynamic sensor data. The hybrid outlier detection method, which combines standard deviation and IQR with drift mitigation techniques (using moving average filters), provides a replicable framework for robust IoT-based systems in noisy industrial environments. Notably, the feature selection method that selects eight gas sensors (MQ135, MQ136, MQ137, MQ138, MQ2, MQ4, MQ6, and MQ8) as well as PCA may improve the average performance of most classifiers. The system achieves high predictive accuracy (up to an average of 97%), enabling manufacturers to replace costly lab tests with rapid, reliable AI assessments. The developed web prototype further bridges theory and application, offering a user-friendly interface for real-time coolant quality prediction and seamless model updates through IL.

The proposed system delivers measurable benefits to manufacturing operations. First, it reduces costs by reducing the reliance on manual inspections and laboratory analyses. Second, its IoT-integrated design supports scalable deployment, allowing real-time monitoring of CNC machines to detect coolant degradation early, thereby optimizing maintenance schedules and reducing downtime. Third, the system's IL capability ensures adaptability to new data patterns, addressing challenges like sensor drift or changing environmental conditions without requiring full model retraining. These advantages align with Industry 4.0 goals, emphasizing automation, efficiency, and data-driven decision-making.

Despite its strengths, this study has some limitations. The dataset, while substantial (6,300 records from seven coolant samples), may not capture the full variability of

industrial conditions, indicating the need for validation across broader operational contexts. Additionally, while drift mitigation techniques were effective, long-term sensor degradation or extreme environmental interference (for instance, temperature fluctuations) were not explicitly tested.

We propose several key directions for future research to build on this work. First, exploring advanced algorithms such as deep learning (e.g., LSTMs) or ensemble methods, could enhance anomaly detection in nonlinear sensor data patterns. Second, real-world deployment trials would validate the system's resilience to hardware failures, multisensor fusion challenges, and unpredictable noise. Third, integrating explainable AI techniques (such as SHAP values) could improve user trust by allowing non-technical operators to interpret model decisions. These steps would transition the system from a prototype to a comprehensive tool for smart manufacturing predictive maintenance.

Acknowledgements

This research was supported by LPDP of Indonesia under-Contract No.B 802/II.7.5/FR/6/2022.

Data Availability

The research data are available upon request. Contact the first author of the article to request the data.

References

1) A. Chandra, A. Yadav, S. Singh, and P.K. Arora, "Optimisation of machining parameters for cnc milling of fibre reinforced polymers," Evergreen, (2023). doi:10.5109/6792826.
2) M. Kuruc, T. Vopát, J. Peterka, M. Necpal, V. Šimna, J. Milde, and F. Jurina, "The influence of cutting parameters on plastic deformation and chip compression during the turning of c45 medium carbon steel and 62simncr4 tool steel," Materials (Basel)., (2022). doi:10.3390/ma15020585.
3) A.R. Suryaningrat, Nasril, A. Musthofa, H. Purnama, M. Ardhany, and M. Al Huda, "Optimization of dynamic characteristics of boring bar for chatter mitigation in boring process," AIP Conf. Proc., 3069 (1) 20032 (2024). doi:10.1063/5.0206651.
4) M.S.H. Ador, S. Kabir, F. Ahmed, F. Ahmad, and S. Adil, "Effects of minimum quantity lubrication (mql) on surface roughness in milling al alloy 383 / adc 12 using nano hybrid cutting fluid," Evergreen, (2022). doi:10.5109/6625790.
5) A.M. Sabri, N. Talib, A.S.A. Sani, S. Kunar, and K. Kamdani, "Investigation of modified rbd palm oil-based hybrid nanofluids as metalworking fluid," Evergreen, 11 (2) 797–805 (2024). doi:10.5109/7183360.
6) S. Roy, R. Kumar, A.K. Sahoo, and R.K. Das, "A brief review on effects of conventional and nano particle based machining fluid on machining performance of minimum quantity lubrication machining," in: Mater. Today Proc., 2019. doi:10.1016/j.matpr.2019.07.571.
7) H. Wang, T. Reponen, S.A. Lee, E. White, and S.A. Grinshpun, "Size distribution of airborne mist and endotoxin-containing particles in metalworking fluid environments," J. Occup. Environ. Hyg., (2007). doi:10.1080/15459620601144883.
8) J. Józwik, M. Zawada-Michałowska, G. Budzik, S. Legutko, and M. Kupczyk, "Microbiological analysis of coolant used in machining," Adv. Sci. Technol. Res. J., (2023). doi:10.12913/22998624/157421.
9) F.E. Mirer, "New evidence on the health hazards and control of metalworking fluids since completion of the osha advisory committee report," Am. J. Ind. Med., 53 (8) 792–801 (2010). doi:10.1002/ajim.20853.
10) Y. Kondo, M. Yamaguchi, S. Sakamoto, and K. Yamagchi, "A dynamic observation concept to keep water-soluble coolant in normal condition for long-time," in: Adv. Mater. Res., 2013. doi:10.4028/www.scientific.net/AMR.652-654.2119.
11) H.M. Liu, Y.H. Lin, M.Y. Tsai, and W.H. Lin, "Occurrence and characterization of culturable bacteria and fungi in metalworking environments," Aerobiologia (Bologna)., (2010). doi:10.1007/s10453-010-9169-8.
12) R.T. Chander, P.S. Chakraborty, S. Nallusamy, and V.S. Hariharan, "Precision maintenance strategies : understanding and addressing coolant oil contamination ' s influence on grinding machine bearing integrity," 45 (2) 4465–4479 (2024). https://doi.org/10.52783/tjjpt.v45.i02.6664.
13) K. Li, F. Aghazadeh, S. Hatipkarasulu, and T.G. Ray, "Health risks from exposure to metal-working fluids in machining and grinding operations," Int. J. Occup. Saf. Ergon., (2003). doi:10.1080/10803548.2003.11076555.
14) J.P. Byers, "Metalworking fluids (2nd Ed.)," 2006. doi:10.1201/9781420017731.
15) M. Opachak, "Industrial Fluids: Controls, Concerns and Costs," Society of Manufacturing Engineers, Marketing Services Department, 1982.
16) F.J. Passman, "Metalworking fluid microbes—what we need to know to successfully understand cause-and-effect relationships," Tribol. Trans., 51 (1) 107–117 (2008). doi:10.1080/10402000701691720.
17) Y. Kondo, and Y. Miyake, "A study on anomaly detection of water-soluble coolant using internal-sensors," Int. J. Autom. Technol., (2022). doi:10.20965/ijat.2022.p0175.

18) G.E. Adjovu, H. Stephen, D. James, and S. Ahmad, "Measurement of total dissolved solids and total suspended solids in water systems: a review of the issues, conventional, and remote sensing techniques," Remote Sens., (2023). doi:10.3390/rs15143534.
19) M. Veillette, P.S. Thorne, T. Gordon, and C. Duchaine, "Six month tracking of microbial growth in a metalworking fluid after system cleaning and recharging," Ann. Occup. Hyg., (2004). doi:10.1093/annhyg/meh043.
20) N. Bulatov, A. Uvaliyeva, K. Kassymzhanova, M. Izteleuova, and I. Saukenova, "Intelligent systems for managing and monitoring the collection, sorting, and transportation of solid waste for processing," Evergreen, 11 (2) 938–948 (2024). doi:10.5109/7183376.
21) Y. Duan, J.S. Edwards, and Y.K. Dwivedi, "Artificial intelligence for decision making in the era of big data – evolution, challenges and research agenda," Int. J. Inf. Manage., 48 63–71 (2019). doi:10.1016/j.ijinfomgt.2019.01.021.
22) L. Erhan, M. Ndubuaku, M. Di Mauro, W. Song, M. Chen, G. Fortino, O. Bagdasar, and A. Liotta, "Smart anomaly detection in sensor systems: a multi-perspective review," Inf. Fusion, 67 64–79 (2021). doi:https://doi.org/10.1016/j.inffus.2020.10.001.
23) P. Visconti, G. Rausa, C. Del-Valle-Soto, R. Velázquez, D. Cafagna, and R. De Fazio, "Machine learning and iot-based solutions in industrial applications for smart manufacturing: a critical review," Futur. Internet, 16 (11) (2024). doi:10.3390/fi16110394.
24) D. Singh, and A. Singh, "Role of building automation technology in creating a smart and sustainable built environment," Evergreen, (2023). doi:10.5109/6781101.
25) M.S. Sumathi, J. Shruthi, V. Jain, G.K. Kumar, and Z.Z. Khan, "Using artificial intelligence (ai) and internet of things (iot) for improving network security by hybrid cryptography approach," Evergreen, (2023). doi:10.5109/6793674.
26) F. Jurina, J. Peterka, M. Vozar, B. Patoprsty, and T. Vopat, "Design of real-time monitoring system for cutting fluids," Designs, (2023). doi:10.3390/designs7020043.
27) R. Fajar, K.F.A. Sukra, F. Hidiyanto, N.S. Octaviani, and D.A. Sugeng, "A data-driven machine learning approach to identify end-of-life vehicles in indonesia," Evergreen, 11 (3) 2504–2514 (2024). doi:10.5109/7236891.
28) R. Huerta-Soto, F. Francis, M. Asís-López, and J. Panduro-Ramirez, "Implementation of machine learning in supply chain management process for sustainable development by multiple regression analysis approach (mraa)," Evergreen, (2023). doi:10.5109/6793671.
29) Y. Luo, L. Yin, W. Bai, and K. Mao, "An appraisal of incremental learning methods," Entropy, (2020). doi:10.3390/e22111190.
30) D. Nallaperuma, R. Nawaratne, T. Bandaragoda, A. Adikari, S. Nguyen, T. Kempitiya, D. De Silva, D. Alahakoon, and D. Pothuhera, "Online incremental machine learning platform for big data-driven smart traffic management," IEEE Trans. Intell. Transp. Syst., (2019). doi:10.1109/TITS.2019.2924883.
31) B. Yu, L. Fang, Z. Wu, C. Shen, and X. Hu, "A study on short-term air consumption prediction model for air-jet looms combining sliding time window and incremental learning," Energies, 17 (16) (2024). doi:10.3390/en17164052.
32) H.A. Gabbar, O.G. Adegboro, A. Chahid, and J. Ren, "Incremental learning-based algorithm for anomaly detection using computed tomography data," Computation, (2023). doi:10.3390/computation11070139.
33) M. Sakthivel, and P. Pandiyan, "Stochastic modelling on new mixture distribution with properties and their application," Int. J. Math. Stat. Comput. Sci., 2 (SE-Original Research Articles) 259–275 (2024). doi:10.59543/ijmscs.v2i.9015.
34) D. Ketseas, "Stochastic response of an airfoil and its effects on lco's behavior under stall flutter regime," Int. J. Math. Stat. Comput. Sci., 2 (SE-Original Research Articles) 168–172 (2024). doi:10.59543/ijmscs.v2i.8663.
35) E. Belouadah, A. Popescu, and I. Kanellos, "A comprehensive study of class incremental learning algorithms for visual tasks," Neural Networks, (2021). doi:10.1016/j.neunet.2020.12.003.
36) J. Hu, C. Yan, X. Liu, Z. Li, C. Ren, J. Zhang, D. Peng, and Y. Yang, "An integrated classification model for incremental learning," Multimed. Tools Appl., (2021). doi:10.1007/s11042-020-10070-w.
37) B. Nemade, and D. Shah, "An efficient iot based prediction system for classification of water using novel adaptive incremental learning framework," J. King Saud Univ. - Comput. Inf. Sci., 34 (8, Part A) 5121–5131 (2022). doi:https://doi.org/10.1016/j.jksuci.2022.01.009.
38) C. Monday, M.S. Zaghloul, D. Krishnamurthy, and G. Achari, "Incremental machine learning and genetic algorithm for optimization and dynamic aeration control in wastewater treatment plants," J. Water Process Eng., 69 106600 (2025). doi:https://doi.org/10.1016/j.jwpe.2024.106600.
39) J. Huang, "Organic photodetectors to monitor water contaminants," Nat. Water, 2 (6) 505–506 (2024). doi:10.1038/s44221-024-00259-w.
40) I.A. Siahaan, G.A. Mutiara, and M.I. Sani, "A Low-Cost Water Quality Monitoring Based on Photodiode

and LDR," in: 2021 IEEE Asia Pacific Conf. Wirel. Mob., 2021: pp. 141–146. doi:10.1109/APWiMob51111.2021.9435280.

41) C.D. Fay, and A. Nattestad, "Advances in optical based turbidity sensing using led photometry (pedd)," Sensors, 22 (1) (2022). doi:10.3390/s22010254.
42) Winsen, "MQ-2 flammable gas sensor, rev. 1.4," 7 (2015). https://www.winsen-sensor.com/d/files/PDF/Semiconductor Gas Sensor/MQ-2 (Ver1.4) - Manual.pdf (accessed February 18, 2025).
43) Thorlabs, "FD05D - ingaas photodiode, 17 ns rise time, 900-2600 nm, ø0.5 mm active area," 1 (2014). https://www.thorlabs.com/thorproduct.cfm?partnumber=FD05D (accessed February 18, 2025).
44) M. Henschke, X. Wei, and X. Zhang, "Data Visualization for Wireless Sensor Networks Using ThingsBoard," in: 2020 29th Wirel. Opt. Commun. Conf. WOCC 2020, 2020. doi:10.1109/WOCC48579.2020.9114929.
45) V. Patel, A. Kapoor, A. Sharma, and S. Chakrabarti, "Taxonomy of outlier detection methods for power system measurements," Energy Convers. Econ., 4 (2) 73–88 (2023). doi:https://doi.org/10.1049/enc2.12082.
46) C.S.K. Dash, A.K. Behera, S. Dehuri, and A. Ghosh, "An outliers detection and elimination framework in classification task of data mining," Decis. Anal. J., 6 100164 (2023). doi:https://doi.org/10.1016/j.dajour.2023.100164.
47) A. Abuzaid, and I. Alkrunz, "A comparative study on univariate outlier winsorization methods in data science context," Stat. Appl., 36 (1) 85–99 (2024). doi:10.26398/IJAS.0036-004.
48) S. Alelyani, J. Tang, and H. Liu, "Feature Selection for Clustering: A Review," in: Data Clust., Chapman and Hall/CRC, 2018: pp. 29–60. doi:10.1201/9781315373515-2.
49) U. Stańczyk, "Feature evaluation by filter, wrapper and embedded approaches," Stud. Comput. Intell., (2015). doi:10.1007/978-3-662-45620-0_3.
50) M. Ardhany, A.C. Zuhri, A. Widodo, A.R. Suryaningrat, G.I. Islami, A. Musthofa, Nasril, D.M. Gandana, S.A. Kunharyanto, and R. Mayasari, "Early detection of motor vehicle exhaust gas using a gas sensor array with multiple kernel learning," Evergreen, 11 (3) 2678–2690 (2024). doi:10.5109/7236907.
51) Y. Meng, S.N. Qasem, M. Shokri, and S. S, "Dimension reduction of machine learning-based forecasting models employing principal component analysis," Mathematics, 8 (8) (2020). doi:10.3390/math8081233.
52) Z. Rizgar, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction," J. Appl. Sci. Technol. Trends, 1 (1) 56–70 (2020). doi:10.38094/jastt1224.
53) Q. Jiang, X. Yan, and B. Huang, "Performance-driven distributed pca process monitoring based on fault-relevant variable selection and bayesian inference," IEEE Trans. Ind. Electron., 63 (1) 377–386 (2016). doi:10.1109/TIE.2015.2466557.
54) R. Bro, and A.K. Smilde, "Principal component analysis," Anal. Methods, (2014). doi:10.1039/c3ay41907j.
55) G. Zhu, and Q. Dai, "EnsPKDE&IncLKDE: a hybrid time series prediction algorithm integrating dynamic ensemble pruning, incremental learning, and kernel density estimation," Appl. Intell., (2021). doi:10.1007/s10489-020-01802-4.
56) Z. Tan, J. Chen, Q. Kang, M. Zhou, A. Abusorrah, and K. Sedraoui, "Dynamic embedding projection-gated convolutional neural networks for text classification," IEEE Trans. Neural Networks Learn. Syst., (2022). doi:10.1109/TNNLS.2020.3036192.
57) R.H. Al-Furaiji, and H. Abdulkader, "A comparison of the performance of six machine learning algorithms for fake news," EAI Endorsed Trans. AI Robot., 3 (SE-Research article) (2024). doi:10.4108/airo.4153.
58) S. Naiem, A.E. Khedr, A.M. Idrees, and M.I. Marie, "Enhancing the efficiency of gaussian naïve bayes machine learning classifier in the detection of ddos in cloud computing," IEEE Access, (2023). doi:10.1109/ACCESS.2023.3328951.
59) S. Morita, H. Iguchi, and T. Hoya, "A Class Incremental Learning Algorithm for a Compact-Sized Probabilistic Neural Network and Its Empirical Comparison with Multilayered Perceptron Neural Networks," in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2023. doi:10.1007/978-3-031-47634-1_22.
60) J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: a review," IEEE Trans. Knowl. Data Eng., 31 (12) 2346–2363 (2019). doi:10.1109/TKDE.2018.2876857.
61) R.S.M. Barros, D.R.L. Cabral, P.M. Gonçalves, and S.G.T.C. Santos, "RDDM: reactive drift detection method," Expert Syst. Appl., 90 344–355 (2017). doi:https://doi.org/10.1016/j.eswa.2017.08.023.
62) J. Gama, I. Žliobaitundefined, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," ACM Comput. Surv., 46 (4) (2014). doi:10.1145/2523813.
63) W.X. Zhao, N. Zhou, W. Zhang, J.-R. Wen, S. Wang, and E.Y. Chang, "A probabilistic lifestyle-based trajectory model for social strength inference from

human trajectory data," ACM Trans. Inf. Syst., 35 (1) (2016). doi:10.1145/2948064.

64) R. Gavaldà, "Adaptive Windowing BT - Encyclopedia of Big Data Technologies," in: S. Sakr, A. Zomaya (Eds.), Springer International Publishing, Cham, 2018: pp. 1–6. doi:10.1007/978-3-319-63962-8_194-1.

65) P. Geladi, and J. Linderholm, "2.03 - Principal Component Analysis☆," in: S. Brown, R. Tauler, B. Walczak (Eds.), Compr. Chemom. (Second Ed., Second Edi, Elsevier, Oxford, 2020: pp. 17 − 37. doi:https://doi.org/10.1016/B978-0-12-409547-2.14892-9.

66) İ.H. Gümüş, C. Karakuzulu, S. Güldal, and M. Yavaş, "Determining the number of principal components with schur's theorem in principal component analysis," Bitlis Eren Üniversitesi Fen Bilim. Derg., 12 (2) 299–306 (2023). doi:10.17798/bitlisfen.1144360.

67) E. Morgado, L. Martino, and R.S. Millán-Castillo, "Universal and automatic elbow detection for learning the effective number of components in model selection problems," Digit. Signal Process., 140 104103 (2023). doi:https://doi.org/10.1016/j.dsp.2023.104103.

68) B. Bischl, M. Binder, M. Lang, T. Pielok, J. Richter, S. Coors, J. Thomas, T. Ullmann, M. Becker, A.-L. Boulesteix, D. Deng, and M. Lindauer, "Hyperparameter optimization: foundations, algorithms, best practices, and open challenges," WIREs Data Min. Knowl. Discov., 13 (2) e1484 (2023). doi:https://doi.org/10.1002/widm.1484.

69) M. Feurer, and F. Hutter, "Hyperparameter Optimization BT - Automated Machine Learning: Methods, Systems, Challenges," in: F. Hutter, L. Kotthoff, J. Vanschoren (Eds.), Springer International Publishing, Cham, 2019: pp. 3–33. doi:10.1007/978-3-030-05318-5_1.

70) J. Frankle, and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," in: 7th Int. Conf. Learn. Represent. ICLR 2019, 2019.

71) S. Raschka, Y. (Hayden) Liu, V. Mirjalili, and D. Dzhulgakov, "Machine Learning with PyTorch and Scikit-Learn: Develop machine learning and deep learning models with Python," Packt Publishing, 2022. http://ieeexplore.ieee.org/document/10162164.

72) A. Géron, "Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems," O'Reilly Media, Incorporated, 2019. https://books.google.co.id/books?id=OCS1twEACAAJ.