# Real Time Drift Detection and Adaptation Using Hybrid ADWIN in Agricultural Environmental Monitoring System

Hezal Lopes[1,2,*] and Prashant Nitnaware[2]
[1] D.J. Sanghvi College of Engineering, Vile Parle, India
[2] Pillai College of Engineering, New Panvel, India
Email: hezal.lopes@gmail.com (H.L.), pnitnaware@mes.ac.in (P.N.)

Manuscript received May 31, 2025; revised June 18, 2025; accepted July 21, 2025
*Corresponding author

***Abstract*—Real-time analysis of streaming data is crucial in agricultural environmental monitoring to address quickly changing conditions like seasonal weather changes. Concept drift, where the statistical characteristics of input data evolve, poses a significant problem for static machine learning models. This research presents a drift-aware framework based on a hybrid adaptive windowing method combined with an Online Sequential Extreme Learning Machine (OS-ELM). The strategy involves a multi-dimensional extension of Adaptive Windowing (ADWIN) that is supplemented by the Kolmogorov–Smirnov statistical test and Hoeffding's bound to identify and respond to real-time drift. An experimental Internet of Things (IoT) platform was constructed to gather environmental parameters such as temperature, humidity, soil moisture, light, pH, and rainfall. Empirical tests on real and synthetic datasets show that the new framework greatly enhances predictive performance, from 85.86 percent to 97.29 percent when drift handling is activated. The findings emphasize the significance of combining adaptive learning with drift detection for accurate and dependable prediction in precision agriculture.**

***Index Terms*—adaptive windowing, concept drift, data stream mining, drift detection, Extreme Learning Machine (ELM), Internet of Things (IoT), sensor data, precision agriculture**

## I. Introduction

Online learning experiences a change in the distribution of data while extracting meaningful information from data streams. That change in data distribution is known as concept drift. The classifier classifies incoming instances using past training data. The accuracy of the classifier deteriorates because of the concept drift. The traditional classifiers are not trained to learn the patterns in an evolving distribution of data [1].

One such classic example is weather forecasting, where models predicting based on seasonal data become ineffective in light of changing climatic patterns or anomalies. Seasonal shifts, like monsoon and dry seasons, can bring in large variations, making previously learned patterns ineffective. Existing sensor anomaly detection techniques are mostly designed for general purposes and may not be suitable for climate sensors, which require complex handling of seasonality, spatial relationships, and sensor interdependency [2]. Another example Intrusion detection system where shifts in statistical distributions within data streams pose critical cybersecurity threats [3].

To counter this, several drift detection methods have been suggested, e.g., Adaptive Windowing (ADWIN), Drift Detection Method (DDM), and Early Drift Detection Method (EDDM), which track performance metrics of a classifier and statistical characteristics over time [1]. Most of them employ sliding or adaptive window mechanisms to monitor changing distributions, often improving detection effectiveness when combined with statistical tests like the Kolmogorov–Smirnov (K-S) test [4] or Cumulative Sum (CUSUM) [5]. Besides, ensemble classifiers have proved to be a more robust alternative to individual-model methods, providing better adaptability and robustness to concept drift. A real-time system is implemented to detect malware within large-scale network traffic streams in the presence of concept drift [6]. Various drift detection techniques are applied to the Network Security Laboratory-Knowledge Discovery in Databases (NSL-KDD) and IoT-23 datasets. K-S test can able to identify concept drift correctly. Long Short-Term Memory (LSTM) learner is used [7] with Page History Table (PHT), ADWIN, and K-S test. K-S method with LSTM works well as compared to other drift detection methods.

In such a context, this current study presents a new framework devised to detect and accommodate drift in real-time data streams through the application of a hybrid adaptive windowing method. The system utilizes an Extreme Learning Machine (ELM) as the underlying classifier. Whenever a decline in classifier performance is noticed, the system triggers a hybrid ADWIN drift detection module. This module utilizes the K-S statistical test to detect changes in distributions within the data window. When a drift is detected, a new ELM classifier is trained on the latest data and is added to an ensemble of models, thus maintaining adaptability while enhancing predictive performance.

This model is specifically used for actual field agricultural environmental data gathered through an experimental IoT system. The dataset contains variables like temperature, humidity, light, gas, pressure, and rainfall, which are naturally seasonal. Acknowledging that a fixed window size can fail to capture actual drift during changing conditions, the system proposed here uses an adaptive windowing approach. The smaller windows are used when there is rapid change, like monsoon periods, and larger windows are used during steadier periods to minimize false alarms.

The rest of this paper follows the following structure: Section II describes background ideas and current approaches. Section III details the drift detection framework. Section IV provides an overview of the datasets employed in this study. Section V explains the suggested hybrid architecture and detection procedure. Section VI provides an overview of the experimental setup and outcomes. Section VII presents the industrial implications of real-time drift detection in IoT systems, and Section VIII concludes the paper and suggests future work.

## II. Literature Review

Concept drift detection is critical in real-time systems, particularly in applications like precision agriculture, where data distributions change. Several algorithms have been proposed to cope with the problem, and they can be classified based on the underlying principles they use. The following subtopics critically review major classes of drift detection approaches, emphasizing strengths, weaknesses, and applicability to dynamic, sensor-based applications.

*Similarity and Dissimilarity-Based Methods:* These methods measure similarity or difference in data distributions through time to detect drift [8]. DDM applies a binomial distribution model to track the error rates of the classifier. While DDM works well for sudden drift, it is susceptible to insensitivity towards gradual drift. EDDM builds on this by emphasizing the distance between classification errors. Exponentially Weighted Moving Average (EWMA) [9] places a heavier emphasis on the latest values, employing a recursive weighting function that enables high-speed reaction times to change. The EWMA statistic is calculated recursively using the Eq. (1):

$$\mathrm{EWMA}(t) = \lambda X(t) + (1 - \lambda)\mathrm{EWMA}(t) \tag{1}$$

where $\lambda$ (lambda) determines the weight assigned to the current observation. A higher $\lambda$ gives more importance to recent observations, while a lower $\lambda$ places more emphasis on past data. The Reactive Drift Detection Method (RDDM) supplements DDM's lack of responsiveness by increasing detection with massive concept shifts. However, these approaches tend to be limited in multivariate or seasonally changing environments because they rely solely on monitoring one-dimensional errors.

*Sequential Analysis-Based Methods*: These techniques track data sequentially and produce drift signals when user-specified deviations are observed beyond thresholds [10]. The Page-Hinkley test, for example, identifies changes in the mean of a data stream. Although effective in cases with uniform patterns, these techniques can be challenged by minor or feature-specific changes in high-dimensional data.

*Statistical-Based Methods:* Statistical methods employ hypothesis testing and distribution-based measures on past and present data to identify drift. The Cumulative Sum (CUSUM) scheme [11] detects long-standing differences from a baseline. McDiarmid drift detection methods (MDDM-A, MDDM-G, MDDM-E) employ McDiarmid's inequality across a sliding window. These versions employ arithmetic, geometric, or Euler-based weighting schemes, respectively. While potent, these methods operate with binary prediction windows and do not fully capture interactions between two or more sensor variables.

*Window-Based Approaches:* Two sliding windows hold past and present data in window-based approaches. Drift is identified statistically by comparing the windows. CVFDT is a well-known instance that learns alternative decision trees on new information and updates poor-performing models accordingly. ADWIN and its enhanced variant, ADWIN2, dynamically scale windows and compare means of subwindows to identify changes. Although these approaches are adaptive and effective, classical ADWIN [12] processes only accept single-dimensional input. This is a constraint that limits its utility in applications with richer, multivariate streams, like those encountered in sensor-based monitoring networks.

*Block-Based Ensembles:* Block-based ensemble algorithms build classifiers out of sequential data blocks and update model relevance with pruning. Streaming Ensemble Algorithm (SEA) rebalances the ensemble by removing low-performing models. Accuracy Weighted Ensemble (AWE) chooses top-performing classifiers based on mean squared error, whereas Accuracy Updated Ensemble (AUE) [13] updates model parameters directly. These are effective in batch learning, yet they tend to need large quantities of labeled data and are inflexible in changing data streams.

*Incremental-Based Ensembles:* Also called online ensembles, these strategies update model weights constantly depending on performance. The Dynamic Weighted Majority (DWM) [14] algorithm penalizes an error in prediction by lowering model weight and eliminates models below a tolerance. Learn++ builds several classifiers for a data batch, updating weights based on error patterns. These schemes are stable in dynamic environments, though their performance will be lower in noisy or weakly supervised scenarios.

*Rule-Based Methods:* Rule-based classifiers make predictions based on interpretable rules that can be separately tested and adapted [15]. Such models have the benefit of transparency and simple modification. But they become unreliable in sophisticated or rapidly changing contexts, where coarse rules do not adequately reflect detailed drift behavior.

*Tree-Based Methods:* Tree-based methods like CVFDT [10] replace subtrees dynamically if accuracy falls. Random Forest classifiers and Vertical Hoeffding

Trees (VHT) are some other methods that further extend flexibility using ensemble learning and distributed processing. Even though these algorithms facilitate continuous learning, they are computationally heavy, especially for resource-constrained deployments.

*Naive Bayes Methods:* Naive Bayes models apply Bayes' theorem to estimate class-conditional probabilities. They are efficient in computation but rely on feature independence, which is frequently not the case for multivariate sensor data. Bayesian networks provide more flexibility since they learn inter-feature relations dynamically. The Fast-Hoeffding Drift Detection Method (FHDDM) [16] monitors classification accuracy over time and raises a drift alarm when performance falls below a level set by Hoeffding's inequality. Although they are efficient, they might confuse natural noise with drift under real-world circumstances.

*Significance Analysis-Based Methods:* This category of methods relies on statistical hypothesis testing for comparison between historical and current data distributions [17]. A two-step process is usually followed, including distribution comparison followed by significance testing, e.g., the Wilcoxon rank sum test. These tests are strong and non-parametric but tend to rely on the presence of labeled data and can create latency owing to dual-window analysis.

*Data Distribution-Based Methods:* Distribution-based methods emphasize the discovery of changes in the statistical properties of input data [1]. In comparing new and old data samples' distributions, these methods can spot localized and global drift. Least squares density difference and local drift degree-based adaptation are some fine-grained analysis-capable techniques. However, computational complexity in modeling multivariate distributions restricts their applicability in real-time.

*Decision Boundary-Based Methods:* These methods monitor shifts in classifier decision boundaries and not input distributions [1]. A degradation of classification performance or boundary certainty usually indicates drift. MD3 and Re-DBSCAN, a graph-based algorithm, adapt internal models in case of large boundary changes. While these approaches are enabled for early detection, they are potentially expensive in terms of computation and are more effectively applied to offline or periodic analysis.

*Model-Dependent Methods:* These approaches train a model on a first data subset and then observe shifts in the joint probability distribution $P$ (Target|Input) [1]. ExStream and ExStremAttr are tools that are based on model interpretability and observe how pairs of feature values affect predictions on outputs. These are good for explaining drift, but can be challenging in noisy data or when the model structure is outdated.

*Machine Learning-Based Methods:* Sophisticated learning models like restricted Boltzmann machines (RBM) and Bayesian nonparametric methods [18] provide flexible, data-driven drift detection. RBM (neural network) is used for supervised stream classification, and the Bayesian nonparametric method is used for unsupervised detection, which uses structure-learning techniques. But the disadvantage of these methods is that they require large training data and hyperparameter tuning to work in real-world deployments.

*Active Learning-Based Methods:* Active learning methods try to increase the efficiency of labeling during drifts. RAND++ selects instances for labeling according to a pre-defined budget [19], while VAR-UN++ dynamically adjusts the uncertainty threshold based on the previously observed performance and the severity of the drift. Such methods save costs in annotation but depend on reliable drift signals and uncertainty estimates to work optimally.

While most of the drift detection methods discussed in the review have good drift detection strength, there are still some issues. Most of the classical methods are tailored for one-dimensional statistics or based on batch-labeled data. Even the size of the window is important. For fixed-size windows fail to capture seasonal or periodic variations, which are common in environmental monitoring systems. These are the drawbacks that may increase false positives or miss the drifts.

To overcome the above limitations, the author has introduced the hybrid adaptive windowing method that takes advantage of statistical distribution testing and dynamic window adjustment, leveraging ADWIN's functionality to *N*-dimensional data streams. Through the implementation of the K-S test on multiple dimensions of sensors and the use of Hoeffding-based sensor deviation analysis, this novel technique enhances accuracy in detection without compromising on computational efficiency. This renders it particularly well-fitted for real-time agricultural scenarios involving intricate seasonal and environmental dynamics.

## III. Proposed Methodology

This section introduces the novel hybrid adaptive windowing approach for real-time concept drift detection in multi-dimensional sensor data streams with particular emphasis on agricultural environmental monitoring. Conventional drift detection algorithms like ADWIN can only handle one-dimensional (1D) data, which often measures the classification error. In agricultural data streams, there are correlated features like temperature, humidity, rainfall, soil moisture, pH, and light intensity. Processing every feature or attribute with a separate window is very expensive and may miss inter-feature drift. To address and solve this issue, the suggested approach uses N-dimensional K–K-S-based adaptive windowing (NDKSWIN) that leverages the K-S statistical test in multiple dimensions for efficient drift detection.

The K-S test applies to compare two distributions by finding the largest discrepancy between the theoretical cumulative distribution function $F(x)$ and the empirical cumulative distribution function $F_n(x)$ from the windowed data stream:

$$D_n = \overset{\sup}{x} |F(x) - F_n(x)| \tag{2}$$

Here, sup represents the supremum over all possible values of $x$. If $D_n$ exceeds the critical value $D_{\text{critical}}$ $(\alpha, W)$, where $\alpha$ is the significance level and $W$ is the window size, a drift is flagged.

The system needs to accommodate seasonal trends like monsoon and dry seasons. A fixed window size can lead to seasonal changes being mistaken for concept drift. Hence, adaptive windowing is used. In times of high environmental change (e.g., monsoon), the window is dynamically reduced to enhance responsiveness, whereas in steady times, it is increased to reduce false positives.

To determine the origin of drift, Hoeffding's Bound-Based HDDM is used. For every sensor si, the deviation is measured in terms of Hoeffding's inequality:

$$P(\mu - \hat{\mu} > \epsilon) \leq 2e^{-2n\epsilon^2} \tag{3}$$

The sensor with the maximum deviation is termed the source of drift and utilized to cause specific model updates.

## A. Algorithmic Workflow

Fig. 1 summarizes the algorithmic workflow of this study.

Step 1: Initialization

- Initialize adaptive window $W$
- Set significance level α for the K-S test
- Define threshold Hthresh for Hoeffding's bound-based drift detection

Step 2: Data collection and window update

- Sensor readings $S$ are continuously collected and appended to the adaptive window $W$.

Step 3: Drift detection via $N$-dimensional ADWIN (NDKSWIN)

- If the current window exceeds the minimum size threshold, empirical and theoretical cumulative distribution functions are computed:

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{II}(X_i \leq x) \tag{4}$$

- The K-S statistic is evaluated as:

$$D_n = \sup_{x} |F(x) - F_n(x)| \tag{5}$$

- If $D_n > D_{\text{critical}}$ $(\alpha, W)$, drift is detected.

Step 4: Adaptive window size adjustment

- The window size is adjusted depending on the environmental stability:

$$W_{\text{new}} = \begin{cases} W_{\text{old}}\beta & \text{if rapid change detected}, \beta < 1 \\ W_{\text{old}}\gamma & \text{if stable conditions}, \gamma > 1 \end{cases} \tag{6}$$

Step 5: Sensor deviation analysis using HDDM

Hoeffding's bound is applied across all sensors to identify which sensor contributed most significantly to the drift:

Step 6: Model update and adaptation

Upon drift detection, the learning model is updated using the new data pattern associated with the drifted sensor:

These steps are repeated in a continuous monitoring loop, enabling the system to adapt in real-time to evolving environmental patterns.

Here is the detailed real-time adaptation and drift detection process with the use of NDKSWIN, followed by the pseudocode for the same (Algorithm 1).

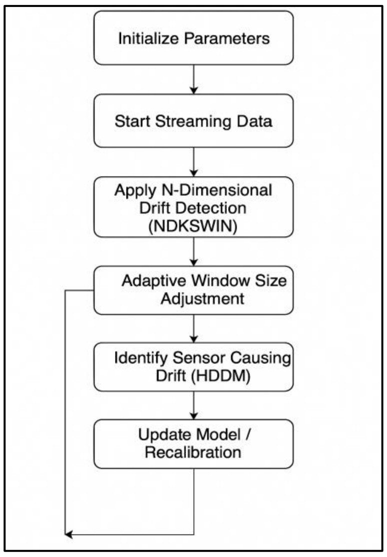

Initialize Parameters
Start Streaming Data
Apply N-Dimensional Drift Detection (NDKSWIN)
Adaptive Window Size Adjustment
Identify Sensor Causing Drift (HDDM)
Update Model / Recalibration

Fig. 1. Algorithm flow diagram.

Algorithm 1. Pseudo-code for Hybrid NDKSWIN-HDDM Drift Detection and Adaptation

```
1: Initialize parameters
Initialize adaptive window $W$
Set significance level $\alpha$ for the K-S test
Set Hoeffding bound threshold H_thresh
Initialize the base learning model
2: Data collection and window update
while True:   # Continuous real-time monitoring
   new_data ← receive_sensor_data(S)   # Get latest sensor readings
   W.append(new_data)   # Add to adaptive window
3: Drift detection via $N$-dimensional ADWIN (NDKSWIN)
   if len($W$) > minimum_window_size then
      Fn_x ← compute_empirical_CDF(W)   # Empirical CDF
      F_x ←compute_theoretical_CDF(W)   # Theoretical/ Reference CDF
      D_n ← max|F_x - Fn_x|  # K-S statistic
         if D_n > critical_value($\alpha$, $W$) then
            print("Concept drift detected!")
4: Adaptive window size adjustment
         if rapid_environmental_change_detected($W$) then
            $W$ ← shrink_window($W$)   # Reduce window size during fast changes
         else
            $W$ ← expand_window($W$)   # Increase window size during stable periods
5: Sensor deviation analysis using HDDM
      drift_sensor ← detect_drift_source($W$, $H$_thresh)
      print("Drift detected in sensor:", drift_sensor)
6: Model update and adaptation
update_model(drift_sensor) # Retrain using new data
```

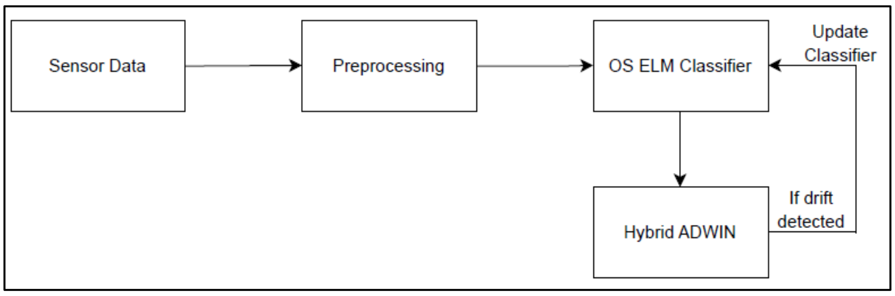


Fig. 2. Proposed hybrid ADWIN-based adaptive monitoring system.

## B. Preprocessing Pipeline

As shown in the architecture Fig. 2, environmental information is gathered through various sensors such as temperature, humidity, light, smoke, pH, soil moisture, rainfall, and particulate matter. Preprocessing steps are performed to provide clean and uniform data before analysis.

Noise is removed via Gaussian smoothing, timestamp synchronization, and missing value handling through KNN imputation, mean imputation, or interpolation. Data integrity validation is done via Little's MCAR test and the K-S test to verify distributional consistency.

Outlier detection employs Z-score ($|X-\mu|>3\sigma$) and Interquartile Range (IQR) methods. Additional anomalies are identified using isolation forest and Local Outlier Factor (LOF) algorithms. Min-max scaling and Z-score Standardization are done to normalize the data as shown in equation (7):

$$X_{\text{scaled}} = \frac{X-X_{\min}}{X_{\max}-X_{\min}}, X_{\text{standardized}} = \frac{X-\mu}{\sigma} \tag{7}$$

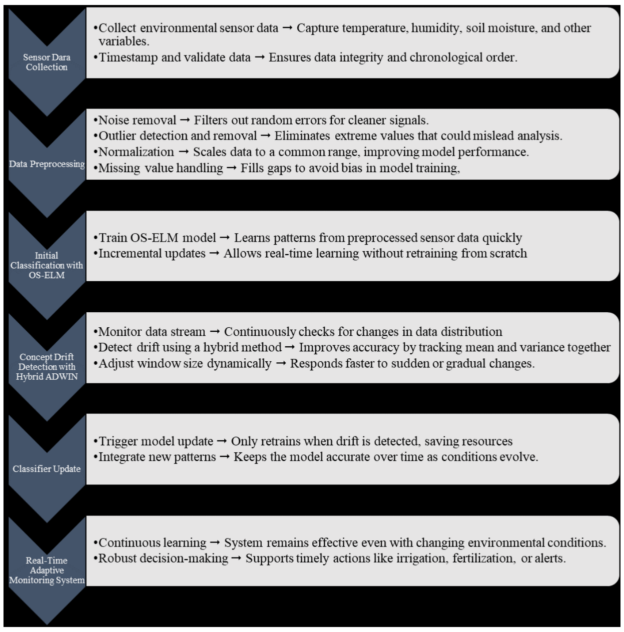
The selection of features is done using Pearson correlation, PCA, mutual information score, and Variance Inflation Factor (VIF) to prevent multicollinearity. The set is subsequently divided into subsets for training and testing purposes under either a 70:30 or an 80:20 proportion. K-S tests both subsets to ascertain statistical similarity.

For time-series data, rolling and expanding window cross-validation methods are employed to ensure stable evaluation while training the model. These preprocessing strategies are essential for guaranteeing trustworthy and accurate real-time drift detection in agricultural environments.

## C. Classifier and Adaptation Mechanism

Extreme Learning Machine (ELM) is used as the base classifier because it has the capability of fast training and good generalization performance. When the accuracy of the classifier falls below a specified threshold, a drift warning is triggered. This is followed by a formal drift test with the Hybrid ADWIN detection unit. Once drift is confirmed, the system re-trains the classifier from new instances, re-tuning the model to rebuild predictive performance and adjust to prevailing conditions.

To further elaborate, Fig. 3 details each step of the framework, outlining the sub-processes and their specific contributions toward achieving accurate and efficient drift detection and adaptation.


Sensor Dara Collection
Data Preprocessing
Initial Classification with OS-ELM
Concept Drift Detection with Hybrid ADWIN
Classifier Update
Real-Time Adaptive Monitoring System

•Collect environmental sensor data → Capture temperature, humidity, soil moisture, and other variables.
•Timestamp and validate data → Ensures data integrity and chronological order.

•Noise removal → Filters out random errors for cleaner signals.
•Outlier detection and removal → Eliminates extreme values that could mislead analysis.
•Normalization → Scales data to a common range, improving model performance.
•Missing value handling → Fills gaps to avoid bias in model training,

•Train OS-ELM model → Learns patterns from preprocessed sensor data quickly
•Incremental updates → Allows real-time learning without retraining from scratch

•Monitor data stream → Continuously checks for changes in data distribution
•Detect drift using a hybrid method → Improves accuracy by tracking mean and variance together
•Adjust window size dynamically → Responds faster to sudden or gradual changes.

•Trigger model update → Only retrains when drift is detected, saving resources
•Integrate new patterns → Keeps the model accurate over time as conditions evolve.

•Continuous learning → System remains effective even with changing environmental conditions.
•Robust decision-making → Supports timely actions like irrigation, fertilization, or alerts.

Fig. 3. Detailed process breakdown of the proposed hybrid ADWIN framework.

To ensure real-time responsiveness, the system combines lightweight statistical operations (K-S test and Hoeffding bound) with the fast-retraining ability of OS-ELM. This makes it suitable for high-frequency data streams, where frequent drift events require low-latency adaptation without overburdening computational resources.

## D. Experimental Setup

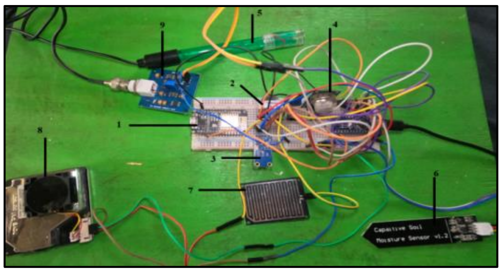


Fig. 4. Experimental setup.

Experimental setup consists of a processing board with a network of environmental sensors: humidity (2), temperature (3), smoke (4), analog pH (5), capacitive soil moisture (6), rainfall (7), PM sensor (8), and light sensor

(9), as shown in Fig. 4. The sensor readings are gathered in real-time and communicated to the ThingSpeak cloud platform, which uses them for detecting drift and model adaptation.

### *E. Datasets*

To compare the given method, several real-world and synthetic datasets are employed:

*Airlines Dataset* [20]: Holds 539,384 instances of flight delay information based on airline, airport, and day of the week.

*Electricity Dataset* [21]: Produced by the Australian New South Wales Electricity Market, it is an example of price variation due to demand and supply.

*Poker-Hand Dataset* [22]: Includes 829,201 examples of poker hands with 11 features each, for multi-class classification.

*Sensor Stream Dataset* [23]: Genuine sensor data provided by Intel Berkeley Research Lab in the form of temperature, humidity, light, and voltage readings.

*SEA and Hyperplane Datasets:* Controlled datasets that experience gradual and abrupt concept drifts according to drifting decision boundaries.

*Agarwal Dataset:* A mixed-attribute dataset having six numeric attributes and three categorical attributes, largely employed for the assessment of drift detection.

## IV. Implementation and Results

To compare the performance of the suggested Hybrid ADWIN-based drift detection system, a set of experiments was performed using several real-world and synthetic datasets. All classifier–drift detection pairs were assessed based on common metrics such as accuracy, precision, recall, and F1-score. The experimental environment was Google Colab as the processing platform, and sensor readings like pressure, temperature, humidity, rainfall, gas, and soil moisture were obtained using an IoT-based sensor system (Fig. 5).

### *A. Dataset-Wise Performance Analysis*

*SensorStream Dataset:* The best performance on this dataset was seen with the Online Sequential Extreme Learning Machine (OS-ELM) in conjunction with Hybrid ADWIN, with an F1-score of 90%. Other high-performing models were Random Forest with ADWIN and ELM with ADWIN, which also provided competitive results. This demonstrates the power of ensemble classifiers when used with adaptive drift detection methods in dynamic sensor environments.

*PokerHand Dataset:* Here, too, OS-ELM with Hybrid ADWIN performed best, achieving an F1-score of 82.5%. The rest of the combinations trailed behind with lower F1-scores. Overall performance of all classifiers, though, was relatively lower, which is probably due to the class imbalance and complexity in the PokerHand dataset. ELM-based models, nonetheless, improved over conventional models under these difficulties.

*Airlines Dataset:* On this dataset, OS-ELM using Hybrid ADWIN achieved an F1-score of 87.3%, consistent with its top position. The ELM and Decision Tree models also fared well when coupled with ADWIN, indicating the strength of adaptive drift detection for time-sensitive periodic data such as airline delays.

*Electricity Dataset:* This dataset yielded the best performance metrics overall, with OS-ELM and Hybrid ADWIN having a very high F1-score of 92.6%. ELM with ADWIN and Decision Tree with ADWIN also performed well. The resilience of ELM-based approaches under this dataset highlights their flexibility to market-driven, volatile data streams.

These results are tabulated in Table I, offering a comparative summary of each classifier–drift detection pair across all datasets.

To support the tabulated performance metrics in Table I, Fig. 5 presents a comparative visualization of classifier performance across four benchmark datasets: Airlines, Electricity, PokerHand, and SensorStream. Each subplot visualizes the Accuracy, Precision, Recall, and F1-score achieved by Decision Tree, ELM, Naive Bayes, OS-ELM, Random Forest, and SVM classifiers.

TABLE I: Classifier Performance across Datasets

| Dataset | Classifier | Drift Detection | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) |
|---|---|---|---|---|---|---|
| SensorStream | SVM | DDM | 84.5 | 83.2 | 81.7 | 82.4 |
| | Random Forest | ADWIN | 88.1 | 87.9 | 86.5 | 87.2 |
| | Naive Bayes | PH Test | 80.3 | 78.6 | 77.2 | 77.9 |
| | Decision Tree | CUSUM | 85 | 84.1 | 82.4 | 83.2 |
| | ELM | ADWIN | 86.7 | 85.5 | 84 | 84.7 |
| | OS ELM | Hybrid ADWIN | 91.2 | 90.7 | 89.4 | 90 |
| PokerHand | SVM | DDM | 74.5 | 72 | 70.1 | 71 |
| | Random Forest | CUSUM | 76.4 | 75.3 | 74 | 74.6 |
| | Naive Bayes | PH Test | 70.2 | 69.4 | 67.8 | 68.6 |
| | Decision Tree | ADWIN | 78.6 | 77.5 | 76 | 76.7 |
| | ELM | ADWIN | 79.1 | 78.2 | 76.4 | 77.3 |
| | OS ELM | Hybrid ADWIN | 83.8 | 83.1 | 81.9 | 82.5 |
| Airlines | SVM | PH Test | 81 | 80.4 | 78.7 | 79.5 |
| | Random Forest | DDM | 83.7 | 82.8 | 81.3 | 82 |
| | Naive Bayes | CUSUM | 77.5 | 76 | 74.3 | 75.1 |
| | Decision Tree | ADWIN | 82.1 | 81.2 | 79.8 | 80.5 |
| | ELM | ADWIN | 84 | 83.1 | 81.5 | 82.3 |
| | OS ELM | Hybrid ADWIN | 88.6 | 88 | 86.7 | 87.3 |
| Electricity | SVM | CUSUM | 85.3 | 84.1 | 83.2 | 83.6 |
| | Random Forest | PH Test | 87.4 | 86.2 | 85 | 85.6 |
| | Naive Bayes | DDM | 83.2 | 82.5 | 81 | 81.7 |

| | | | | | |
|---|---|---|---|---|---|
| Decision Tree | ADWIN | 89.1 | 88.2 | 87.1 | 87.6 |
| ELM | ADWIN | 89.8 | 88.9 | 87.9 | 88.4 |
| OS ELM | Hybrid ADWIN | 93.5 | 93.1 | 92.2 | 92.6 |

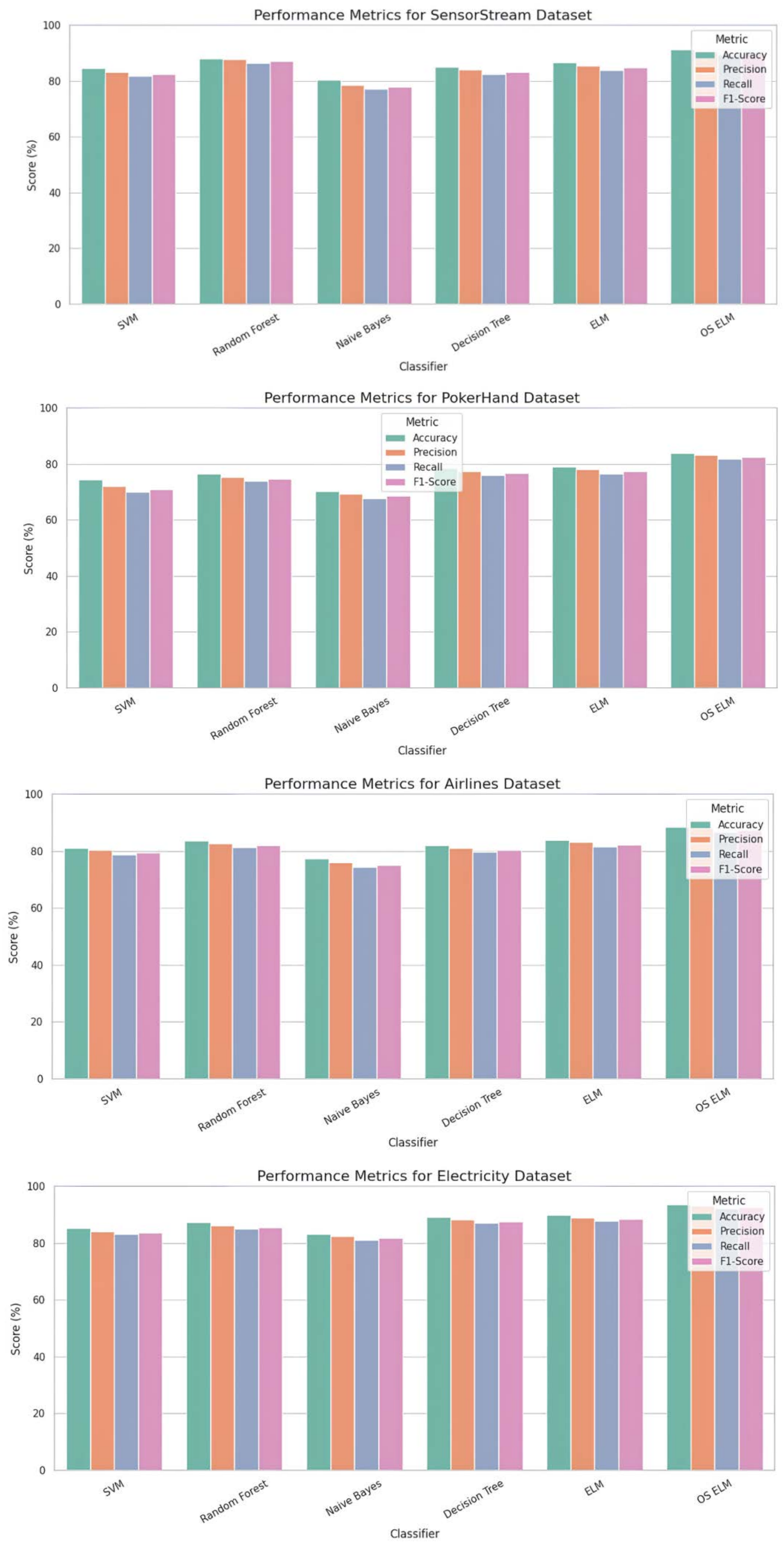


Fig. 5. Visual comparison across datasets and classifiers.

Notably, OS-ELM consistently outperforms other classifiers across most metrics, especially on the Airlines and Electricity datasets. Random Forest and SVM also maintain competitive performance on the SensorStream dataset. These results reinforce the robustness of the Hybrid ADWIN-based concept drift detection approach when paired with adaptive classifiers like OS-ELM. Overall, the trends observed validate the superiority of online sequential learning models in dynamic streaming environments.

## B. Classifier Comparisons and Observations

Overall, OS-ELM with Hybrid ADWIN was the top-performing setup on all four datasets. It had performance levels above 88% on all measures, reflecting high stability and responsiveness to sudden and gradual concept drifts. This attests to its fitness for use in dynamic environments like sensor-driven monitoring systems, online transaction streams, and real-time analytics.

ELM with ADWIN also did very well, typically being the second-best model. Random Forest with either ADWIN or DDM produced good results on most datasets, providing a good trade-off between performance and interpretability. Decision Tree models with ADWIN were consistent but performed slightly worse than ELM-based approaches.

Conversely, Naive Bayes classifiers, especially when used with the PH test, performed persistently poorly on datasets, confirming their poor ability to deal with shifting distributions. SVMs also performed poorly, especially on more complicated datasets like PokerHand, where data imbalance and class diversity may have impeded their learning capacity.

## C. Dataset-Specific Insights

Among all the datasets, the Electricity dataset recorded the best performance results, especially under the OS-ELM + Hybrid ADWIN model (93.5% accuracy), proving its appropriateness for real-time predictive analytics. The PokerHand dataset presented the most challenging scenario, as all classifiers had relatively lower marks. This can be explained by the high-class imbalance and complicated feature relationships in the dataset. The SensorStream and Airlines datasets had mid-to-high scores, with patterns similar to those in the Electricity dataset, thereby confirming the system's generalizability.

## D. Experimental Setup and Sensor Readings

All experiments were performed using the Google Colab environment. The real-time data used in this study was collected through an IoT-based experimental setup comprising sensors for pressure, temperature, humidity, rainfall, gas, and soil moisture. These readings were continuously streamed and visualized, and served as the primary source for testing the proposed hybrid drift detection framework.

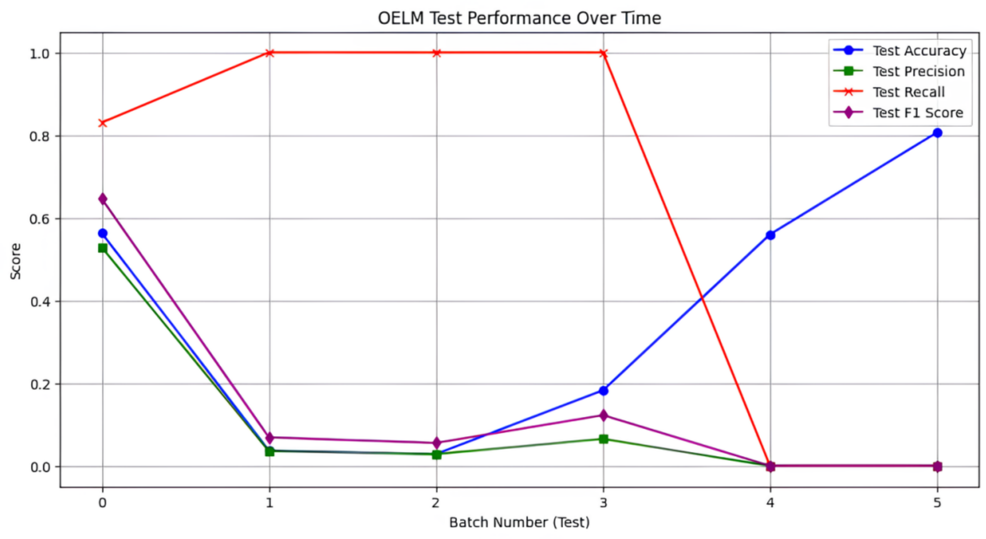


Fig. 6. OSELM without drift detection module.

## E. Performance with and without Drift Handling Module

Fig. 6 shows the performance of the OS-ELM classifier without the inclusion of the drift detection module. Even though the model works well up to batch 3, between batch 3 and batch 4, an observable accuracy deterioration occurs because the concept drift has not been caught. The training accuracy, precision, recall, and F1-score are all on average at 85.86%, 94.74%, 15.52%, and 26.67%, respectively. The classifier's accuracy significantly drops in testing to 36.30%, showing the classifier's high sensitivity to drift when no adaptation mechanisms are employed.

Contrarily, Fig. 7 illustrates the improved performance of OS-ELM coupled with the Hybrid ADWIN module. The classifier accurately identifies drift at various intervals (e.g., batches 1, 2, 4, 5, and 6), facilitating timely model updating. Consequently, the average training metrics are improved to 97.29% accuracy, 94.10% precision, 72.82% recall, and 75.58% F1-score. Corresponding test scores also significantly improve to 93.60% accuracy, 78.45% precision, 60.41% recall, and 61.03% F1-score.

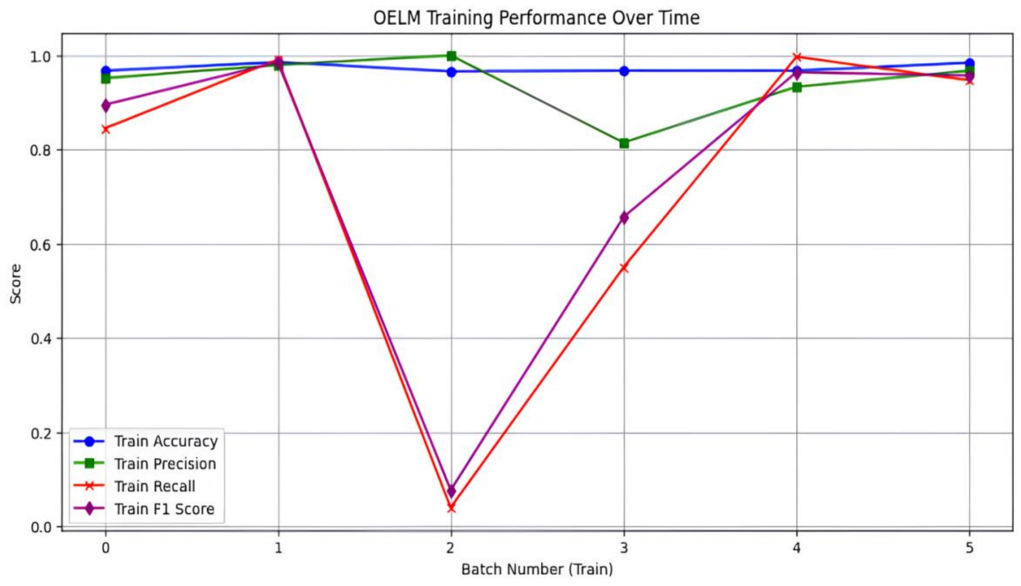


Fig. 7. OELM with drift detection module.

Table II and Table III show the performance measure of the classifier during the training and testing period, and it is observed that the system can identify the drift with the help of an external drift detection module and attempts to maintain the accuracy at 97.29%.

TABLE II: PERFORMANCE MEASURE DURING TRAINING

| Drift Handling Strategy | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| Without the drift module | 0.8586 | 0.9474 | 0.1552 | 0.2667 |
| With the drift module | 0.9729 | 0.9410 | 0.7282 | 0.7558 |

TABLE III: PERFORMANCE MEASURE DURING TESTING

| Drift Handling Strategy | Test Accuracy | Test Precision | Test Recall | Test F1 Score |
|---|---|---|---|---|
| Without the drift module | 0.3630 | 0.1097 | 0.6384 | 0.1489 |
| With the drift module | 0.9360 | 0.7845 | 0.6041 | 0.6103 |

The robust and persistent performance of the proposed OS-ELM with the Hybrid ADWIN framework illustrates its practicality in many real-time scenarios. Its capacity to sense and adjust to both sudden and gradual concept drift makes it especially apt for dynamic settings like streaming sensor data, finance, cybersecurity, and telecom. For example, in intrusion detection systems, emerging patterns of cyberattacks like phishing or DDoS attacks can be identified in advance with the help of adaptive drift-aware models. Likewise, industrial monitoring systems, where there are multiple sensors, appreciate early detection of sensor failure or signal degradation.

In weather forecasting, under which conditions in the environment may change immediately due to natural or human reasons, the suggested system guarantees model stability over time. News recommendation systems, spam filtering systems, and navigation systems are some other applications, all of which are required to learn and evolve continuously based on user actions or from the outside world. Experimental evidence consistently validates the system's scalability and flexibility and makes it a stable solution for intelligent decision-making under high-frequency, changing data settings.

## V. CONCLUSION

Real-time crop monitoring systems generate continuous streams of sensor observations that are extremely dynamic and affected by seasonal and environmental fluctuations. Static machine learning models tend to be incapable of preserving accuracy in such scenarios because of their inflexibility to adapt to concept drift. After a systematic review of existing methodologies, this work presents a hybrid ADWIN-based drift detection approach integrated with OS-ELM, which is an effective remedy to such constraints. The suggested system employs a multidimensional K-S test and Hoeffding's bound-based deviation analysis for identifying and responding to concept drift by dynamic window adjustment and model retraining.

Experimental verification on several benchmark datasets verifies the better performance of the proposed method. The OS-ELM classifier integrated with Hybrid ADWIN outperformed all other classifiers and drift detection combinations consistently, achieving a remarkable improvement in classification accuracy from 85.86 percent to 97.29 percent. These findings bring out the importance of incorporating drift-aware mechanisms in streaming data applications.

Besides agricultural monitoring, the proposed framework is useful to a wide variety of application domains, such as cybersecurity (Intrusion Detection System), weather prediction, telecommunications, and industrial process monitoring. Moreover, the system's lightweight and modular architecture enables practical deployment within real-world environments. It can be implemented efficiently on edge devices or cloud platforms (e.g., AWS Greengrass, Azure IoT Hub) using Docker containers or Python-based microservices, thereby supporting scalable integration into existing agricultural monitoring infrastructures. Future research could investigate the integration of deep learning models and online active learning approaches to further improve adaptability in more challenging and high-dimensional data streams.

## Conflict of Interest

The authors declare no conflict of interest.

## Author Contributions

Hezal Lopes: Conceptualization, Data Curation, Methodology, Software Implementation, Formal Analysis, Writing – Original Draft, Visualization. Prashant Nitnaware: Supervision, Project Administration, Methodology Review, Validation, Writing – Review & Editing, Resources. All authors had approved the final version.

## References

[1] S. Agrahari and A. K. Singh, "Concept drift detection in data stream mining: A literature review," *J. King Saud Univ. Comput. Inf. Sci*., vol. 34, no. 10, pp. 9523–9540, 2022. doi: 10.1016/j.jksuci.2021.11.006

[2] B. Chen, V. Huang, and C. Wang, "Sensor-drift-aware time-series anomaly detection for climate stations," in *Proc. 2024 IEEE Conf. Artif. Intell. (CAI)*, 2024. doi: 10.1109/CAI59869.2024.00228

[3] E. Garcıa-Huitzitl, L. Bustio-Martínez, and R. Cumplido, "Adaptive intrusion detection system: hybrid k-means and random forest approach with concept drift detection," *Comput. Systems*, vol. 29, no. 1, pp. 29–42, 2025. doi: 10.13053/cys-29-1-5528

[4] P. Porwik and B. M. Dadzie, "Detection of data drift in a two-dimensional stream using the Kolmogorov–Smirnov test," *Procedia Comput. Sci*., vol. 207, pp. 168–175, 2022. doi: 10.1016/j.procs.2022.09.049

[5] F. Yi and P. Qiu, "An adaptive CUSUM chart for drift detection," *Qual. Reliab. Eng. Int.*, vol. 38, no. 2, pp. 887–894, 2022. doi: 10.1002/qre.3020

[6] A. S. Palli, J. Jaafar, A. R. Gilal *et al*., "Online machine learning from non-stationary data streams in the presence of concept drift and class imbalance: A systematic review," *J. Inf. Commun. Technol.,* vol. 23, no. 1, pp. 105–139, 2024.

[7] H. Mehmood, A. Khalid, P. Kostakos *et al*., "A novel edge architecture and solution for detecting concept drift in smart environments," *Future Gener. Comput. Syst.*, vol. 150, pp. 127–143, Jan. 2024.

[8] G. Y. Sakurai, J. F. Lopes, B. B. Zarpelão, and S. Barbon, "Benchmarking change detector algorithms from different concept drift perspectives," *Future Internet*, vol. 15, no. 5, p. 169, 2023. doi: 10.3390/fi15050169

[9] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, "Exponentially weighted moving average charts for detecting concept drift," *Pattern Recognit. Lett*., vol. 33, no. 2, pp. 191–198, 2012.

[10] S. Zhan, Y. Li, C. Liu, and Y. Zhao, "Unsupervised concept drift detection based on stacked autoencoder and Page-Hinckley test," in *Green, Pervasive, and Cloud Computing*, *GPC 2023*, H. Jin, Z. Yu, C. Yu, X. Zhou *et al*., Eds., 2024. doi: 10.1007/978-981-99-9893-7_15

[11] S. Wang and F. Machida, "A robustness evaluation of concept drift detectors against unreliable data streams," in *Proc. 2021 IEEE 7th World Forum Internet of Things (WF-IoT)*, 2021. doi: 10.1109/WF-IoT51360.2021.9595202

[12] R. Richard and N. Belacel, "An online, adaptive and unsupervised regression framework with drift detection for label scarcity contexts," *CoRR*, 2023. doi: 10.48550/arXiv.2312.07682

[13] O. A. Mahdi, E. Pardede, and N. Ali, "A hybrid block-based ensemble framework for the multi-class problem to react to different types of drifts," *Cluster Comput*., vol. 24, no. 3, pp. 2327–2340, 2021.

[14] A. Angelopoulos, A. E. Giannopoulos, N. C. Kapsalis *et al*., "Impact of classifiers to drift detection method: A comparison," in *Proc. 22nd Eng. Appl. Neural Netw. Conf. (EANN)* . doi: 10.1007/978-3-030-80568-5_33

[15] S. Zhang, P. Tino, and X. Yao, "Hierarchical reduced-space drift detection framework for multivariate supervised data streams," *IEEE Trans. Knowl. Data Eng.,* 2021. doi: 10.1109/TKDE.2021.3111756

[16] M. S. Althabiti and M. Abdullah, "CDDM: Concept drift detection model for data stream," *Int. J. Interact. Mob. Technol*., vol. 14, no. 10, pp. 90–106, 2020.

[17] L. Yang and A. Shami, "A lightweight concept drift detection and adaptation framework for IoT data streams," *IEEE Internet Things Mag*., vol. 4, no. 2, pp. 96–101, 2021.

[18] J. Xuan, J. Lu, and G. Zhang, "Bayesian nonparametric unsupervised concept drift detection for data stream mining," *ACM Trans. Intell. Syst. Technol.,* vol. 12, no. 1, 2020. doi: 10.1145/3420034

[19] B. Krawczyk, B. Pfahringer, and M. Woźniak, "Combining active learning with concept drift detection for data stream mining," in *Proc. 2018 IEEE Int. Conf. Big Data*, 2018. doi: 10.1109/BigData.2018.8622549

[20] I. Žliobaitė, A. Bifet, B. Pfahringer, and G. Holmes, "Active learning with evolving streaming data," *Lect. Notes Comput. Sci.*, vol. 6913, 2011. doi: 10.1007/978-3-642-23808-6_39

[21] MOA Data Stream Datasets Classification. [Online]. Available: https://sourceforge.net/projects/moadatastream/files/Datasets/Classification/

[22] L. Frosini, M. Manca, and F. Paternò, "A framework for the development of distributed interactive applications," in *Proc. 5th ACM SIGCHI Symp. Eng. Interactive Comput. Syst. (EICS)*, pp. 249–254, Jun. 2013. doi: 10.1145/2494603.2480328

[23] Stream Data Mining. [Online]. Available: http://www.cse.fau.edu/~xqzhu/stream.

**Hezal Lopes** is currently pursuing a Ph.D. in computer engineering from Mumbai University. She holds an M.E. in computer engineering (2013) and a B.E. in computer engineering (2007) from Mumbai University and St. Francis Institute of Technology, respectively. Her areas of interest include artificial intelligence, machine learning, IoT, and data warehousing and mining. She has published several papers in international conferences focusing on smart agriculture, mobile security, and blockchain solutions for social welfare. She has also conducted expert sessions and co-authored a book on system programming and compiler construction.

**Prashant Nitnaware** received his Ph.D. from Mumbai University and holds an M.E. in information technology (2012) and a B.E. from Dr. Babasaheb Ambedkar Marathwada University, Aurangabad (2003). He is currently affiliated with Pillai College of Engineering, New Panvel. His research interests span artificial intelligence, data warehousing and mining, big data analytics, usability engineering, and human-computer interaction. Dr. Nitnaware has published extensively in reputed journals and conferences and has served as a resource person for various academic and professional programs in areas such as cyber security, oracle SQL, and engineering education.