S & M 3613

Dual-stage Classification Framework for Detecting Rare or Unseen Patterns Based on Novelty Detection and Supervised Learning

Natthakritta Rungtalay and Somyot Kaitwanidvilai*

School of Engineering, King Mongkut's Institute of Technology Ladkrabang,
Bangkok 10520, Thailand

(Received March 1, 2024; accepted April 10, 2024)

Keywords: novelty detection, supervised learning, fault detection, imbalanced dataset, machine learning, continuously changing environments

In this article, we propose a dual-stage classification framework designed for identifying rare or unseen patterns in the hard disk drive (HDD) industrial test process. The proposed framework integrates novelty detection and supervised learning methodologies to effectively address the challenges associated with imbalanced datasets and the detection of infrequent or unseen patterns within continuously changing environments. By employing novelty detection as the first-stage classifier followed by supervised learning as the second-stage classifier, the proposed method demonstrates an increased capacity to adapt to fluctuating environments, consequently enhancing the overall accuracy of process classification in practical manufacturing settings. To strengthen the robustness of novelty detection methods, an ensemble model technique is employed. Notably, the accuracy of the novelty detection methods in the first stage can be further enhanced with the incorporation of supervised learning techniques, particularly when a sufficiently large number of labeled samples are amassed. The proposed method consistently maintains accuracy, even in the face of changing environments, as it demonstrates the ability to adapt to data drift without necessitating the acquisition of new labeled data in the initial stage. This adaptability makes it particularly well suited for managing imbalanced datasets, rendering it highly practical for industrial applications. In a comprehensive case study conducted within the HDD industry, the framework exhibits immediate adaptability to rapidly changing environments while preserving high accuracy. This highlights the practical effectiveness of the proposed dual-stage classification framework in addressing the unique challenges posed by industrial scenarios.

1. Introduction

The hard disk drive (HDD) test process generates over a thousand parameters, making it difficult and time-consuming to identify outlier samples, but this is an essential task to improve the quality of the industrial test process. To systematically monitor and ensure product quality,

*Corresponding author: e-mail: somyot.ka@kmitl.ac.th
https://doi.org/10.18494/SAM5036

an additional screening process has been proposed for outlier samples. However, the HDD qualification process is not static, and changes may occur as a result of improvements in materials or testing algorithms, necessitating the need for a reanalysis process. This can be a laborious task for humans, which is why machine learning is proposed as a solution. The application of a supervised algorithm(1–3) has been proposed to classify outliers as failure patterns in the data. However, this approach has limitations in practical production environments where the data can change frequently. Retraining supervised algorithms becomes necessary when changes occur, and collecting labeled data, which can be time-consuming, limits their applicability. In our case study, some rare patterns can take time (up to many months) just to collect data for training. In the domain of supervised learning, the enhancement of predictive model accuracy is directly proportional to the augmentation of training samples. The insufficiency of samples within any class can result in an underfit model, highlighting the importance of acquiring a sufficiently large dataset for all classes. The principal challenge for imbalanced data lies in the need for an ample collection of samples. In the industrial test process, numerous pass samples will be generated over time but failing samples or defects will rarely occur. Defective parts per million (DPPM) is a critical indicator of quality performance. The increased challenge associated with lower DPPM values lies in the complexities of acquiring failing samples. This challenge is particularly pronounced in the context of supervised learning when applied to industrial test processes. The difficulty is further compounded in situations where the process undergoes frequent changes, necessitating frequent retraining of the model. In contrast, novelty detection algorithms only require pass samples for training and do not necessitate failing samples. Consequently, their application to the industrial test process is notably more straightforward. The duration allocated for collecting data has been significantly reduced, resulting in a shortened data collection period. This adjustment proves beneficial in efficiently addressing the challenges posed by a rapidly changing environment. However, initiating the development of a novelty detection model requires careful feature selection. Unlike supervised learning, these models are highly sensitive to feature dimensions. The prudent selection of features is crucial for the model's effectiveness in identifying novel patterns. Several groups have addressed the novelty detection and identification task. For instance, Peng *et al.*(4) applied the local outlier factor (LOF) algorithm to identify outliers effectively by comparing the density of each data. Velasquez *et al.*(5) suggested an ensemble anomaly detection approach that integrates three models, namely, LOF, one-class support vector machine (OCSVM), and autoencoder. Termite *et al.*(6) introduced a method for monitoring conditions, which diagnoses anomaly types and identifies novel conditions. This method uses a one-nearest classifier based on a regularly updated dictionary. Saucedo-Dorantes *et al.*(7) applied a self-organizing map (SOM) model for each machinery condition. The authors conducted novelty detection and fault diagnostics by introducing new data samples to each SOM and measuring their similarity to the data used in constructing SOM. However, these methodologies possess certain limitations, including the imperative for careful feature selection. Moreover, novelty detection methods are trained on a single class. Novelty detection algorithms never learn about other classes during training, and they lack a comprehensive understanding of the data distribution across different classes. Single class training can lead to a lower accuracy than those of supervised learning methods that have learned from all classes.

The proposed method addresses the limitations associated with both novelty detection and supervised learning approaches by seamlessly integrating them into a more robust framework. This framework effectively eliminates the disadvantages associated with novelty detection and supervised learning, while retaining their respective advantages. Within the proposed framework, an ensemble novelty detection strategy is employed as the initial stage classifier, combining two models, namely, LOF and OCSVM. The dual-stage classification framework has been carefully designed to align with real world scenarios such that healthy-condition samples will be generated much faster than unhealthy-condition samples, which rarely occur and require a much longer time to collect data. The application of ensemble novelty detection in the initial stage expedites the sample collection, as it eliminates the need for unhealthy-condition samples at this stage. After generating a sufficient amount of unhealthy sample data, the classifier transitions to a supervised classifier in the second stage. This results in an enhanced level of accuracy. The advantages of this approach include rapid adaptation, increased robustness, and high precision. Furthermore, the proposed framework addresses challenges related to overfitting and underfitting. This methodology is applicable across various industrial domains, particularly industrial testing processes.

2. Methodology

2.1 Novelty detection algorithms

Novelty detection algorithms are designed to detect unlabeled samples that deviate significantly from normal patterns. Equation (1) is a mathematical model for classifying an event as a novelty or a normal event.

$$f\left(s_x, t_c\right) = \begin{cases} \textit{Novelty} \ \text{if} \left(s_x > t_c\right) \\ \textit{Normal} \ \text{otherwise} \end{cases} \tag{1}$$

Equation (1) represents a function with two input variables: s_x, and t_c. The variable s_x denotes the abnormality score, whereas the variable t_c corresponds to the current temporal threshold. The threshold could be changed over time on the basis of the current distribution of training samples.

2.2 Supervised learning algorithms

Supervised learning is a type of algorithm that learns from labeled examples. It adopts the designation "supervised" because of the guidance provided by previously known output results during the learning process. During the training phase, the algorithm learns from examples where it receives inputs paired with their corresponding labels. The algorithm is exposed to examples in which it receives inputs paired with their corresponding labels. This means that the algorithm learns from instances where it sees both the input data and the expected output, allowing it to understand and generalize patterns for future predictions or classifications. After the completion of the training phase, the algorithm becomes capable of processing new inputs and making predictions for their associated labels. The principal goal of a supervised learning model is to provide precise label predictions for new input data. Figure 1 depicts the general process of supervised learning.

Supervised learning algorithms generally exhibit greater accuracy than those of novelty detection algorithms such as LOF and OCSVMs. Nevertheless, the limitation of supervised learning lies in its dependence on labeled data, which can pose challenges in practical implementation. Furthermore, the process of acquiring labeled data for infrequent patterns can be both cumbersome and time-consuming.

2.3 LOF

The LOF algorithm, as described in Ref. 8, is a method used to find unusual or novel data points. It accomplishes this by measuring how much the density of a data point differs from its neighbors. If a point has a lower density than its neighbors, LOF labels it as abnormal or novel. The output of LOF is an anomaly score, reflecting the degree of dissimilarity of each data point from its neighboring points. The LOF method consists of the following four steps.

1. Set Cluster Size: Choose the cluster size by deciding on a *k*-value.
2. Calculate Reachability Distance (RD): Measure the RD for each data point, considering its proximity to neighboring points.
3. Compute Local Reachability Distance (LRD): Combine the individual reachability distances to determine the LRD for each data point.
4. Generate Anomaly Scores: Finally, assign a LOF score or anomaly score to each data point to identify potential outliers or anomalies in the dataset.

The *k*-value plays a crucial role in defining the cluster size, influencing the method's ability to detect abnormal data. The RD represents the distance between a data point and its farthest neighbor among the *k* nearest points. The RD can be calculated using Eq. (2), where *XA* and *XA*' denote data points, and *K* is the chosen *k*-value.

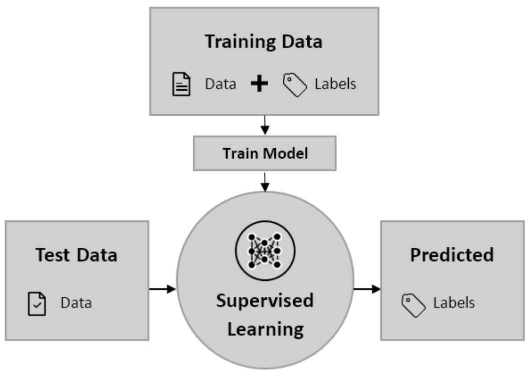

Fig. 1. Supervised learning high-level approach.

$$RD(XA, XA') = \max(K - distance(XA'), distance(XA, XA')) \tag{2}$$

After acquiring the RD, the LRD is calculated using Eq. (3), where $N_k(A)$ is the k nearest neighbors.

$$LRD_k(A) = \frac{1}{\sum_{X_j \in N_k(A)} \frac{RD(A, X_j)}{|N_k(A)|}} \tag{3}$$

The computation of the LOF score or anomaly score of individual data points relies on their corresponding LRD values. This score reflects the ratio of the LRD value associated with a specific data point to the cumulative LRD values across all data points. This metric serves as an indicator for evaluating the relative distance ratio between a given data point and the remaining dataset. The anomaly score is determined by applying Eq. (4),(9) where LOF stands for the LOF score, LRD denotes the LRD value, and $N_k(A)$ represents the k nearest neighbors.

$$LOF_k(A) = \frac{\sum_{X_j \in N_k(A)} LRD_k(X_j)}{|N_k(A)|} \frac{1}{LRD_k(A)} \tag{4}$$

As shown in Fig. 2, the initial stage (referred to as step A) involves the determination of the k-value, which represents the number of nearest neighbors considered. This parameter selection is crucial in establishing the optimal neighborhood size for subsequent computations. Proceeding to step B, the algorithm calculates the RD for each data point.

In step C, the LRD for each data point is computed by aggregating the RD values of its k nearest neighbors. This process measures how closely packed the data is in a local area and makes it easier to see where the data is more or less concentrated.

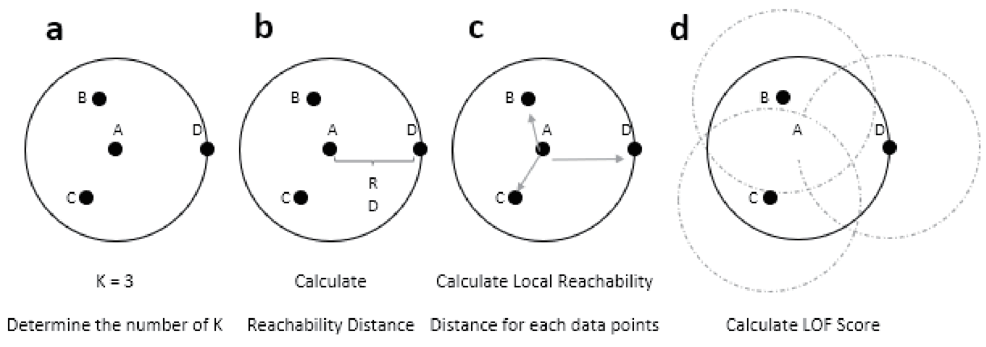
a
b
c
d
B
A
C
D
R
D
K = 3
Determine the number of K
Calculate
Reachability Distance
Calculate Local Reachability
Distance for each data points
Calculate LOF Score

Fig. 2. Illustration of LOF process.

Lastly, in Step D, the LOF score is calculated for each data point, leveraging the LRD values to evaluate its deviance behavior relative to the local neighborhood. This step provides valuable insights into potential anomalies or novelty within the dataset.

2.4 OCSVM

The OCSVM algorithm, introduced by Schölkopf *et al.*,[10] functions as a technique for novelty detection, focusing on identifying outliers. In contrast to the conventional SVM model that necessitates the inclusion of samples from both the majority class and the class representing novel instances for classifier construction, OCSVM only requires normal data samples during the training phase. The objective of OCSVM is to distinguish potential novelty samples from among samples indicative of normal operational conditions. Throughout the training stage, OCSVM calculates the support margin or "boundary" that covers the majority of the training points on the basis of the normal operating training samples. Subsequently, during testing, if a sample lies within this boundary, it is classified as a normal operational sample; conversely, if it falls outside this range, it is designated as a novelty sample. However, for nonlinear boundaries, kernel transformation is employed to project the data into a higher-dimensional feature space. Figure 3 illustrates a visual presentation of the fundamental concept underlying OCSVM.

In the context of OCSVM, it is assumed that the origin in the feature space belongs to the negative or novelty class. The primary aim is to maximize the separation between the origin and the cluster of normal samples in the feature space. In summary, the objectives of OCSVM are twofold. First, the goal is to construct a classifier within the feature space. This classifier assigns positive values to all the samples within the normal cluster and negative values to those outside it. The second goal is to maximize the distance of this hyperplane from the origin, grounded on the inherent assumption that the origin belongs to the novelty class.

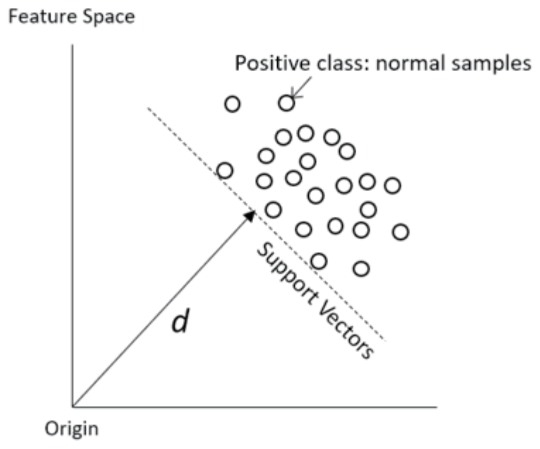

Fig. 3. Illustration of the fundamental concept underlying OCSVM.

2.5 Ensemble methods

A machine learning ensemble combines several algorithms to achieve higher accuracy than that of any single classifier.(11) There are three main types of ensemble: bagging, boosting, and stacking. Bagging and boosting both use the same learning algorithm to predict output labels. The distinction between these two methods lies in their approach to generating successive subsets during classification. In boosting, datasets are generated randomly, whereas in bagging, elements are assigned weights, resulting in varying probabilities of selection for each element.(12,13) The third category, known as stacking or voting, utilizes multiple diverse algorithms operating on the same dataset.(14,15) In essence, bagging is employed to mitigate the model's variance, boosting addresses the model's bias, and stacking combines predictions from various classifiers to enhance performance. A concise comparison of each method is presented in Table 1.

In the proposed method, the stacking ensemble was applied for combining results from multiple models to improve the predictive performance. It is a more accurate and robust classifier than the single predictive model. In the proposed methodology, a stacking ensemble was utilized to merge outcomes from multiple models, enhancing predictive efficacy. This approach represents a more precise and robust classification technique than using just one predictive model.

2.6 Deep learning

Deep learning(16) is a subset of artificial neural networks distinguished by the construction of networks comprising multiple layers, often ranging from tens to potentially thousands of layers, which is why it is referred to as "deep". The depth of these networks empowers them to address more complex problems by autonomously extracting data features without relying on predefined logic.

In general, deep learning involves using a series of layers to extract features from data. Linear models or shallow neural networks (SNNs) might not be good enough for making the best predictions. Consequently, deep neural networks (DNNs) have become a way of making a model more accurate. Figure 4 illustrates the structure of an SNN, and Fig. 5 illustrates that of a DNN.

Table 1
Ensemble methods: boosting, bagging, and stacking.

Method	Characteristics	Result
Bagging	Homogeneous learners, parallel fitting	Focus on reducing variance, i.e., random forest
Boosting	Homogeneous learners, sequential fitting	Focus on reducing bias, model can be improved with gradient descent approach, i.e., XGBoost
Stacking	Heterogeneous learners, parallel fitting, combines learner results	Focus on reducing bias, i.e., majority vote

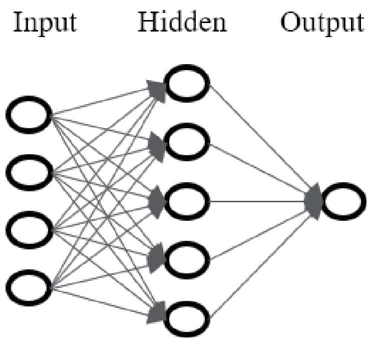

Fig. 4. Structure of SNN.

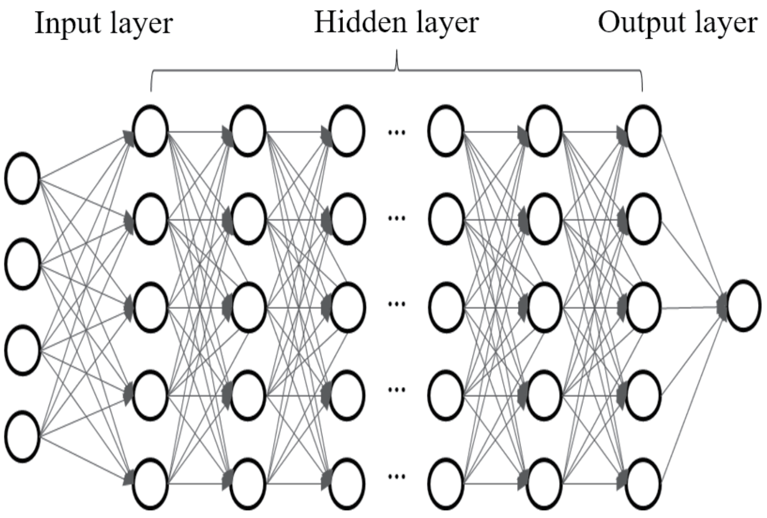

Fig. 5. Structure of DNN.

2.7 Proposed method

The proposed method is aimed at combining the strengths and eliminating the weaknesses of both novelty detection and supervised algorithms. Initially, when changing conditions are applied, no failing samples are available while many pass samples are present. Therefore, a robust novelty detection framework based on ensemble learning with LOF and OCSVM is proposed as the first-stage classifier to classify abnormally different samples from normal patterns.

The first stage operates until enough failing samples are collected to train the second-stage classifier, which is a supervised model. In the second stage, the supervised model is trained and compared against the first-stage classifier. If the supervised model outperforms the first-stage classifier, it is applied as the second-stage classification and the pattern is defined as a new known pattern. Furthermore, the proposed method is an incremental learning approach that allows new patterns to be added as new models. The performance of the dual-stage classification framework showed greater robustness to overfitting and underfitting problems. Additionally, the proposed method can adapt to drift in the data without requiring new labeled data, making it more practical for industrial applications. The proposed method architecture is shown in Fig. 6.

3. Experimental Procedure

The proposed methodology is evaluated and compared with various novelty detection and supervised learning algorithms. The total number of collected samples is 381,882, i.e., 73 failing samples and 381,809 pass samples. This framework detects failing samples using a predictive model. As such, failing samples are positive class and pass samples are negative class. All data are collected from an actual industrial test process. The experimental process consists of four main steps: data preparation, feature engineering, feature score ranking, and feature selection.

3.1 Data preparation

In this step, two datasets are prepared: one containing failing drives associated with the chosen symptom and the other representing a sample that is representative of a pass drive.

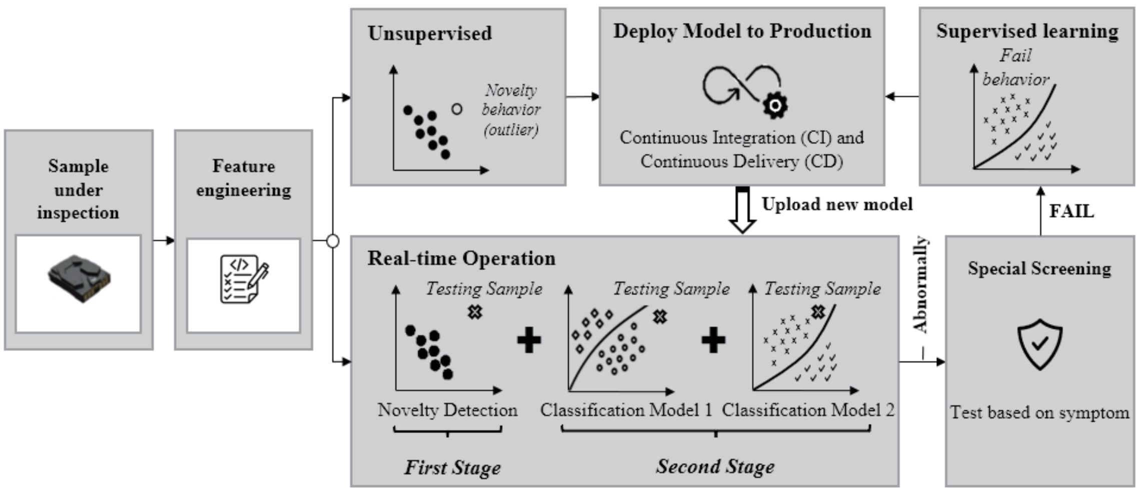

Fig. 6. Overview architecture of proposed method.

3.2 Feature engineering

In this step, domain experts brainstormed and created a list of all potential features. First, domain experts started with 218 features, which is considered high dimensionality in data science. To reduce the dimensionality, data scientists used a decision tree algorithm to identify the key features. The feature selection process of the decision tree algorithm helped domain experts see which features were most useful for predicting outcomes. After a careful review together, the team of data scientists and domain experts chose four features for constructing the model.

3.3 Feature score ranking

The pruned decision tree algorithm was utilized for feature score ranking. This technique provides feature membership, which represents the percentage of potential feature sets where a given feature is presented in a pruned decision tree.

3.4. Feature selection

To ensure our model works well not just on the training data but also on new data it has not yet seen, we need to think about more than just how well it performs on the training data. We should look at other important measures to make the model reliable. The following metrics should be taken into consideration.

- Metric 1: number of features (fewer is better)
- Metric 2: number of test algorithms applied to generate the features (more is better).

After careful review, domain experts selected four features to build the final model.

3.5 Building predictive models

Creating and selecting meaningful features are essential for achieving success in the proposed method. Configuration refers to the arrangement or settings of hardware and software components. On the basis of experimental results, it has been found that building a common model without separate configurations leads to a lower predictive performance in the final model. Figure 7 shows the performance of the common model without separate configurations and with a lot of false positives and false negatives. To improve the predictive performance, an approach was implemented where the model was built using separate configurations, specifically focusing on the main configuration. This approach resulted in significant improvements in prediction accuracy, as depicted in Fig. 8.

Table 2 shows the performance of the different algorithms, all of which used the same features and preprocessing techniques. Ten models were created per algorithm, with random training and testing sets. Table 2 presents the average, minimum, and maximum accuracies for each algorithm.

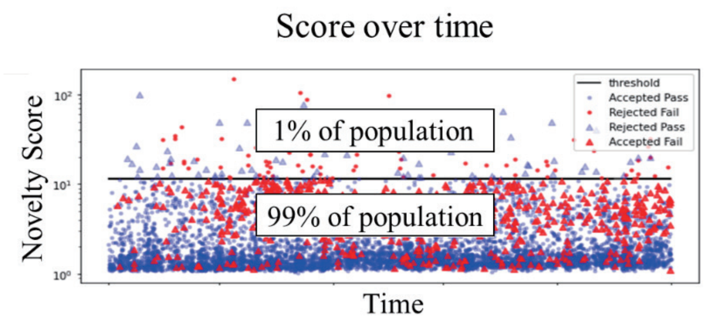

Fig. 7. (Color online) Performance of common model without separate configurations.

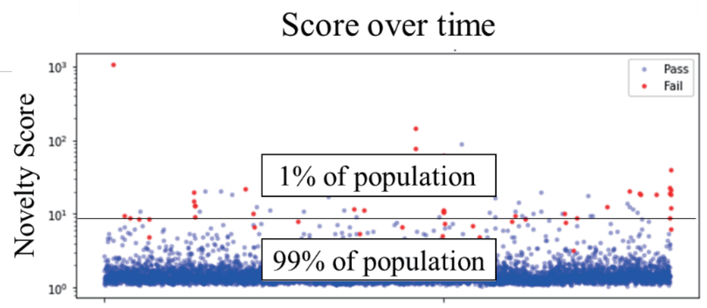

Fig. 8. (Color online) Performance of separate configuration model.

Table 2
Accuracies of algorithms.

Type	Algorithm	Min	Max	Mean
Novelty Detection	Ensemble of LOF or OCSVM	0.87	0.92	0.89
Novelty Detection	OCSVM	0.87	0.90	0.89
Novelty Detection	LOF	0.80	0.90	0.87
Novelty Detection	Isolation Forest	0.10	0.67	0.37
Novelty Detection	Elliptic Envelope	0.84	0.85	0.84
Supervised	SVM	0.85	0.97	0.91
Supervised	Deep Neural Network	0.84	0.95	0.91
Supervised	AdaBoost	0.69	0.95	0.88
Supervised	Gaussian Naïve Bayes	0.82	0.89	0.87
Supervised	Quadratic Discriminant Analysis	0.80	0.90	0.86
Supervised	*k* Nearest Neighbors ($k = 3$)	0.70	0.92	0.84
Supervised	Random Forest	0.59	0.95	0.84

Table 3 shows the parameter details of each algorithm. The algorithms in Table 3 utilize the Python programming language for their implementation.

To address the class imbalance, a 70:30 split ratio for the testing and training sets was utilized. However, since the negative class has an excessive number of samples compared with

Table 3
Parameters of algorithms.

Algorithm	Input Parameters		Output	Negative Class		Positive Class	
	Original	Reduced		Original Samples	Training Samples	Original Samples	Training Samples
Ensemble of LOF or OCSVM	218	4	1	381,882	1,000	78	0
OCSVM	218	4	1	381,882	1,000	78	0
LOF	218	4	1	381,882	1,000	78	0
Isolation Forest	218	4	1	381,882	1,000	78	0
Elliptic Envelope	218	4	1	381,882	1,000	78	0
SVM	218	4	1	381,882	1,000	78	20
Artificial neural network	218	4	1	381,882	1,000	78	20
AdaBoost	218	4	1	381,882	1,000	78	20
Gaussian Naïve Bayes	218	4	1	381,882	1,000	78	20
Quadratic Discriminant Analysis	218	4	1	381,882	1,000	78	20
k Nearest Neighbors (*k* = 3)	218	4	1	381,882	1,000	78	20
Random Forest	218	4	1	381,882	1,000	78	20

the positive class, downsampling was applied in addition. This technique ensures a balanced representation of both classes during training, enhancing the model's ability to learn patterns and make accurate predictions.

The experimental results showed that the proposed methodology achieves up to 92% accuracy for the best model and an average of 89% using the ensemble model of LOF and OCSVM for novelty detection. While supervised algorithms such as SVM and DNN can achieve higher accuracy, they require a significant amount of time to collect labeled data. This time-consuming process is exacerbated by rare patterns such as failure patterns in manufacturing processes. In the proposed method, supervised learning algorithms are not fixed, and on the basis of the concept of Industry 4.0, modern software development is supported by continuous integration (CI) and continuous delivery (CD) or machine learning operation,(17) allowing all models from supervised learning algorithms to be mixed and pushed to the running production environment without interrupting the run. In other words, the system can execute multiple algorithms in real time and support the addition of new models on the fly.

4. Conclusions

The proposed method addresses the classification of rare or unseen patterns in a continuously changing environment such as the HDD test process. The method involves a dual-stage classification approach using both ensemble novelty detection and supervised learning algorithms. The first stage utilizes an ensemble learning approach with LOF and OCSVM, achieving an accuracy of up to 92%. The second stage involves supervised learning using either SVM or DNN, which achieves an accuracy of up to 97%. The combination of these two algorithms allows the system to adapt to data drift without needing new labeled data. Additionally, once labeled data are available, the system can further improve the classification accuracy.

The proposed framework helps eliminate the weaknesses of each algorithm. For example, underfitting in the first-stage novelty detection can be improved by the supervised learning algorithm in the second stage, while overfitting in the supervised learning algorithm can be prevented by a long validation time. The long validation time is acceptable in practical applications because of the support provided by the ensemble novelty detection model.

Furthermore, the proposed method supports incremental learning, where new patterns can be added to the system without interruption. This is made possible by the continuous integration and delivery (CI/CD) process, which is a modern software development practice. The system can execute multiple algorithms in real time, and new predictive models can be added to the production environment without interrupting the run. By solving problems related to underfitting, overfitting, and real-time operation, the proposed dual-stage classification concept offers a more robust and efficient approach to classification in a continuously changing environment.

The proposed method was successfully applied to quality monitoring in the HDD test process, and it has a significantly improved defect detection capability. By utilizing this concept, the sampling rate can be reduced by 80% compared with the traditional decision boundary defined by humans. This approach can also be applied to other industrial sectors, which is the benefit of the proposed method.

References

1 R. K. Lala: Proc. 2021 IEEE 8th Uttar Pradesh Section Int. Conf. Electrical, Electronics and Computer Engineering Conf. (IEEE, 2021) 1–6. https://doi.org/10.1109/UPCON52273.2021.9667609

2 P. Sihakhom, S. Sulistyo, and I W. Mustika: Proc. 2020 6th Int. Conf. Science and Technology (IEEE, 2020) 1–6. https://doi.org/10.1109/ICST50505.2020.9732829

3 J. Sakhnini, H. Karimipour, and A. Dehghantanha: Proc. 2019 IEEE 7th Int. Conf. Smart Energy Grid Engineering (IEEE, 2019) 108–112. https://doi.org/10.1109/SEGE.2019.8859946

4 Y. Peng, Y. Yang, Y. Xu, Y. Xue, R. Song, J. Kang, and H. Zhao: IEEE Access **9** (2021) 107250. https://doi.org/10.1109/ACCESS.2021.3100980

5 D. Velásquez, E. Pérez, X. Oregui, A. Artetxe, J. Manteca, J. E. Mansilla, M. Toro, M Maiza, and B. Sierra: IEEE Access **10** (2022) 72024. https://doi.org/10.1109/ACCESS.2022.3188102

6 M. R. Termite, P. Baraldi, S. Al-Dahidi, L. Bellani, M. Compare, and E. Zio: Energies **12** (2019) 4802. https://doi.org/10.3390/en12244802

7 J. J. Saucedo-Dorantes, M. Delgodo-Prieto, R. A. Osornio-Rios, and R. J. Romero-Troncoso: IEEE Trans. Ind. Inf. **16** (2020) 5985. https://doi.org/10.1109/TII.2020.2973731

8 A. A. Megantara and T. Ahmad: J. Big Data **8** (2021) 1. https://doi.org/10.1186/s40537-021-00531-w

9 M. M. Breunig, H. Kriegel, R. T. Ng, and J. Sander: Association for Computing Machinery Special Interest Group on Management of Data 2000 Int. Conf. (ACM SIGMOD, 2000) 93–104. https://doi.org/10.1145/335191.335388

10 B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson: Neural Comput. **13** (2001) 1443. https://doi.org/10.1162/089976601750264965

11 L. Rokach: Artif. Intell. Rev. **33** (2010) 1. https://doi.org/10.1007/s10462-009-9124-7

12 L. Breiman: Mach. Learn. **24** (1996) 123. https://doi.org/10.1007/BF00058655

13 T. Hastie, R. Tibshirani, and J. Friedman: The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Springer Science+Business Media, USA, 2009) 2nd ed., Chap. 16. https://doi.org/10.1007/b94608

14 Y. Wang, Y. Shen, and G. Zhang: Proc. 2016 7th IEEE Int. Conf. Software Engineering and Service Science Conf. (ICSESS, 2016) 422–425. https://doi.org/10.1109/ICSESS.2016.7883100

15 D. Upadhyay, J. Manero, M. Zaman, and S. Sampalli: IEEE Trans. Network Sci. Eng. **8** (2021) 2559. https://doi.org/10.1109/TNSE.2021.3099371

16 T. Zhu, K. Li, P. Herrero, P. Georgiou: IEEE J. Biomed. Health. Inf. **25** (2021) 2744. https://doi.org/10.1109/JBHI.2020.3040225

17 D, Kreuzberger, N. Kühl, and S. Hirschl: IEEE Access **11** (2023) 31866. https://doi.org/10.1109/ACCESS.2023.3262138