



# OPEN Smart adaptive ensemble model for multiclass imbalanced nonstationary data streams

Abdul Sattar Palli<sup>1,9</sup>✉, Jafreezal Jaafar<sup>1,2</sup>, Mohamad Hanif Md Saad<sup>3</sup>, Ainul Akmar Mokhtar<sup>4</sup>, Heitor Murilo Gomes<sup>5,6</sup>, Afzal Ahmed Soomro<sup>7</sup>✉ & Abdul Rehman Gilal<sup>8</sup>

In real-time streaming data, concept drift and class imbalance may occur simultaneously which causes the performance degradation of the online machine learning models. Most of the existing work is limited to addressing these issues for binary class data streams. Very little focus is given to the multi-class data streams. The most common approach to address these issues is ensemble learning. Ensemble learning consists of multiple classifiers combined which are trained on different subsets of the data to improve the overall accuracy. The performance of the ensemble learning approach suffers in case the new classifier is not trained on appropriate data (the data about the new concept). To address this gap, this study has proposed a Smart Adaptive Ensemble Model (SAEM) to address the issues of concept drift and class imbalance for multi-class data streams. The SAEM monitors the feature-level change in data distribution and creates a background ensemble to train the new classifier on features that observe change. To address the class imbalance issue, SAEM applies higher weights on the minority class instances using the dynamic class imbalance ratio. The proposed model outperformed the existing state-of-the-art approaches on the eight different data streams. The results showed an average improvement of 15.857% in accuracy, 20.35% in Kappa, 16.12% in F1-score, 15.58% in precision, and 16.42% in recall. The Friedman test confirmed statistically significant performance differences among all models across five key metrics. Based on the obtained results, the research findings strongly support the notion that SAEM exhibits enhanced effectiveness and efficiency as a solution for online learning applications.

**Keywords** Concept drift, Multi-class imbalance, Non-stationary data stream, Concept adaptation, Online machine learning

Advancements in technology have greatly increased the ability to generate and collect data in real-time, particularly in the areas of IoT (Internet of Things), 5G, and cloud computing<sup>1</sup>. These technologies have enabled the deployment of a vast number of connected devices and sensors, as well as the ability to process and store large amounts of data. This has led to an explosion in the amount of data being generated and collected in real-time, from various sources such as social media, e-commerce transactions, and sensor data from industrial systems. The applications such as, monitoring remaining useful life in an industrial environment<sup>2–4</sup>, monitoring the traffic within the city such as traffic congestion<sup>5</sup>, whether prediction<sup>6</sup>, monitoring customer satisfaction in banking<sup>7</sup>, forecasting the share values in stock markets<sup>8</sup>, education<sup>9</sup>, telecommunications<sup>10</sup>, healthcare<sup>11</sup>, network data monitoring<sup>12</sup> in computer-based distributed applications all are common examples of applications that generate data streams. As a result, the importance of real-time analysis of data streams is growing as the number of applications in this field increases.

<sup>1</sup>Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, 32610 Seri Iskandar, Perak Darul Ridzuan, Malaysia. <sup>2</sup>Centre for Research in Data Science, Universiti Teknologi PETRONAS, 32610 Seri Iskandar, Perak Darul Ridzuan, Malaysia. <sup>3</sup>Department of Mechanical and Manufacturing Engineering, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia. <sup>4</sup>Department of Mechanical Engineering, Universiti Teknologi PETRONAS, 32610 Seri Iskandar, Perak Darul Ridzuan, Malaysia. <sup>5</sup>School of Engineering and Computer Science, Victoria University of Wellington, Wellington 6012, New Zealand. <sup>6</sup>AI Institute, University of Waikato Wellington, Hamilton 3240, New Zealand. <sup>7</sup>Interdisciplinary Research Center for Industrial Nuclear Energy (IRC-INE), King Fahd University of Petroleum and Minerals (KFUPM), 31261 Dhahran, Saudi Arabia. <sup>8</sup>Knight Foundation School of Computing and Information Sciences, Florida International University, Florida, FL 33199, USA. <sup>9</sup>Anti-Narcotics Force, Ministry of Interior and Narcotics Control, Islamabad 46000, Pakistan. ✉email: abdulsattarpalli@gmail.com; afzals.09in09.mueta@gmail.com

The analysis of such data streams requires special algorithms that can classify the continuous data as soon as it is received without any delay. Online Machine Learning (OML) is a branch of machine learning that deals with continuous data streams with data being available only sequentially, in the form of a stream. In OML, the ML model continuously learns from new data instances and updates itself as it classifies them. The continuous data generated by these applications is non-stationary. The non-stationary data stream has specific characteristics such as data being inconsistent which causes a change in data distribution, is infinite in length, and it keeps evolving over time. This non-stationary nature of the data causes a change in the underlying concept of data over time called Concept Drift (CD) and a change in class distribution called Class Imbalance (CI)<sup>13–16</sup>. Due to the non-stationary nature of the data stream, current online machine learning approaches suffer from issues of concept drift and class imbalance especially in multi-class data, as illustrated in Fig. 1.

These issues are addressed independently, and very little focus has been given to the combined issue (when both issues occur simultaneously), especially for multi-class data streams. Hence, there is a need to further explore this research domain<sup>13,17</sup>.

The novel Smart Adaptive Ensemble Model (SAEM) is proposed in this study to address the joint issue of concept drift and class imbalance in multi-class data streams. The SAEM is based on ensemble approach, as the ensemble approach is considered better approach than single-class approach for continuous learning from streaming data<sup>18</sup>. In SAEM, each member of the ensemble is initially trained on a random feature subset to maintain the diversity among the classifiers. The random feature subset is simply the random selection of features from the original feature space. The size of the feature subset is based on the hyper parameter; however, the default feature subset size is 70% of the total feature space. The idea of using random feature space has also been discussed in other studies such as<sup>19,20</sup>. The SAEM uses input feature-based and output-based drift detection mechanism to counter both the virtual drift and the real drift of any type such as sudden drift, incremental drift, gradual drift, and recurring drift. The feature-based drift detection not only help in early drift detection but also help in preparing a training set containing drifted features along with random features for appropriate adaptation of the ensemble. The SAEM also addresses the multi-class imbalance issue using the dynamic weight-based approach which is based on the dynamic class imbalance ratio of each class. The core contribution of SAEM which makes it different from other ensemble approaches is its feature-based drift detection; feature subset preparation mechanism; and building a smart background ensemble; to adapt the new concept efficiently.

In the general following are the key features of the proposed model:

- A novel feature-level drift detection mechanism, enabling drift-aware feature selection for classifier updates.
- A smart background ensemble update strategy, using the most drift-relevant features.
- Dynamic class-level imbalance monitoring, which uses both current and cumulative imbalance ratios to assign instance-level weights.

These contributions collectively enhance the model's adaptability to evolving data streams with imbalanced class distributions and frequent concept drift.

The work presented in this manuscript focuses on developing an adaptive model that can identify new concepts in the streaming data and update the underlying classifier to correctly classify the new concept of the streaming data. For example, in healthcare applications, the model can detect new diseases or treatment methods and adapt the classifier to correctly identify these new concepts in the patient data. The outcome of this work may benefit real-time applications that use continuous data streams include Healthcare<sup>11,37,38</sup>; Manufacturing Industry<sup>2–4,21,22</sup>; Traffic Monitoring Systems<sup>5</sup>; Recommendation Systems<sup>39</sup>; Fraud Detection and Security<sup>40</sup>; Automated Data Analysis<sup>41</sup>; Finance<sup>7,8</sup>; and Telecommunication<sup>10,12</sup>.

## Related work

According to<sup>17</sup>, learning from data stream which has both concept drift and class imbalance issues, requires three steps or phases in OML environment. These are 1) Concept Drift Detection, Class Imbalance Handling, and 3) Concept Drift Adaptation. The discussion on each of these steps is given in following subsections.

## Concept drift detection

A phenomenon known as concept drift describes a circumstance in which the statistical properties of a target class fluctuate randomly over time. These modifications may be the result of changes in elements that cannot be easily quantified or identified. Two types of concept drift exist virtual drift and real drift<sup>23</sup>. Virtual drift may be thought of as a shift in the conditional probabilities  $P_t(y | X)$ , whereas real drift can be thought of as a shift in the unconditional probability distribution  $P_t(X)$ . The virtual drift occurs when the distribution of input data  $P_t(X)$  changes over time, but the posterior probability of the output  $P_t(y | X)$ , which shows the mapping relationship between  $X_t$  and  $y_t$ , doesn't change with time. In real concept drifts, the posterior



**Fig. 1.** Issues of concept drift and class imbalance in non-stationary data streams.

Aspect	Existing					(Proposed)
	AWE	AUE	KUE	CALMID	ROSE	SAEM
Feature subset strategy	Full feature space	Full feature space	Random subset	Full feature space	Random subset (variable size)	Drift-based feature subset
Drift detection	None	Implicit (via errors)	Kappa-based	Adaptive ensemble update	Background ensemble drift detection	Multi-level: feature-level + performance-based
Drift adaptation	Classifier replacement	Weight update	Classifier selection via Kappa	Adaptive sampling + ensemble update	Background ensemble update	Background ensemble training using Drift-based feature subset
Imbalance handling	Not addressed	Not addressed	Not addressed	Proactive sampling of minority class	Per-class sliding windows	Dynamic + cumulative imbalance ratio-based weighting
Learning type	Chunk-based	Incremental	Hybrid (online + block)	Active learning	Online	Online

**Table 1.** Comparative summary of SAEM with existing methods for handling concept drift and class imbalance in data streams.

probability distributions  $P_t(y \mid X)$  change over time, but this change is not caused by changes in  $P_t(X)$ . In contrast, hybrid drift is the combination of both virtual drift and real drift. In simple words, the data, even after facing changes in input features, it represents to the different target output. Mathematically, concept drift can be defined as a change in the distribution  $P(X, y)$  of the input-output pairs  $(X, y)$  over time, where  $X$  represents the features and  $y$  represents the target variable<sup>24–26</sup>. According to<sup>23,27,28</sup> the major types of drift are further classified into four different ways based on the time frequency, such as (1) sudden drift (2) gradual drift (3) incremental drift, and (4) recurring drift.

A classifier’s performance is affected when the concept of the data or target output changes. The primary objective of concept drift detection methods is to offer an efficient approach that collaborates with the classification model, recognizing instances of drift or novelty when there is a significant alteration in data characteristics<sup>23,29</sup>. This approach ensures that the classification model is updated accordingly, preventing it from being impacted by the change and thereby enhancing its predictive performance. These established drift detection techniques are typically classified into three categories<sup>28</sup>: those reliant on error rates<sup>30–32</sup>; those centered on data distribution<sup>33</sup>, and those based on multiple hypothesis tests or statistical tests<sup>34,35</sup>. The detailed discussion of these techniques is out of the scope of this study, the literature studies<sup>27,36</sup> can be referred for further details.

**Class imbalance handling**

Class imbalance, on the other hand, refers to the situation where the number of instances of one class is significantly higher than the number of instances of another class. This can lead to bias in the learning process of the ML model, as the majority class will be more heavily represented in the training data, hence, the ML model gives poor performance to the minority class<sup>37,38</sup>.

Class imbalance is a problem in data of many real-time applications where the data distribution among the classes is not the same. Instances in one class are significantly underrepresented known as minority class in comparison to instances in another class having a huge amount of data samples known as majority class<sup>39</sup>. If a classifier is trained on imbalanced data it becomes biased toward majority class instances and misclassifies the minority class instances<sup>40</sup>. Various methods have been suggested to tackle the problem of enhancing the classifier’s performance specifically for the minority class. These methods can be generally categorized into four groups: Data-level strategies<sup>41,42</sup>; Algorithm-level strategies<sup>43,44</sup>; Ensemble strategies<sup>45,46</sup>; and Hybrid strategies<sup>47–49</sup>.

The Jiang, Zhen, et al. in<sup>50</sup> proposes a novel approach to address the challenges posed by class-imbalanced datasets. The method leverages a semi-supervised learning framework combined with resampling techniques to improve the classification performance of minority class samples. One of the key advantages of this approach is its ability to effectively utilize both labeled and unlabeled data, reducing the reliance on labeled samples, which are often scarce in imbalanced datasets. However, a potential downside is that the effectiveness of the method may be limited in extremely imbalanced scenarios or when the quality of unlabeled data is low, which can lead to overfitting or inaccurate decision boundaries.

Another study<sup>51</sup> introduces an innovative approach that combines boosting with co-training to address the challenges of class imbalance in machine learning. The method enhances the performance of classifiers by iteratively improving the accuracy of both the minority and majority class predictions through collaborative learning. One of the major strengths of this approach is its ability to exploit the strengths of multiple classifiers, leading to improved robustness and accuracy in imbalanced datasets. However, the method may be computationally expensive due to the iterative nature of boosting and may face challenges in scenarios where the individual classifiers perform poorly, potentially impacting the overall performance.

In a recent work<sup>52</sup>, authors proposes a post-processing technique to improve the performance of classifiers on imbalanced datasets, particularly in a transductive learning context. The framework refines the model’s output by adjusting the decision thresholds based on the classifier’s confidence scores, enhancing the classification of minority class instances. One of the key benefits of this approach is its simplicity and flexibility, as it can be applied to a wide range of existing classifiers without the need for significant modifications. However, a potential drawback is that its effectiveness depends heavily on the quality of the initial model and may not perform well in highly complex or noisy datasets where the classifier’s confidence scores are less reliable.

Dai et al., in<sup>53</sup> introduces a heterogeneous clustering ensemble method to detect class overlap in multi-class imbalanced datasets. While effective in addressing class overlap, the method is tailored for static datasets and does not account for concept drift in evolving data streams.

The study<sup>54</sup> presents GQEO, an oversampling technique utilizing nearest neighbor graphs and generalized quadrilateral elements to address class imbalance. While GQEO effectively improves minority class representation, it is developed for static datasets and does not consider the challenges posed by concept drift in streaming data.

In a more recent study<sup>55</sup>, the authors propose a mutually supervised heterogeneous selective ensemble framework based on matrix decomposition to tackle class imbalance. Although the approach enhances classification performance in imbalanced settings, it is primarily designed for static environments and lacks mechanisms to adapt to concept drift in dynamic data streams.

### Concept drift adaptation

Concept drift adaptation refers to the process of adjusting or updating a machine learning model to accommodate changes in the underlying data distribution over time. The concept drift adaptation approaches can be further classified in many forms, including online learning algorithms<sup>56–58</sup> versus batch learning algorithms<sup>59,60</sup>; single classifier<sup>61</sup> versus ensemble-based or multiple classifiers approaches<sup>62–65</sup>.

The single classifier approach retrains the existing classifier on the latest data to adapt the new concept like<sup>66,67</sup>. These approaches get the knowledge on the latest data and forget the knowledge learned from old data. An ensemble approach combines multiple models to improve the performance of a machine learning system. This can be done in a variety of ways, such as by averaging the predictions of multiple models, by training a meta-model to make predictions based on the outputs of multiple models, or by training multiple models and selecting the one that performs best on a validation set. Ensemble approaches can be used to improve the accuracy, robustness, and generalization of a machine learning system<sup>20</sup>.

In machine learning, due to the class imbalance, it is very difficult for the classifier to learn tiny class events in cases of skew data or infrequent or uneven class distribution<sup>68</sup>. In situations where both issues (concept drift and class imbalance) occur simultaneously, the classification performance is more affected<sup>69–71</sup>. The complex multi-class imbalance with other factors such as class overlapping, and rare examples cause more difficulty in retraining the new classifier to address the changing behavior of data<sup>72</sup>.

Besides this, the change may occur in multiple classes, and the number of instances of each class may also vary, leading to a class imbalance problem<sup>13,73–75</sup>. In case the stream is multi-class, the changes may occur in multiple classes simultaneously; this becomes more challenging for the drift detection methods to detect the new concept and adapt the existing model to the latest concept<sup>76</sup>.

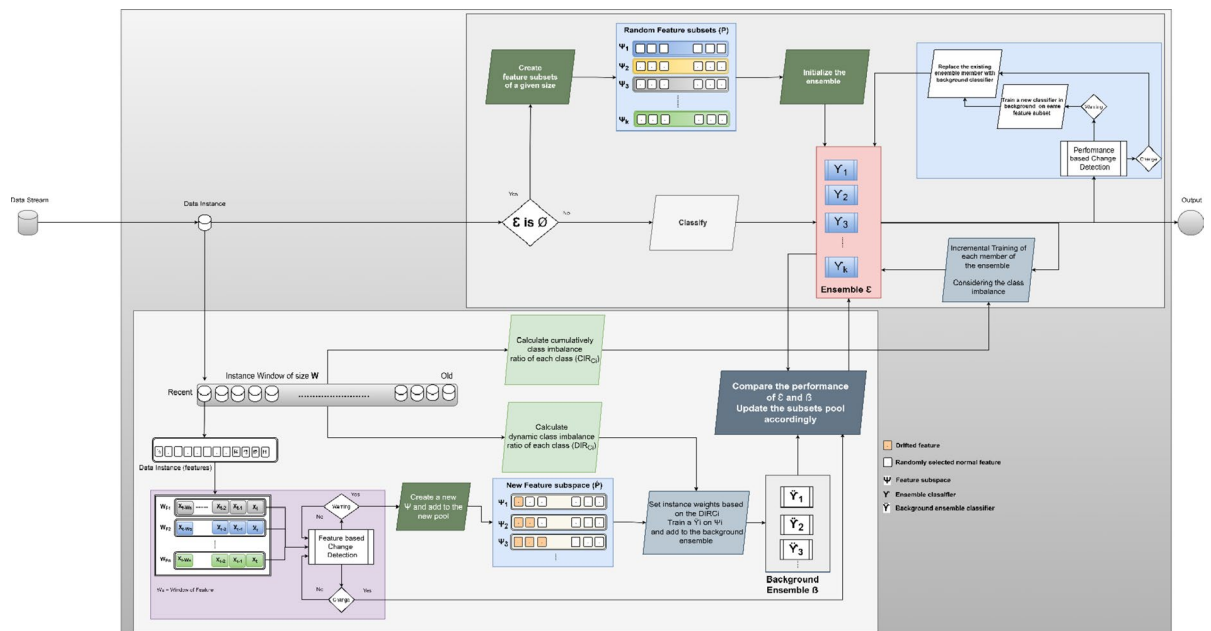
In the OML environment, where the data is non-stationary, there is a higher probability that the problem of class imbalance and concept drift can occur together (simultaneously), due to which the deployed models suffer in maintaining the required accuracy<sup>17,77–79</sup>. During online learning, there are constraints of limited data, time, and memory for retaining the new model<sup>80</sup>.

The ensemble approaches are considered as the most appropriate for concept adaptation. However, ensemble approaches mostly train a new classifier on the latest data using the window-based approach. In case of gradual drift, the window may not fully contain the latest concept. The other method ensemble approaches apply is to train a new classifier on a random feature set to maintain the diversity among the ensemble members. In case the drift occurs in some of the features (not all), the selection of a random feature set can miss the features of real interest which will ultimately causes inappropriate concept drift adaptation<sup>81</sup>. Also, it is difficult it detect and adapt the concept drift when the data has multiple imbalance classes<sup>72</sup>.

### State-of-the-art methods

Details of the approaches used in comparison to validate the proposed model SAEM is given below. These approaches are considered as the benchmark in the data stream learning approaches<sup>82</sup>.

1. *Accuracy Weighted Ensemble (AWE)*<sup>83</sup>: Reads the data streams in chunks and adds a new classifier after every chunk and removes the old classifier. In AWE, the weight of classifiers are actually the error rates produced during testing on the latest data chunk. It selects the classifiers with low error for the prediction of the new data chunk.
2. *Accuracy Update Ensemble (AUE)*<sup>84</sup>: AUE uses the incremental learning classifier to continuously update the weights of the classifier as well as incrementally update the classifier on each data sample. It initially applies the weights on each classifier and updates the weights after every chunk based on the error of the classifier. It removes the classifier with high error rate.
3. *Kappa Updated Ensemble (KUE)*<sup>85</sup>: The KUE is an ensemble learning method that combines online and block-based approaches. KUE utilizes the Kappa statistic to dynamically weight and select base classifiers. To increase the diversity of base learners, each classifier is trained using a unique subset of features, and they are updated with new instances based on a Poisson distribution. Also, the ensemble is only updated with new classifiers when they positively contribute to improving its quality.
4. *Comprehensive Active Learning method for Multiclass Imbalanced streaming Data with concept drift (CALMID)*<sup>86</sup>: CALMID introduces a comprehensive active learning method for handling multiclass imbalanced data streams with concept drift. The method combines proactive sampling adaptive ensemble modeling, and uncertainty-based instance selection to improve classification performance in real-time environments. The proactive sampling strategy is employed to select instances from the minority class and conceptually relevant instances from the majority class, ensuring a balanced representation of classes. The ensemble adapts to concept drift by continuously updating the pool of classifiers with new instances and adjusting their weights accordingly.



**Fig. 2.** Smart adaptive ensemble model.

5. *Robust Online Self-Adjusting Ensemble (ROSE)*<sup>20</sup>: ROSE is an innovative online ensemble classifier that can effectively handle various challenges. Its main characteristics are as follows: (i) it trains base classifiers online on random subsets of features with varying sizes; (ii) it can detect concept drift online and create a background ensemble to enable faster adaptation to changes; (iii) it employs a sliding window per class approach to build classifiers that are insensitive to skew, irrespective of the current imbalance ratio; and (iv) it utilizes self-adjusting bagging to increase the exposure of challenging instances from minority classes.

How SAEM differs from and improves upon existing methods. A comparative summary is provided in Table 1 below to underscore these differences:

This comparison highlights the core innovations of SAEM. Unlike existing approaches, SAEM employs a feature-level drift detection mechanism that allows for targeted classifier updates using only the most relevant features, rather than relying on random retraining. Furthermore, its dual-level drift detection strategy, combining performance-based and feature-level drift detection mechanisms enhances its ability to handle a variety of concept drift types more effectively. In addition, SAEM introduces a dynamic imbalance handling approach that adjusts instance weights based on both current and cumulative class imbalance ratios, allowing it to adapt to changes in class distribution over time. Together, these innovations contribute to improved classification performance and more efficient learning in dynamic, imbalanced data stream environments. The detailed description of proposed model is provided in the next section.

## Materials and methods

In this section we propose a novel Smart Adaptive Ensemble Model (SAEM) to address the joint issue of concept drift and class imbalance in multi-class data streams. The detailed description of each feature of SAEM is given in following sub sections.

## Ensemble architecture

The SAEM uses the ensemble approach to classify the data streams. The size of the ensemble or the number of classifiers in an ensemble is based on the input hyper parameters. The default size for the ensemble is 10, similar to the ROSE<sup>20</sup>. The base classifier used in ensemble approach is Hoeffding Tree (HD) classifier<sup>87</sup> which is considered as the most suitable base classifier because of its incremental learning approach<sup>20,88</sup>. The initial training of each member of the ensemble is performed on a random feature subset to achieve diversity in the ensemble members. In other words, SAEM builds each base classifier  $\Upsilon_j$  of ensemble  $\epsilon$  on a random  $\eta$ -dimensional feature subset  $\Psi_j$ , from the original  $m$ -dimensional space in the data stream  $S$ , where  $1 \leq \eta \leq m$ . The  $\eta$ -dimensional subset of features  $\Psi_j$  is randomly generated for each base learner  $\Upsilon_j$ . It allows to generate diverse feature subspaces of uniform size. Contrast to the ROSE which uses the variable size feature spaces for each member of the ensemble. The diversity of ensemble is mainly based on the set of features selected for the feature subset or feature space, not on the feature space size. Although keeping the variable size of feature space may further add the diversity in ensemble but it may affect the classifier replacement mechanism in ensemble adaptation phase in case the new classifier is trained on a feature subset which does not include the drifted features. The classifier replacement is a mechanism of comparing the performance of ensemble members with the members of the background ensemble and replace the poor performers of ensemble with the best performers of background ensemble. We will use the notation of an ensemble  $\epsilon$  of  $k$  number of base classifiers  $\Upsilon$  each trained on  $\eta$ -dimensional random



feature subspace  $\Psi_R$  which is subset of  $m$  dimensional space  $X$  of stream  $S$ , Given in (1) and (2). Where,  $\mathcal{F}_R^1$  is first randomly selected feature,  $\mathcal{F}_R^2$  is the second randomly selected feature, and so on.

$$\varepsilon = \{ \Upsilon_1, \Upsilon_2, \dots, \Upsilon_k \} \quad (1)$$

$$\Psi_R = [\mathcal{F}_R^1, \mathcal{F}_R^2, \mathcal{F}_R^3, \dots, \mathcal{F}_R^n] \quad (2)$$

In OML, the early detection of concept drift helps the base classifier to timely learn the new concept. Most of the existing drift detection methods are based on the performance of the classifier<sup>89</sup>. Those methods monitor the error rate of the classifier if it increases and reaches a defined threshold, they consider it as a drift and update the classifier. This approach works well for the sudden drift type where the error rate of the classifier suddenly jumps in case of arrival of new concept. These approaches struggle to timely detect the gradual and incremental drifts. The reason is in both cases the performance of the classifier decreases slowly. As a result of that, it takes longer time to detect the gradual and incremental drifts using the error-rate based drift detection approach. Therefore, this study uses both an output-based approach to detect sudden drift, and data distribution-based approach to timely detect the gradual and incremental drifts.

### Output based drift detection

Output-based drift detection methods focus on monitoring the output of the predictive model to detect changes in the data distribution. The goal of output-based drift detection is to detect changes in the output of the model between two time windows, which may indicate a drift in the output data distribution. These methods either monitor the error rate or the performance of the classifier such as accuracy or f1 score. Sometimes it is possible that a drift may occur even if the data distribution remains unchanged. This can happen if there is a change in the relationship between the input features and the output variable, which can affect the performance of the model. For example, let's say a predictive model that is trained to predict the price of a stock based on its historical data. The data distribution remains the same over time, but a major news event occurs that affects the price of the stock. This news event can change the relationship between the input features and the output variable, and the model may no longer be able to accurately predict the price of the stock.

In this case, a drift has occurred even though the data distribution remains unchanged. Output-based drift detection methods may detect this drift by monitoring the accuracy or other performance metrics of the model, even though the data distribution remains the same. Therefore, this study also uses output-based or performance-based drift detection as a second-level drift detection to avoid performance degradation in such situations. We use ADaptive WINdowing (ADWIN)<sup>59</sup> to monitor the performance of each classifier based on the correct and incorrect ratio. If the performance of the classifier  $\Upsilon$  decreases to certain level  $\delta w$ , we create a new copy of the classifier as  $\Upsilon'$  and start training this classifier in background parallel to the original classifier  $\Upsilon$ . We train the new background classifier on the same feature space  $\Psi$  on which the original classifier generated warning. We do not generate a new feature subset using the random feature selection like the existing approaches such as ROSE do. Again, the reason is simple, we believe a new model should be trained on the feature space which causes performance to decrease due the change in data distribution. Generating a new random feature space may have entirely different feature space. When the same classifier generates the signal for drift  $\delta d$ , we replace the classifier  $\Upsilon$  with the newly trained classifier  $\Upsilon'$  and link it with this particular  $\Psi$  in the feature space pool  $P$ .

### Feature-based drift detection

The data distribution-based drift detection approaches are considered more effective in early detection of the concept drift. Most of these approaches are based on data similarity and dissimilarity. The study<sup>90</sup> proposed feature based drift detection over a complete feature subset of defined chunk size. But this approach is not suitable for real-time online incremental online learning which requires to learn on the instance level not the chunk level. The distribution-based approaches produce effective results in detecting the real concept drift where the change occurs in the class boundary. In case the change occurs in the distribution of the data that does not affect the class boundary, this change is hard to detect with the data distribution-based drift detection methods. The data-distribution based approaches which detect drift from complete feature space may struggle in a situation where data changes for certain features only that does not affect much to the data distribution at complete feature space. Therefore, this study applies the feature-based drift detection method which helps in detecting drift even if it occurs in only single feature.

This study uses DDM<sup>91</sup> to monitor the statistical change in data distribution at each feature separately on incremental basis (instance-level). To initialize DDM object we use one-time fixed window  $\mathcal{W}$  of size 500 instances. After that it monitors the feature level data distribution on arrival of every instance. This study uses two threshold values to indicate the change in data distribution. One represents the warning level  $\mathcal{F}w$  and the other represents the drift level  $\mathcal{F}d$  as given in (3) and (4). Both threshold values are based on the user input as hyperparameter. How we deal such drift once warning is generated, we discuss this in concept adaptation sub section.

$$\text{Warning} : P_{t_0}(X, y) - P_{t_1}(X, y) \gg \mathcal{F}w \quad (3)$$

$$\text{Drift} : P_{t_0}(X, y) - P_{t_1}(X, y) \gg \mathcal{F}d \quad (4)$$

### Class-wise dynamic and cumulative imbalance ratio monitoring

There are several approaches used in online data stream learning to address class imbalance during drift adaptation. The most common are Bagging and Boosting<sup>92</sup>. These are two popular ensemble methods that are

used in online data stream learning to address class imbalance during drift adaptation. Bagging (Bootstrap Aggregating) is an ensemble method that involves training multiple classifiers on random subsets of the data and then combining their outputs to make predictions. In the context of class imbalance, bagging can be used to reduce bias towards the majority class by training classifiers on balanced subsets of the data.

Boosting is an ensemble method that involves training multiple classifiers sequentially, with each subsequent classifier focusing on the instances that were misclassified by the previous classifiers. Boosting can be effective in handling class imbalance by giving higher weight to the misclassified instances from the minority or by giving higher weights to the classifier which produces more correct prediction. Adaptive boosting (AdaBoost) is a commonly used variant of boosting that can be adapted to online data stream learning.

Instead of training the classifier on balanced subsets, applying higher weights on misclassified instances or classifier, the SAEM uses the imbalance ratio of a class to assign the weights to its instances. SAEM monitors dynamic imbalance ratio (calculated over the current data window) as well as cumulative imbalance ratio (calculated from the beginning of the stream up to the current window) of each class. Dynamic imbalance ratio or current imbalance ratio is the imbalance ratio calculated on the current batch or window of data using (5). The advantage of using a dynamic imbalance ratio is that it can adapt to changes in the class distribution over time.

$$DIR_{Ci} = \frac{W - W_{N_{Ci}}}{W_{N_{Ci}}} \quad (5)$$

where  $DIR_{Ci}$  is the dynamic imbalance ratio of class 'Ci' and  $N_{Ci}$  is the number of instances of class 'Ci', so  $W_{N_{Ci}}$  is the number of instances of class 'Ci' in current window. The  $W$  is number of instances in current window. SAEM calculates the class imbalance ratio after every  $W$  instances and subsequently updates the weights of class instances using the (6) during the concept adaptation phase while training the background ensemble.

$$\bar{w}_{Ci} = DIR_{Ci} + \mathcal{C} \quad \text{Where } 0 \leq \mathcal{C} \leq \lambda \quad (6)$$

The  $\bar{w}_{Ci}$  is the weight of instances of Class Ci, and  $\mathcal{C}$  is random constant value based on the input hyper parameter  $\lambda$ . On the other hand, the cumulative imbalance ratio is the imbalance ratio calculated from the beginning of the data stream up to the current time using (7). The advantage of using a cumulative imbalance ratio is that it provides a long-term perspective on the class distribution and can capture trends and patterns over time.

$$CIR_{Ci} = \frac{N_T - N_{Ci}}{N_{Ci}} \quad (7)$$

The  $CIR_{Ci}$  is cumulative imbalance ratio of class Ci,  $N_T$  is total number of instances processed so far, and  $N_{Ci}$  is total number of instances of class Ci seen so far. SAEM uses the cumulative imbalance ratio to update the weights of class instances using the (8) during the incremental learning of the main ensemble.

$$\bar{w}_{Ci} = CIR_{Ci} + \mathcal{C} \quad \text{Where } 0 \leq \mathcal{C} \leq \lambda \quad (8)$$

Combining dynamic and cumulative imbalance ratios can be a useful approach in situations where the class distribution in the data stream is both changing over time and exhibits long-term trends. Therefore, SAEM uses cumulative imbalance ratio to address the class imbalance during incremental training of ensemble members, whereas it uses dynamic imbalance ratio to apply higher weights to the minority instances of current window during the training of classifier of background ensemble. The reason is, the background classifiers are trained on more recent data to adapt the new concept, therefore class imbalance ratio of recent data is considered. Whereas the ensemble members also have knowledge about the old concepts therefore we consider cumulative imbalance ratio for updating the ensemble members. By combining the two measures, we can obtain a more accurate estimate of the class distribution and better adjust the weights assigned to instances.

### Concept drift adaptation

The concept drift adaptation is the mechanism to learn the new concept from recent data and update the machine learning model to avoid the performance degradation during online classification of data streams<sup>93</sup>. The background ensemble is an ensemble learning technique that can be used in online data stream learning to address the concept drift during drift adaptation. The idea behind background ensemble is to maintain a background set of classifiers that are trained on past data and use them to provide additional support to the current classifier during drift adaptation.

In a typical background ensemble, the new classifier is trained on the most recent data and is responsible for making predictions on the incoming data stream. However, when a drift is detected or when the class distribution changes significantly, the background ensemble is activated, and the predictions of the current classifier are combined with those of the background classifiers to improve accuracy and adapt the concept drift and class imbalance. The background ensemble can consist of several classifiers trained on past data and stored in memory. These classifiers can be trained using various methods, such as random subspace method, or random incremental ensemble. The random subset sometimes may not contain the features which observed the change in data distribution. This will cause poor adaptation of the new concept.

The model proposed in this study introduces novel approach for generating feature subset for each classifier  $\tilde{Y}$  of background ensemble  $\mathcal{B}$ . The feature subset generated by SAEM will always have the features where the change in data distribution has occurred. The background ensemble  $\mathcal{B}$  is empty ( $\emptyset$ ) initially, once a feature

level warning is generated on any of the feature  $\mathcal{F}$ , it generates a new feature subset  $\Psi$  containing the feature on which a warning is generated, and the randomly selected features. This new feature subset is used to trains a new background classifier  $\Upsilon$ . Later, if a warning is generated on another feature, a new feature subset is generated containing the two features on which warning is generated and the rest features are randomly selected to meet the size of the feature subset. In SAEM, background ensemble  $\mathcal{B}$  contains the classifiers which are trained on smartly generated feature subsets.

The performance of  $\epsilon$  and  $\mathcal{B}$  classifiers is compared after (i) an interval of every  $\mathcal{W}$  instances; (ii) a feature-based drift detector generates a drift signal. The  $\epsilon$  is updated with the best performing classifiers from both main ensemble and background ensemble. A similar approach is used in<sup>20</sup>, but they train the background ensemble on randomly generated feature subsets in parallel to the main ensemble from the beginning. Also, they keep the same size (number of classifiers) for the background ensemble as of the main ensemble. Different from<sup>20</sup>, this study adds a classifier to the background ensemble only if there is a drift warning on the feature level. In SAEM the background ensemble classifier is trained on a feature subset that must contain the drifted feature which is the main limitation of<sup>20</sup>. Moreover, in SAEM, the number of classifiers in  $\mathcal{B}$  is always less the than the size of  $\epsilon$  if the drift occurs in limited features. The overall working of SAEM is presented in Fig. 2.

Overall, combining the concept of random subspace and incremental ensemble pruning, SAEM generates the feature subset which must contain the features which observed the change in data distribution. The core aim of the background classifier is to learn the changes, if the subset used to train the background classifier does not have those features where actual change has occurred, it cannot achieve the core objective of its building. The overall functionality of the SAEM is provided in the algorithms 1 to 6. The symbols used in these algorithms to represent different terminologies are given be low in Table 2.

Symbol	Description	Symbol	Description
$\mathcal{S}$	Data stream	$\dot{\mathcal{S}}$ :	List: samples seen for each class
$\mathbf{K}$ :	Ensemble size	$\dot{\mathbf{R}}$ :	List: current imbalance ratio of each class
$\eta$	Feature subset size	$\ddot{\Upsilon}$ :	Learner in background ensemble
$\mathcal{W}$	Window size	$\mathcal{F}$	Feature
$\dot{\mathbf{I}}$ :	Instance	$\epsilon$ :	Ensemble
$\Psi$ :	Feature subset	$\mathcal{B}$ :	Background ensemble
$\mathcal{F}\mathbf{w}$ :	Warning level for change in feature data	$\mathbf{DIR}_C$	Dynamic imbalance ratio of class C
$\delta_w$	Performance threshold for warning	$\bar{\mathbf{w}}$ :	Class weight
$\mathbf{DIR}_C$	Cumulative imbalance ration of class C	$\mathbf{y}$ :	Class label
$\mathcal{F}\mathbf{d}$ :	Drift level for change in feature data	$\mathcal{Z}$ :	Number of classes
$\delta_d$	Performance threshold for drift	$\mathbf{P}$ :	Feature subset pool
$m$ :	Number of features	$\Upsilon'$ :	Background Learner in main ensemble
$\Upsilon$ :	Learner in main ensemble		

Table 2. Symbols descriptions.

Stream	Drift control parameters					CIR	CI	A	I
	K	K'	Speed	Seed	Width				
Stream1	0,30,60,90	100	0.001, 0.001, 0.001, 0.001	1,2,3,4	1	S	5	21	100 K
Stream2	0,30,60,90	100	0.001, 0.001, 0.001, 0.001	1,2,3,4	1	D	5	21	100 K
Stream3	3,6,9,12	21	0.01, 0.01, 0.01, 0.01	1,2,3,1	5000	S	5	21	100 K
Stream4	3,6,9,12	21	0.01, 0.01, 0.01, 0.01	1,2,3,1	5000	D	5	21	100 K
Stream5	10,20,10,20	30	0.01, 0.01, 0.01, 0.01	1,2,1,2	5000	S	5	30	100 K
Stream6	10,20,10,20	30	0.01, 0.01, 0.01, 0.01	1,2,1,2	5000	D	5	30	100 K
Stream7	15,30,15,30	60	0.01, 0.01, 0.01, 0.01	1,2,3,1	5000	S	7	60	100 K
Stream8	10,20,30,40	60	0.01, 0.011, 0.02, 0.02	1,2,3,1	1	D	5	30	100 K

Table 3. Summary of the data streams. CIR, class imbalance ratio; CI, classes; A, attributes; I, instances; S, static; D, dynamic; K, number of centroids with drift; K', total number of centroids; Speed, speed of change; Seed, sequence of stream; Width, transition period in terms of sample to completely shift from one concept to other.



**Input**

$\mathcal{S}$ : Data stream,  $K$ : Ensemble size,  $\eta$ : Feature subset size,  $W$ : Window size

**Output:**

Predictions for instances in the data stream

```

1  for  $S_i$  in  $\{S_1, \dots, S_n\}$  Do
2      if  $S_1$  then
3          Call Algorithm 2
4      else
5          // code for feature drift detection and adaptation
6          if  $S_i \geq W$  then // don't check warning for initial  $W$  instances
7              Call algorithm 3
8              if  $\hat{P}$  is not  $\emptyset$  then
9                  Call algorithm 4
10             end if
11             if  $S_i \% W == 0$  OR drift level then // drift level
12                 Call algorithm 5
13             end if
14         end if
15         Call algorithm 6
16     end if
17 end for

```

**Algorithm 1.** Smart adaptive ensemble model (SAEM).

**Input**

$S_1$ : Instance of Data stream

**Output**

Initialized arrays and lists

```

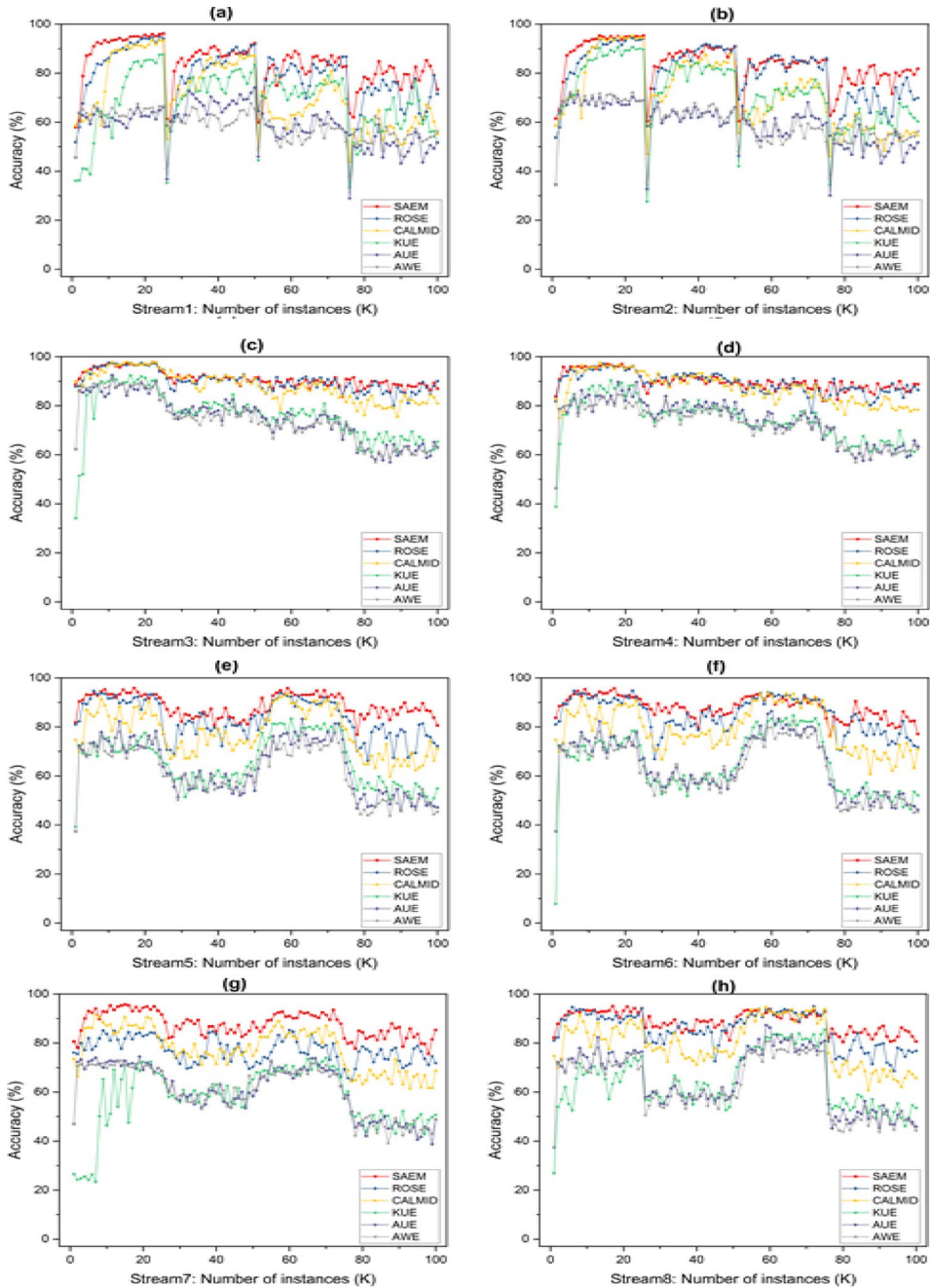
1  for  $j$  in  $\{1, \dots, k\}$  do
2       $\Psi \leftarrow$  generate  $\eta$  dimensional feature subset
3       $P_j \leftarrow$  add feature subset  $\Psi$  to pool  $P$ 
4  end for
5  for  $j$  in  $\{1, \dots, k\}$  do
6       $\Psi \leftarrow$  get the  $j$ th subset from pool  $P_j$ 
7       $\mathcal{E}_j \leftarrow$  new base learner  $\Upsilon$  on  $\Psi(S_1)$ 
8  end for
9  for  $j$  in  $\{1, \dots, \eta\}$  do
10      $DDM[j] \leftarrow$  add feature value to window  $F(S_1)$ 
11 end for

```

**Algorithm 2.** Initialization.

Model	Data stream							
	S1	S2	S3	S4	S5	S6	S7	S8
SAEM	<b>83.81</b>	<b>84.1</b>	<b>91.15</b>	<b>90.14</b>	<b>88.77</b>	<b>88.17</b>	<b>87.66</b>	<b>88.7</b>
ROSE	79.02	80.25	90.16	89.38	83.36	85.08	77.02	85.63
CALMID	71.01	72.06	88.12	87.3	77.67	79.6	77.47	79.8
KUE	68.14	71.36	75.98	74.26	65.88	64.82	57.39	64.39
AUE	58.66	59.17	74.85	74.14	63.41	64.36	60.79	64.91
AWE	58.44	60.02	74.17	72.56	61.63	62.66	60.67	62.75

**Table 4.** Accuracy. Bold values denote the highest (best) performance achieved among all methods for each respective datastream.



**Fig. 3.** Accuracies of all models on 8 data streams.

Model	Data stream							
	S1	S2	S3	S4	S5	S6	S7	S8
SAEM	<b>78.71</b>	<b>78.72</b>	88.29	<b>86.86</b>	<b>85.25</b>	<b>84.51</b>	<b>85.03</b>	<b>85.14</b>
ROSE	72.59	74.04	<b>90.16</b>	85.93	78.36	80.62	72.22	81.28
CALMID	61.84	63.08	84.27	83.16	70.8	73.32	72.67	73.49
KUE	57.58	62.06	68.01	65.71	54.93	54.32	47.58	52.78
AUE	45.04	45.46	74.85	65.64	51.9	53.83	52.53	54.41
AWE	44.57	46.32	65.19	63.45	49.27	51.4	51.87	51.32

**Table 5.** Kappa. Bold values denote the highest (best) performance achieved among all methods for each respective datastream.

---

#### Input

$S_i$ :  $i$ th instance of Data stream

#### Output

Updated feature pool, Drift status

```

1  if  $S_i \geq W$  then // don't check warning for initial  $W$  instances
2      for  $j=1$  to  $m_j$  do
3          if  $DDM[j]$  warning  $\geq F_w$  then // (equation 3)
4               $\hat{P} \leftarrow$  Generate new feature subset, add to new pool
5          end if
6          if  $DDM[j]$  warning  $\geq F_d$  then // (equation 4)
7              Set drift_status as true
8          end if
9      end for
10 end if

```

---

**Algorithm 3.** Drift detection.

---

#### Input

$S_i$ :  $i$ th instance of Data stream

#### Output

Updated Background Ensemble, instance weights

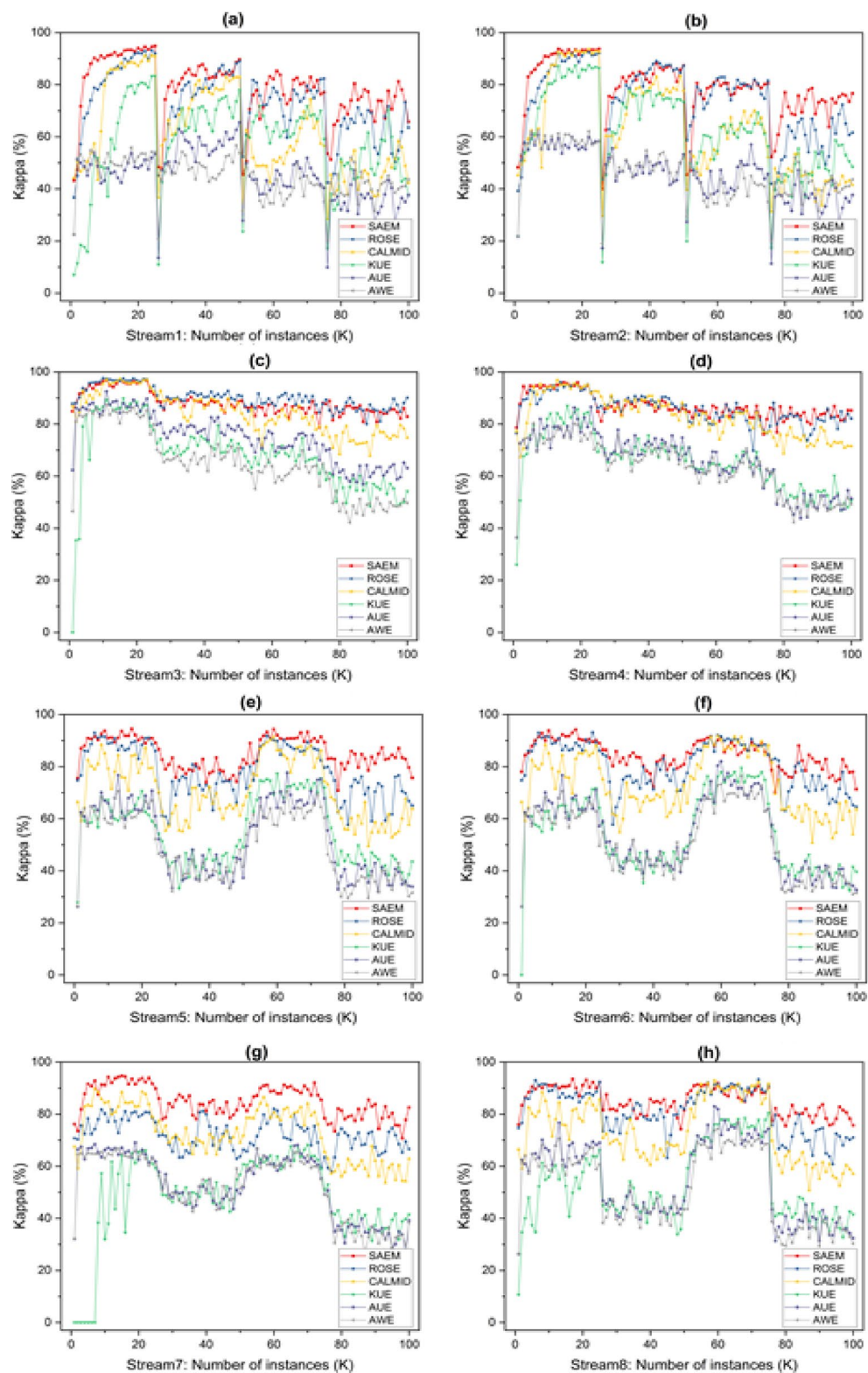
```

1  if  $\hat{P}$  is not  $\emptyset$  then
2      for  $j=1$  to  $size(\hat{P})$  do
3          if  $\beta_j == \emptyset$  then
4               $\beta_j \leftarrow$  initialize  $\tilde{Y}$  on  $\hat{P}_j(S_i)$ 
5          else
6               $\bar{w} \leftarrow$  calculate the instance weights using (6)
7               $\beta_j \leftarrow$  train  $\tilde{Y}$  on  $\hat{P}_j(S_i, \bar{w})$ 
8          end if
9      end for
10 end if

```

---

**Algorithm 4.** Train background ensemble members.



**Fig. 4.** Kappa of all models on 8 data streams.

**Input** $S_i$ :  $i$ th instance of Data stream,  $W$ **Output**

Updated Main Ensemble, Updated Feature subsets pool

```

1  if  $S_i \% W == 0$  then // after interval of every  $W$  instances
2    for  $c=1$  to  $Z$  // calculate imbalance ratio of each class
3       $\hat{R}[c] \leftarrow (S_i - \hat{S}[c]) / \hat{S}[c]$ 
4    end for
5    if drift_status == true then
6       $\Upsilon \leftarrow$  get poor performing classifier from  $\mathcal{E}$ 
7       $\tilde{\Upsilon} \leftarrow$  get poor performing classifier from  $\mathcal{B}$ 
8      if accuracy of  $\tilde{\Upsilon} >$  accuracy of  $\Upsilon$  then
9        Compare the performances of  $\mathcal{E}$  and  $\mathcal{B}$ 
10       Replace the poor classifiers of  $\mathcal{E}$  with best
          classifiers of  $\mathcal{B}$ 
11       Replace feature subsets of pool with the feature
          subsets of new pool (for classifier which shifted
          from  $\mathcal{B}$  to  $\mathcal{E}$ )
12     end if
13   end if
14 end if

```

**Algorithm 5.** Drift adaptation after interval of every  $W$  instances.**Input** $S_i$ :  $i$ th instance of Data stream,  $W$ **Output**

Predicted label, Updated Background and main Ensembles

```

1   $(x, y) \leftarrow S_i$  // read input features and label current instance
2  for  $j=1$  to  $k$  do // Prediction and incremental learning
3     $y \leftarrow$  predict ( $\mathcal{E}_j, x$ )
4     $\bar{w} \leftarrow$  calculate the instance weights using equation (8)
5    get the  $j$ th subset from pool  $P$ 
6    Train( $\mathcal{E}_j, x, \bar{w}, \mathcal{I}$ )
7    if ( $\mathcal{E}_j, x, y, \delta_w$ ) then // output-based drift warning
8      Create a background learner on  $(x, \bar{w}, \mathcal{I})$ 
9    end if
10   if ( $\mathcal{E}_j, x, y, \delta_d$ ) then // output-based drift detecting
11     Replace  $\mathcal{E}_j$  with background learner
12   end if
13 end for

```

**Algorithm 6.** Prediction and incremental learning.**Experiments**

In this section we discuss the evaluation components used to validate the proposed model in detail. We first discuss the data streams used for the experiments and then discuss the performance metrics used in the experiments.



Model	Data stream							
	S1	S2	S3	S4	S5	S6	S7	S8
SAEM	<b>82.7</b>	<b>82.8</b>	<b>89.99</b>	<b>89.01</b>	<b>88.7</b>	<b>87.98</b>	<b>88.07</b>	<b>88.39</b>
ROSE	77.69	78.56	89.14	88.34	83.14	84.94	77.77	85.42
CALMID	69.23	69.49	86.51	85.63	77.22	78.27	76.93	78.54
KUE	67.79	69.98	74.04	72.45	67.5	65.28	60.4	63.72
AUE	56.43	55.52	72.71	72.76	63.93	63.65	60.97	64.34
AWE	56.12	56.34	72.85	71.71	61.46	61.77	63.08	61.98

**Table 6.** F1 score. Bold values denote the highest (best) performance achieved among all methods for each respective datastream.

### Data streams

This study proposes an ensemble based adaptive model to address the joint issue of concept drift and class imbalance as well as the dynamic class imbalance issue. Therefore, to test the proposed model, we generated synthetic multi-class data streams containing different drift types. Each data stream represents a different case. Therefore, to know where exactly the drift is occurring and what is the actual number of samples in each class i.e., the class imbalance ratio; we simulated different data streams in a way that each stream has two versions, the data stream with static imbalance ratio and the data stream with dynamic imbalance ratio. Because we are uninformed of the drift location and drift type in real data streams, evaluating the performance of proposed model on such data streams is difficult<sup>94</sup>. As a result, for our investigations, we use synthetically produced data streams in which we know the drift locations, drift type, and class imbalance ratio at the time of drift occurrence.

The massive online analysis (MOA)<sup>95</sup> framework was used to generate the data streams with different drift types. MOA provides different generators for generating data streams. In this study, the RandomRBFGeneratorDrift was used to generate different multi-class data streams. The RandomRBFGeneratorDrift is a stream generator that generates a stream of instances based on Radial Basis Function (RBF). This generator helps in generating concept drift based on the change in the data distribution over time and has been used by<sup>26,27,96</sup>.

Every generated stream contains 100K samples, after every 25K samples, a new concept replaces the old one. The ‘sudden’ and ‘gradual’ drifts are easy to generate using the width (w) property. For sudden, we set  $w = 1$ , and for gradual  $w = 5$  K, which means during these 5 K instances, the concept will change gradually. The position (p) is set as 25 K in all cases, which means it is the center of each drift (i.e., concept drift occurs after every 25 K). Table 1 shows the summary of the generated streams. The data stream is generated considering the following definition given in<sup>90</sup>:

“Given data streams  $X$ , every instance  $X_t$  is generated by a data source or a concept  $S_t$ . If all the data is sampled from the same source, i.e.  $S_1 = S_2 = \dots = S_t = S$ , we say that the concept is stable. If for any two time points  $i$  and  $j$   $S_i \neq S_j$ , we say that there is a concept drift.”

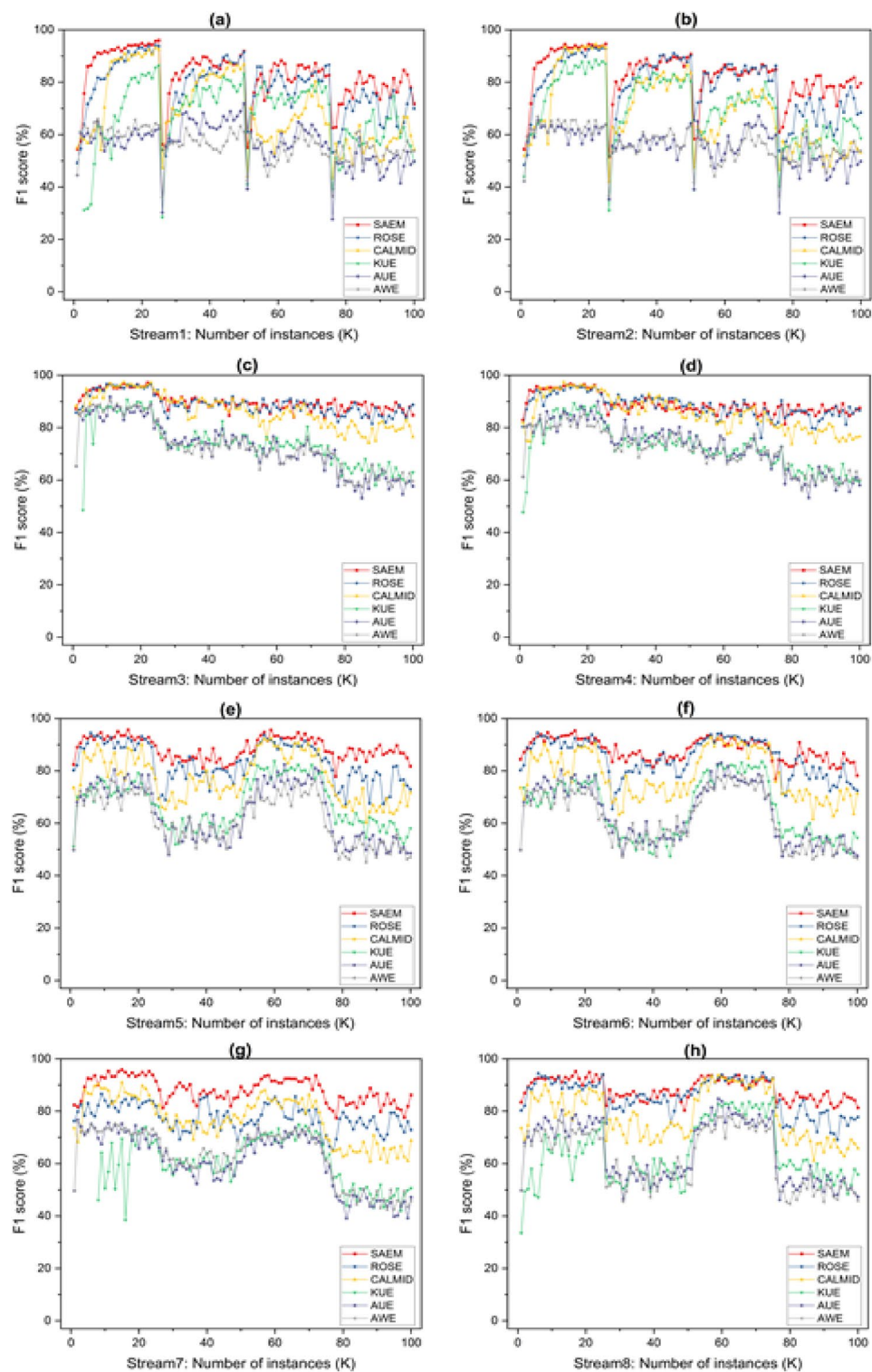
The RandomRBFGeneratorDrift class is used to generate different data streams by controlling the following parameters. The overall summary of the generated data streams is given in the Table 3. Whereas, The performance of the proposed model is measured on five performance measure metrics which are: the Accuracy, Precision, Recall, F1 score and Kappa metrics, mostly used in the literature when data is imbalanced<sup>17,97</sup>. The summary of each data stream is given below:

- *Stream1*: Four incremental concepts with varying drift centroid numbers and sudden drifts every 25 K instances. Class imbalance ratio remains constant.
- *Stream2*: Like Stream1 but with dynamic class imbalance ratios, where class proportions change with concept shifts.
- *Stream3*: Four incremental concepts with varying drift centroid numbers. Instead of sudden drifts, gradual shifts occur over 5000 instances.
- *Stream4*: Same as Stream3 but with dynamic class imbalance ratios, combining concept drift with changing class proportions.
- *Stream5*: Focuses on reoccurring drifts with consistent centroid values for alternating concepts, combined with gradual drift. Class imbalance is constant, and dimensionality is increased to 30 features.
- *Stream6*: A copy of Stream5 with dynamic class imbalance, examining concept drift with changing class proportions.
- *Stream7*: A complex data stream with 60 features and 7 classes, featuring gradually reoccurring drifts, increasing drift severity, and a constant class imbalance ratio.
- *Stream8*: Similar to Stream7 but with sudden shifts instead of gradual ones and an increasing number of centroids over time, introducing gradual drift. Speed hyperparameter also increases.

### Bench marking and parameter setting

The performance of the proposed model SAEM is compared with existing state-of-the-art ensemble approaches. These approaches are:

1. Accuracy Weighted Ensemble (AWE)<sup>83</sup>.
2. Accuracy Update Ensemble (AUE)<sup>84</sup>.
3. Kappa Updated Ensemble (KUE)<sup>85</sup>.



**Fig. 5.** F1 scores of all models on 8 data streams.

Model	Data stream							
	S1	S2	S3	S4	S5	S6	S7	S8
SAEM	<b>84.06</b>	<b>86.13</b>	<b>91.72</b>	<b>91.54</b>	<b>90.66</b>	<b>90.21</b>	<b>89.91</b>	<b>90.66</b>
ROSE	78.2	79.63	90.75	89.84	84.16	86.06	78.86	86.69
CALMID	70.88	71.42	88.19	86.82	78.77	79.39	78.89	79.81
KUE	71.04	73.33	76.82	76.62	72.48	68.71	64.9	69.05
AUE	59.28	58.62	76.42	75.33	66.75	65.41	62.74	66.36
AWE	59.41	60.38	79.08	75.27	65.37	64.49	69.94	65.13

**Table 7.** Precision. Bold values denote the highest (best) performance achieved among all methods for each respective datastream.

4. Comprehensive Active Learning method for Multiclass Imbalanced streaming Data with concept drift (CALMID)<sup>86</sup>.
5. Robust Online Self-Adjusting Ensemble (ROSE)<sup>20</sup>.

Fair comparison requires all the models to use the same parameters to generate the results. Therefore, this study uses the same values for the common parameters such as window size for all the approaches. SAEM and ROSE use the feature subset to train the classifier, therefore, for both approaches, the value 0.7 (70% of the total feature space) is used for the feature subset size. Another common parameter was the ensemble size which represents the number of classifiers in the ensemble. So, for ensemble size, value 10 was set for all the models. The default values were used for any additional parameters. The performance of our proposed model SAEM is compared with AUE, AWE, KUE, CALMID, and ROSE on eight data streams. Each data stream has four concepts of size 25 K instances.

## Results and discussions

The results are discussed below in a separate sub-section for each data stream. As discussed earlier, each data stream is the representation of a separate case.

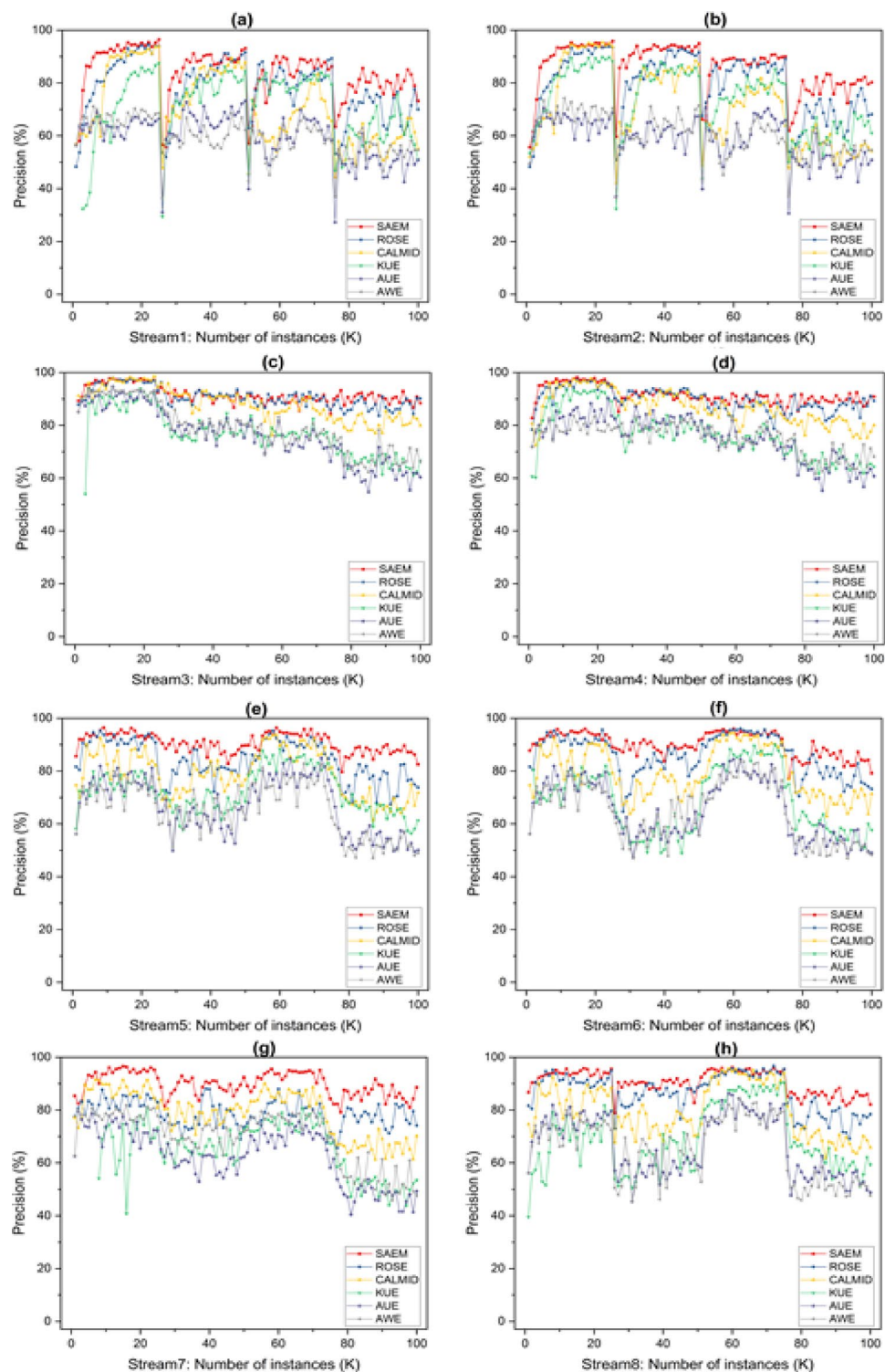
### Accuracy

Table 4 shows the average accuracy of all six models used in the comparison. The analysis of accuracy-based performance differences between SAEM and the other models (ROSE, CALMID, KUE, AUE, and AWE) across all eight data streams reveals a consistent pattern of SAEM's superior performance. SAEM surpasses ROSE, CALMID, KUE, AUE, and AWE on all data streams, showcasing an average improvement of 4.1%, 8.7%, 20.1%, 22.8%, and 23.7%, respectively. These results emphasize the effectiveness of SAEM in maintaining high accuracy levels, especially during concept drift scenarios, and highlight its robustness across a variety of data streams. SAEM's feature-level monitoring and smart background ensemble training distinguish it as a promising approach for addressing concept drift and maintaining model stability in online machine learning applications.

The results are also presented in Fig. 3a–h for each data stream. The x-axis represents the number of instances processed in thousands (K), and y-axis represents the accuracy produced in percentage. SAEM produced high accuracy compared to the other models for all concepts. SAEM produced high accuracy compared to the other models for all concepts. From instance, on stream1 from 1 to 25 K, the accuracies are much more stable, meaning there is less fluctuation in the accuracy over the period, compared to other concepts especially the 3rd (50 K to 75 K) and 4th concept (75 K to 100 K). The reason is, in the first concept the number of centroids with drift was zero, so there was no incremental drift, so the performance for the first concept is better and more stable. The number of centroids with drift for the 23rd and 4th concept in stream1 is 30,60,90, respectively. This shows that the more severity of drift the lesser the accuracy of the model. On stream2 (Fig. 3b) which has dynamic class imbalance ratio along with concept drift, results show that SAEM remained the best among all compared models.

Like stream1, stream3 also has four incremental concepts with a constant speed of 0.01 (faster than stream1) but each concept has a different number of drift centroids which are 3,6,9 and 12, whereas total centroids are 21. After every 25 K instances, instead of sudden drift, stream3 introduces a gradual shift (drift) from one concept to the other. Due to gradual drift, one concept is shifted to the other over a span of 5,000 instances. So, there are four incremental concepts and three gradual drifts. Each concept is generated using a different seed from 1 to 4, to generate a different sequence of stream. The class imbalance ratio in stream3 is constant over time. Different from stream1 and stream2 where the shift among concepts was sudden, which is comparatively easy to identify, stream3 is more complex. This shows that it is difficult to cope with the combination of gradual drift and incremental drifts in imbalanced data streams.

The results on stream3 (Fig. 3c) show that the SAEM and ROSE produced approximately 97% accuracy before the arrival of the first drift. After the arrival of the first drift, an approximately 10 to 15% decrease can be seen in accuracy. Later, SAEM and ROSE managed to maintain the performance to a certain level that is approximately 90%. On the other hand, the performance of CALMID, KAU, AUE and AWE kept falling from the level of 90% to the level of 55%. This shows around 35% decrease in performance which is around 12%



**Fig. 6.** Precision of all models on 8 data streams.

Model	Data stream							
	S1	S2	S3	S4	S5	S6	S7	S8
SAEM	<b>81.41</b>	<b>79.87</b>	<b>88.33</b>	86.64	<b>86.89</b>	<b>85.9</b>	<b>86.32</b>	<b>86.27</b>
ROSE	77.2	77.56	87.61	<b>86.91</b>	82.18	83.87	76.73	84.22
CALMID	67.7	67.72	84.92	84.5	75.8	77.24	75.1	77.35
KUE	64.12	67.03	70.7	68.79	63.32	61.74	53.7	59.29
AUE	53.94	52.81	76.42	70.52	61.45	62.05	59.38	62.52
AWE	53.29	52.93	67.62	68.79	58.19	59.43	57.35	59.05

**Table 8.** Recall. Bold values denote the highest (best) performance achieved among all methods for each respective datastream.

further decrease after arrival of every new drift. The models are unable to maintain the stability in performance as produced by SAEM and ROSE.

The stream5 is focused on the concept of reoccurring drifts. Instead of using different values for drifted centroid parameters, stream is generated keeping the same value for the alternative concepts. Such as (10,20,10,20), first and third concepts have same values for the parameter which is 10, and second and fourth concept has the same values which is 20. Similarly, the seed value is also kept the same for alternative concepts such as (1,2,1,2). The results (Fig. 3e) show that the SAEM again produced better results compared to all the other models. Even though stream5 has high dimensional data (30 features) compared to the previously discussed data streams having 21 features. The SAEM showed stability in reoccurring concepts, but all the other models produced poor results in case of reoccurring drift especially for the 4th concept i.e., from 75 K to 100 K instances.

For stream, the results are more likely similar to the results achieved on stream5. However, the change in class imbalance ratio has further affected the performance of all models. The second concept and the fourth concept are more severe, their effect is clearly visible in the performances of all the models. The results show that the SAEM again produced better results compared to all the other models. The stream7 is generated with 60 features and 7 classes to test the proposed model on more complex data stream. Stream7 is based on the gradually reoccurring drifts which means one concept will shift to other gradually over the span of 5000 instances. For stream7 the severity of the drift was also increased by increasing the value of drifted centroids. The SAEM again outperformed the existing state-of-the-art models on a data stream which is more complex (Fig. 3g).

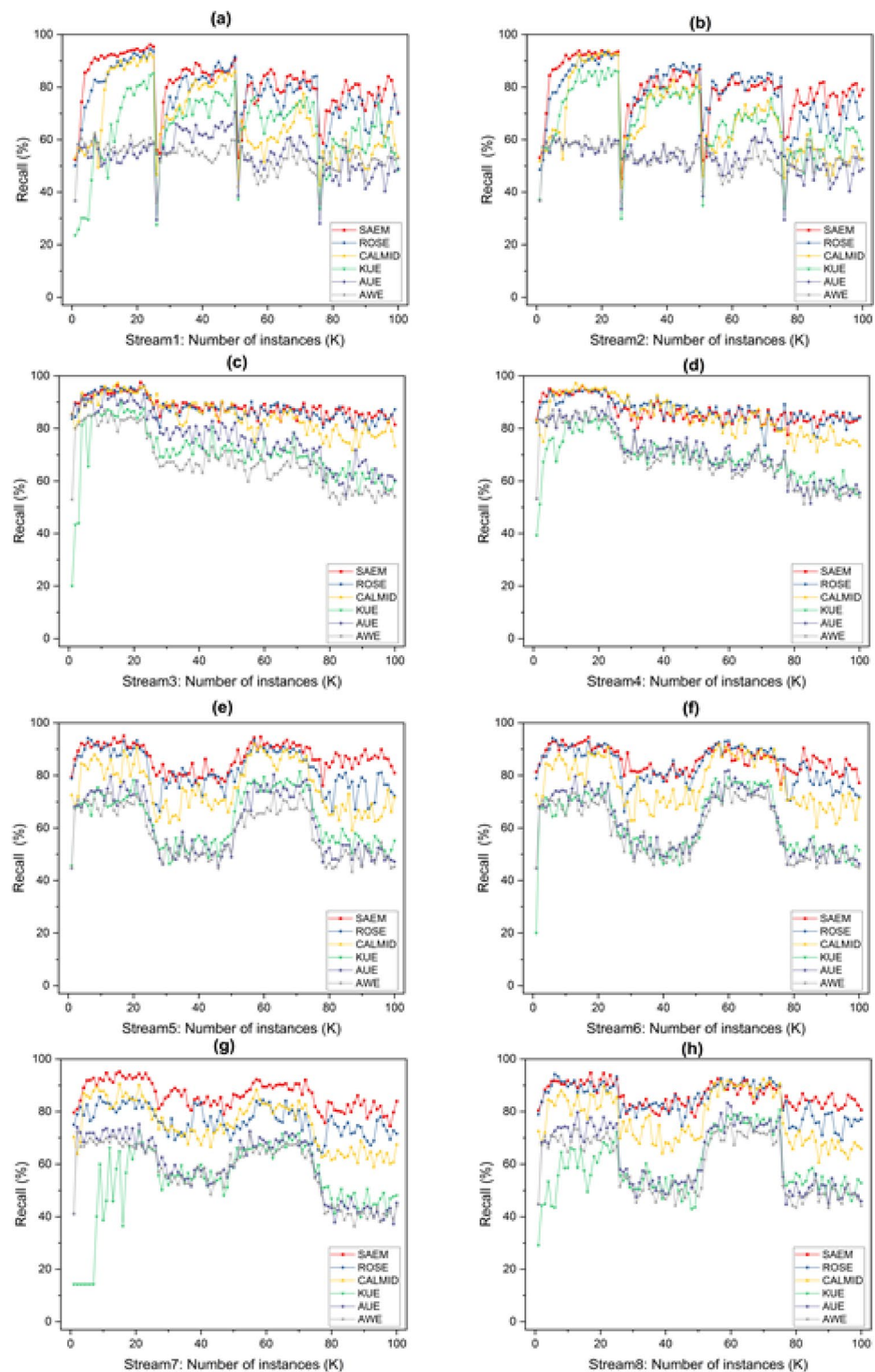
SAEM showed far better performance than CALMID, KUE, AUE, and AWE on stream7 as it did on all previous data streams. However, the CALMID produced almost equivalent results to the SAEM for one concept (50 K to 75 K instances), for the rest concepts SAEM outperformed CALMID. Comparatively, only ROSE showed performance closer to the SAEM so far. But for stream7 SAEM produced far better results than ROSE. The reason is, SAEM trains new classifiers (for background ensemble) on features that are affected due to drift, no matter how complex the data stream is. Whereas the ROSE randomly selects features to train classifiers for the background ensemble and in the case of high feature space data stream there is very much possibility that the features that are affected because of the drift are not selected for the training of new classifier during the concept drift adaptation phase. Hence, the approach proposed in SAEM is to monitor the drift on the feature level and train background ensemble classifiers on the data set which must contain the features where the data distribution has changed.

The stream8 is generated with 30 features and 5 classes. Here, instead of gradually shifting from one concept to another, a sudden shift is applied. Moreover, instead of reoccurring drift, the value of number of centroids in every concept is increased over time to introduce a gradual drift. Also, the value of speed hyperparameter has increased over the period. For stream8, the number of centroids is increased over time for each concept from first to fourth i.e., 10, 20, 30, 40, respectively. SAEM once again produced convincing results on data stream8. The KUE, AUE, and AWE produced much better results than they produced on stream7 for at least the first 3 concepts. The same is the case with ROSE and CALMID, both produced better results for the initial concepts, but when the drift severity was high especially at the second and fourth concepts (instances from 25 K to 50 K and from 75 K to 100 K), the performances of ROSE and CALMID are severely affected. Whereas SAEM produced better results than the other models on all concepts, especially for the last concept, the difference in performance of SAEM and other models like ROSE is more visible.

Kappa

Table 5 shows the average kappa values of all six models used in the comparison. The analysis of performance based on the kappa metric underscores the consistent superiority of SAEM compared to the other five models (ROSE, CALMID, KUE, AUE, and AWE) across all eight data streams. SAEM outperforms ROSE, CALMID, KUE, AUE, and AWE on all streams, with average improvements of 4.7%, 11.23%, 26.19%, 28.6%, and 31.14%, respectively. These results affirm SAEM’s robust adaptability to concept drift, its capacity to maintain stable model performance, and its ability to consistently achieve high kappa scores. The substantial improvements over





**Fig. 7.** Recall of all models on 8 data streams.

Method	Accuracy	Precision	Recall	F1 score	Kappa
SAEM (Proposed)	<b>1.00</b>	<b>1.00</b>	<b>1.13</b>	<b>1.00</b>	<b>1.13</b>
ROSE	2.13	2.13	1.88	2.00	2.00
CALMID	2.88	3.13	3.00	3.00	2.88
KUE	4.38	4.00	4.81	4.38	4.50
AUE	4.88	5.25	4.50	5.00	4.75
AWE	5.75	5.50	5.69	5.63	5.75
Friedman test results:					
X <sup>2</sup> (Accuracy): 37.14, <i>p</i> -value: 5.16 × 10 <sup>−7</sup> (significant at α = 0.05, Reject H <sub>0</sub> )					
X <sup>2</sup> (Precision): 35.64, <i>p</i> -value: 1.12 × 10 <sup>−6</sup> (significant at α = 0.05, Reject H <sub>0</sub> )					
X <sup>2</sup> (Recall): 36.79, <i>p</i> -value: 6.59 × 10 <sup>−7</sup> (significant at α = 0.05, Reject H <sub>0</sub> )					
X <sup>2</sup> (F1 score): 35.79, <i>p</i> -value: 1.05 × 10 <sup>−6</sup> (significant at α = 0.05, Reject H <sub>0</sub> )					
X <sup>2</sup> (Kappa): 36.36, <i>p</i> -value: 8.06 × 10 <sup>−7</sup> (significant at α = 0.05, Reject H <sub>0</sub> )					

**Table 9.** Mean ranking and friedman test results on accuracy, precision, recall, F1 score, and Kappa. Bold values denote the highest (best) performance achieved among all methods for each respective datastream.

other models reflect the effectiveness of SAEM’s feature-level monitoring and intelligent background ensemble training in navigating the challenges of evolving data streams. This emphasizes SAEM’s potential as a valuable solution for online machine learning applications where concept drift poses a significant challenge. The results are also presented in Fig. 4. The x-axis represents the number of instances processed in thousands (K), and y-axis represents the kappa produced in percentage. SAEM produced high accuracy compared to the other models for all concepts.

F1 score

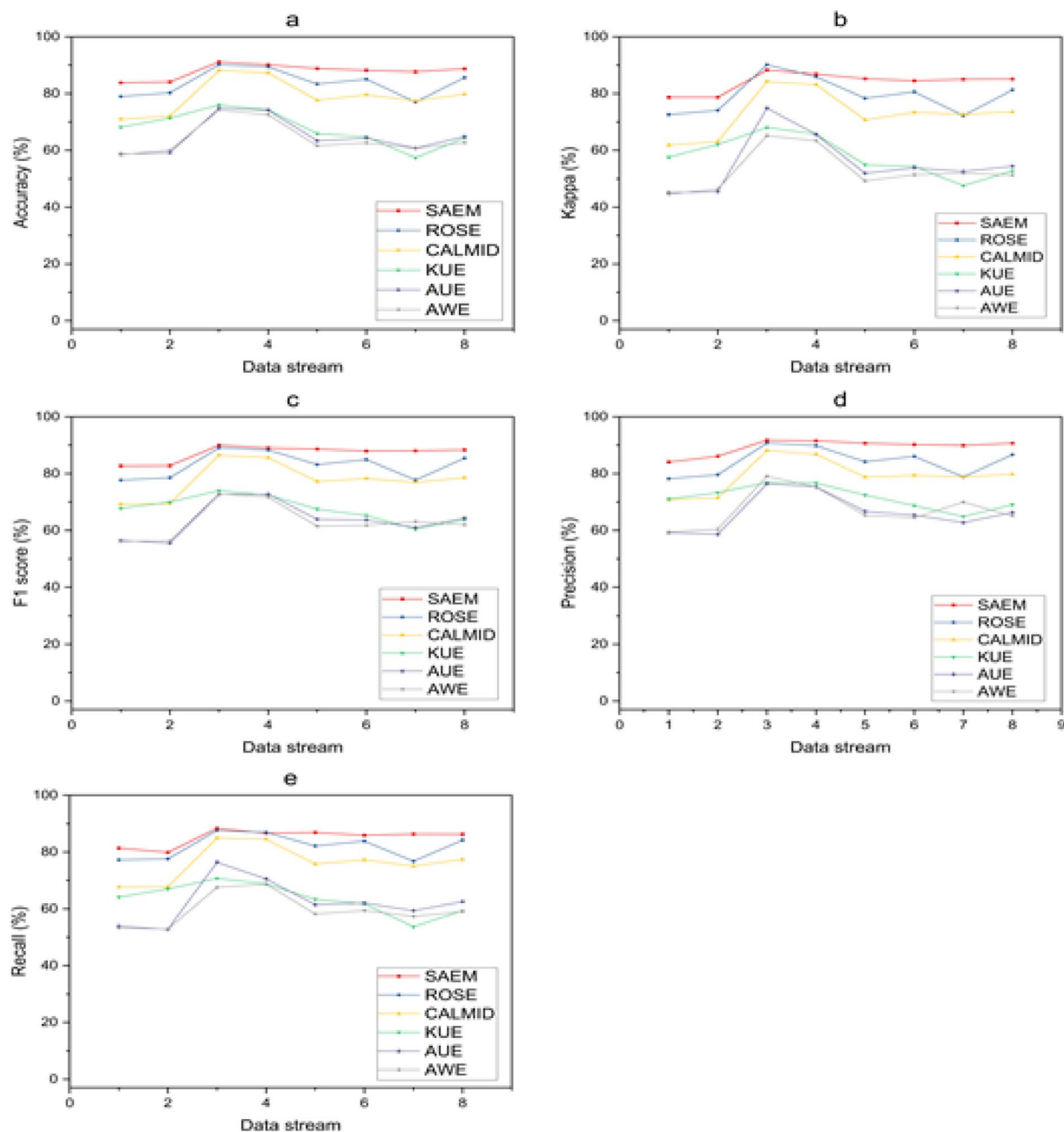
The evaluation of performance based on the F1 score metric reveals a consistent and substantial advantage of SAEM over the other models (ROSE, CALMID, KUE, AUE, and AWE) across all eight data streams, Table 6. SAEM consistently outperforms ROSE, CALMID, KUE, AUE, and AWE on every stream, showcasing average improvements of 4.1%, 11.23%, 19.56%, 23.42%, and 24.05%, respectively. This underscores the remarkable adaptability and stability offered by SAEM in the face of concept drift. SAEM’s ability to consistently achieve higher F1 scores reflects its competence in striking a balance between precision and recall, particularly in dynamic and evolving data stream environments. The noteworthy performance enhancements over alternative models underscore SAEM’s pivotal role as an innovative solution for the intricate challenges posed by concept drift in the realm of online machine learning. The results are also presented in Fig. 5. The x-axis represents the number of instances processed in thousands (K), and y-axis represents the F1 score produced in percentage. SAEM produced high accuracy compared to the other models for all concepts.

Precision

The assessment of performance in terms of precision metric illuminates a compelling narrative of SAEM’s consistent and substantial superiority when compared to the other models (ROSE, CALMID, KUE, AUE, and AWE) across all eight data streams, Table 7. SAEM consistently outperforms ROSE, CALMID, KUE, AUE, and AWE on each stream, with notable average improvements of 5.1%, 10.1%, 17.74%, 23%, and 21.98%, respectively. This underscores SAEM’s exceptional precision in correctly identifying true positives while minimizing false positives, a crucial aspect in concept drift scenarios. The significant performance gains exhibited by SAEM across various data streams highlight its competence in delivering precise and accurate predictions in dynamic and evolving machine learning environments, affirming its status as an innovative and impactful solution for addressing the multifaceted challenges of concept drift. The results are also presented in Fig. 6. The x-axis represents the number of instances processed in thousands (K), and y-axis represents the precision produced in percentage. SAEM produced high accuracy compared to the other models for all concepts.

Recall

The assessment of performance in terms of precision metric illuminates a compelling narrative of SAEM’s consistent and substantial superiority when compared to the other models (ROSE, CALMID, KUE, AUE, and AWE) across all eight data streams, Table 8. SAEM consistently outperforms ROSE, CALMID, KUE, AUE, and AWE on each stream, with notable average improvements of 5.1%, 10.1%, 17.74%, 23%, and 21.98%, respectively. This underscores SAEM’s exceptional precision in correctly identifying true positives while minimizing false positives, a crucial aspect in concept drift scenarios. The significant performance gains exhibited by SAEM across various data streams highlight its competence in delivering precise and accurate predictions in dynamic and evolving machine learning environments, affirming its status as an innovative and impactful solution for addressing the multifaceted challenges of concept drift. The results are also presented in Fig. 7. The x-axis represents the number of instances processed in thousands (K), and y-axis represents the recall produced in percentage. SAEM produced high accuracy compared to the other models for all concepts.



**Fig. 8.** The performance comparison on each metric (a–e) and summary (f) on Stream8.

### Statistical comparison of models using friedman test

To evaluate and compare the performance of all models across multiple evaluation metrics such as Accuracy, Precision, Recall, F1 score, and Kappa, we employed the non-parametric Friedman test. This test ranks each model per dataset for a given metric, where a lower mean rank indicates superior performance. Table 9 presents the average ranks of each model for all five metrics. In contrast, models such as AWE and AUE received the highest mean ranks, reflecting comparatively lower performance.

The proposed SAEM method consistently achieved the lowest mean ranks across all metrics, indicating the most robust and consistent performance. Specifically, SAEM scored the top rank (i.e., rank = 1.00 or close) in all metrics. In contrast, models such as AWE and AUE received the highest mean ranks, reflecting comparatively lower performance. The Friedman test results yielded statistically significant differences among the models across all metrics.

All p-values are significantly below the threshold  $\alpha = 0.05$ , leading to the rejection of the null hypothesis ( $H_0$ ), which assumes that all models perform equally. These findings confirm that there are statistically significant differences in performance among the evaluated models, with SAEM demonstrating clear superiority.

## Findings

Based on the validation experiments conducted eight different data streams, and the performance analysis on the five metrics, SAEM outperforms all the other models (ROSE, CALMID, KUE, AUE, and AWE) on all eight data streams. SAEM achieved higher accuracy, Kappa, F1-score, precision, and recall values compared to the other models. Specifically, SAEM outperformed ROSE on all data streams with an average improvement of 4.1% in accuracy, 4.7% in Kappa, 4.1% in F1-score, 5.1% in precision, and 3.2% in recall. SAEM significantly outperformed CALMID on all data streams with an average improvement of 8.68% in accuracy, 11.23% in Kappa, 9.5% in F1-score, 10.1% in precision, and 8.91% in recall. SAEM significantly outperformed KUE on all data streams with an average improvement of 20.03% in accuracy, 26.19% in Kappa, 19.56% in F1-score, 17.74% in precision, and 21.6% in recall. SAEM significantly outperformed AUE on all data streams with an average improvement of 22.77% in accuracy, 28.6% in Kappa, 23.42% in F1-score, 23% in precision, and 22.8% in recall. Finally, SAEM significantly outperformed AWE on all data streams with an average improvement of 23.7% in accuracy, 31.14% in Kappa, 24.1% in F1-score, 21.98% in precision, and 25.62% in recall. Therefore, we can conclude that SAEM is the best performing approach among the five models evaluated in this study due to its Smart Proactive Adaptive Ensemble approach which make it capable of timely concept adaptation and avoid performance degradation. Moreover, the overall performance average of all the models over each data stream is presented in Fig. 8(a-e) for all performance metrics. The Friedman test confirmed statistically significant performance differences among all models across five key metrics. The proposed SAEM method consistently achieved the best mean ranks, demonstrating superior and robust performance.

## Conclusion

In this research, we proposed SAEM (Smart Adaptive Ensemble Model), a novel framework designed to effectively handle two persistent challenges in online machine learning: concept drift and multi-class imbalance in non-stationary data streams. Unlike traditional ensemble methods that often rely on randomly selected feature subsets, SAEM introduces a feature-level drift detection mechanism to identify features undergoing distributional changes. These drifted features are then explicitly used to guide the training of new classifiers, which are integrated into a background ensemble, enabling more targeted and adaptive concept evolution.

To address class imbalance, SAEM departs from standard techniques such as re-sampling or misclassification weighting. Instead, it employs a dual imbalance monitoring strategy, leveraging both the dynamic imbalance ratio (within the current data window) and the cumulative imbalance ratio (over the stream's history) to assign class-wise instance weights. This allows SAEM to respond effectively to evolving class distributions, maintaining both short-term responsiveness and long-term balance awareness.

We evaluated SAEM across multiple data streams and benchmarked its performance against several state-of-the-art ensemble approaches. The results demonstrate consistent and substantial improvements across various performance metrics, including average gains of 15.86% in accuracy, 20.35% in Kappa, 16.12% in F1-score, 15.58% in precision, and 16.42% in recall, underscoring both the effectiveness and practicality of the proposed model. Moreover, the Friedman test confirmed statistically significant performance differences among all models across five key metrics.

While SAEM excels in adapting to meaningful concept changes, its current approach may be sensitive to drifts occurring in less relevant features. As future work, we plan to incorporate feature importance estimation to further enhance SAEM's robustness and ensure its adaptability is focused on semantically significant changes, thereby improving both efficiency and interpretability in real-time streaming applications.

## Data availability

The datasets used and/or analyzed during the current study are available from the corresponding author upon request.

Received: 2 February 2025; Accepted: 30 May 2025

Published online: 01 July 2025

## References

- Wu, X., Zhu, X., Wu, G. Q. & Ding, W. Data mining with big data. *IEEE Trans. Knowl. Data Eng.* **26**(1), 97–107 (2013).
- Li, X., Ding, Q. & Sun, J. Q. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliab. Eng. Syst. Saf.* **172**, 1–11 (2018).
- Jayasinghe, L., Samarasinghe, T., Yuen, C., Low, J. C. N. & Ge, S. S. Temporal convolutional memory networks for remaining useful life estimation of industrial machinery. *arXiv preprint arXiv:1810.05644* (2018).
- Fauzi, M. F. A. M., Aziz, I. A. & Amiruddin, A. The Prediction of Remaining Useful Life (RUL) in Oil and Gas Industry using Artificial Neural Network (ANN) Algorithm. in *2019 IEEE Conference on Big Data and Analytics (ICBDA)* 7–11 (IEEE, 2019).
- o., D. J. C. S. & Metz, T. P. Tackling urban traffic congestion: The experience of London, Stockholm and Singapore. *Case Stud. Transp. Policy* **6**(4), 494–498 (2018).
- Zhang, G., Yang, D., Galanis, G., Androulakis, E. J. R. & Reviews, S. E. Solar forecasting with hourly updated numerical weather prediction. *Renew. Sustain. Energy Rev.* **154**, 111768 (2022).
- Wang, T., Zhao, S. & Zhu, G. and H. J. A. E. Zheng, A machine learning-based E.rly warning system for systemic banking crises, **53**, 26, pp. 2974–2992, (2021).
- Zhong, X. & Enke, D. J. F. I. Predicting the daily return direction of the stock market using hybrid machine learning algorithms. *Financ. Innov.* **5**(1), 1–20 (2019).
- Pongpaichet, S., Jankapor, S., Janchai, S. & Tongsanit, T. Early detection at-risk students using machine learning. in *2020 International Conference on Information and Communication Technology Convergence (ICTC)* 283–287 (IEEE, 2020).
- Amin, A., Al-Obeidat, F., Shah, B., Adnan, A. & Loo, J. Customer churn prediction in telecommunication industry using data certainty *J. Bus. Res.* **94**, 290–301 (2019).



11. Khan, Z. F., & Alotaibi, S. R. Applications of artificial intelligence and big data analytics in m-health: A healthcare system perspective. *J. Healthc. Eng.* **2020** (2020).
12. Abbasi, M., Shahraki, A. & Taherkordi, A. J. C. C. Deep learning for network traffic monitoring and analysis (NTMA): A survey. *Computer Commun.* **170** 19–41 (2021).
13. Khandekar, V. S. & Srinath, P. Non-stationary data stream analysis: State-of-the-art challenges and solutions. in *Proceeding of International Conference on Computational Science and Applications* 67–80 (Springer, 2020).
14. Ghazikhani, A., Monsefi, R. & Yazdi, H. S. Ensemble of online neural networks for non-stationary and imbalanced data streams. *Neurocomputing* **122**, 535–544 (2013).
15. Palli, A. S. et al. Online machine learning from Non-stationary data streams in the presence of concept drift and class imbalance: A systematic review. *J. Inform. Communication Technol.* **23**(1), 105–139 (2024).
16. Mahapatra, S. & Chandola, V. Learning manifolds from non-stationary streams. *J. Big Data.* **11**(1), 42 (2024).
17. Wang, S., Minku, L. L. & Yao, X. A systematic study of online class imbalance learning with concept drift. *IEEE Trans. Neural Networks Learn. Syst.* **29**(10), 4802–4821 (2018).
18. Gomes, H. M., Barddal, J. P., Enembreck, F. & Bifet, A. A survey on ensemble learning for data stream classification. *ACM Comput. Surv. (CSUR)* **50**(2), 1–36 (2017).
19. Gomes, H. M., Read, J. & Bifet, A. Streaming random patches for evolving data stream classification. in *IEEE International Conference on Data Mining (ICDM)* 240–249 (IEEE, 2019).
20. Cano, A. & Krawczyk, B. ROSE: Robust online self-adjusting ensemble for continual learning on imbalanced drifting data streams. *Machine Learn.* **111**(7), 2561–2599. <https://doi.org/10.1007/s10994-022-06168-x> (2022).
21. Soomro, A. A. et al. Insights into modern machine learning approaches for bearing fault classification: A systematic literature review. *Res. Eng.* 102700 (2024).
22. Soomro, A. A. et al. Data augmentation using SMOTE technique: Application for prediction of burst pressure of hydrocarbons pipeline using supervised machine learning models. *Res. Eng.* **24**, 103233 (2024).
23. Aghahari, S. & Singh, A. K. Concept drift detection in data stream mining: A literature review. *J. King Saud Univ. - Comput. Inform. Sci.* <https://doi.org/10.1016/j.jksuci.2021.11.006> (2021).
24. Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M. & Bouchachia, A. A survey on concept drift adaptation, (in English), *ACM Comput. Surv.* **46**(4), 44. <https://doi.org/10.1145/2523813> (2014).
25. Lu, J. et al. Learning under concept drift: A review. *IEEE Trans. Knowl. Data Eng.* **31**(12), 2346–2363 (2018).
26. Losing, V., Hammer, B. & Wersing, H. KNN classifier with self adjusting memory for heterogeneous concept drift. in *IEEE 16th international conference on data mining (ICDM)* 291–300 (IEEE, 2016).
27. Lu, J. et al. Learning under concept drift: A review. *IEEE Trans. Knowl. Data Eng.* **31**(12), 2346–2363. <https://doi.org/10.1109/TKDE.2018.2876857> (2019).
28. Palli, A. S., Jaafar, J., Gomes, H. M., Hashmani, M. A. & Gilal, A. R. An experimental analysis of drift detection methods on multi-class imbalanced data streams. *Appl. Sci.* **12**(22), 11688. <https://doi.org/10.3390/app122211688> (2022).
29. Chen, J. et al. Multi-type concept drift detection under a dual-layer variable sliding window in frequent pattern mining with cloud computing. *J. Cloud Comput.* **13**(1), 40 (2024).
30. Gama, J., Medas, P., Castillo, G. & Rodrigues, P. Learning with drift detection, in *Advances in Artificial Intelligence—SBIA 2004: 17th Brazilian Symposium on Artificial Intelligence, Sao Luis, Maranhao, Brazil, September 29–October 1, 2004. Proceedings* 17 286–295 (Springer, 2004).
31. Baena-Garcia, M. et al. Early drift detection method, in *Fourth international workshop on knowledge discovery from data streams* vol. 6, 77–86 (2006).
32. Barros, R. S. M., Cabral, D. R. L., Gonçalves, P. M. & Santos, S. G. T. C. RDDM: Reactive drift detection method, *Expert. Syst. Appl.* **90**, 344–355. <https://doi.org/10.1016/j.eswa.2017.08.023> (2017).
33. Dasu, T., Krishnan, S., Venkatasubramanian, S. & Yi, K. An information-theoretic approach to detecting changes in multi-dimensional data streams, in *In Proc. Symp. on the Interface of Statistics, Computing Science, and Applications* Citeseer (2006).
34. Roberts, S. W. Control chart tests based on geometric moving averages. *Technometrics* **42**(1), 97–101 (2000).
35. Nishida, K. & Yamauchi, K. Detecting concept drift using statistical testing. In *International conference on discovery science* 264–269 (Springer, 2007).
36. Hoens, T. R., Polikar, R. & Chawla, N. V. Learning from streaming data with concept drift and imbalance: An overview. *Prog. Artif. Intell. Rev.* **1**(1), 89–101 (2012).
37. Spelman, V. S. & Porkodi, R. A review on handling imbalanced data. in *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)* 1–11 (IEEE, 2018).
38. Branco, P., Torgo, L. & Ribeiro, R. P. A survey of predictive modeling on imbalanced domains. *ACM Comput. Surv. (CSUR)* **49**(2), 1–50 (2016).
39. Haixiang, G. et al. Learning from class-imbalanced data: Review of methods and applications. *Expert Sys Appl* **73**, 220–239 (2017).
40. Zhu, B. & Baesens, B. An empirical comparison of techniques for the class imbalance problem in churn prediction. *Inf. Sci.* **408**, 84–99 (2017).
41. Korycki, L. & Krawczyk, B. Online Oversampling for Sparsely Labeled Imbalanced and Non-Stationary Data Streams. in *International Joint Conference on Neural Networks (IJCNN) held as part of the IEEE World Congress on Computational Intelligence (IEEE WCCI)*, Electr. Network, Jul 19–24 2020, in *IEEE International Joint Conference on Neural Networks (IJCNN)*, (2020).
42. Tsai, C.-F., Lin, W.-C., Hu, Y.-H. & Yao, G.-T. Under-sampling class imbalanced datasets by combining clustering analysis and instance selection. *Inf. Sci.* **477**, 47–54 (2019).
43. SukarnaBarua, Md., Islam, M., Yao, X. & Murase, K. MWMOTE--majority weighted minority oversampling technique for imbalanced data set learning. *IEEE Trans. Knowl. Data Eng.* **26**(2), 405–425. <https://doi.org/10.1109/TKDE.2012.232> (2014).
44. He, H., Bai, Y., Garcia, E. A., & Li, S. ADASYN: Adaptive synthetic sampling approach for imbalanced learning 1322–1328 <https://doi.org/10.1109/IJCNN.2008.4633969> (2008).
45. Breiman, L. Bagging predictors. *Mach. Learn.* **24**(2), 123–140 (1996).
46. Freund, Y. Boosting a weak learning algorithm by majority. *Inf. Comput.* **121**(2), 256–285 (1995).
47. Batista, G. E., Bazzan, A. L., & Monard, M. C. Balancing Training Data for Automated Annotation of Keywords: a Case Study. in *WOB* 10–18 (2003).
48. Wilson, D. L. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Trans. Syst. Man Cybern.* **3**, 408–421 (1972).
49. Palli, A. S., Jaafar, J., Hashmani, M. A., Gomes, H. M. & Gilal, A. R. A hybrid sampling approach for imbalanced binary and multi-class data using clustering analysis. *IEEE Access* **10**, 118639–118653. <https://doi.org/10.1109/ACCESS.2022.3218463> (2022).
50. Jiang, Z., Zhao, L., Lu, Y., Zhan, Y. & Mao, Q. A semi-supervised resampling method for class-imbalanced learning. *Expert. Syst. Appl.* **221**, 119733 (2023).
51. Jiang, Z., Zhao, L. & Zhan, Y. A boosted co-training method for class-imbalanced learning. *Expert. Syst.* **40**(9), e13377 (2023).
52. Jiang, Z., Lu, Y., Zhao, L., Zhan, Y. & Mao, Q. A post-processing framework for class-imbalanced learning in a transductive setting. *Expert Syst. Appl.* **249**, 123832 (2024).
53. Dai, Q., Wang, L.-H., Xu, K.-L., Du, T. & Chen, L.-F. Class-overlap detection based on heterogeneous clustering ensemble for multi-class imbalance problem. *Expert. Syst. Appl.* **255**, 124558. <https://doi.org/10.1016/j.eswa.2024.124558> (2024).
54. Dai, Q., Wang, L., Zhang, J., Ding, W. & Chen, L. GQEO: Nearest neighbor graph-based generalized quadrilateral element oversampling for class-imbalance problem. *Neural Netw.* **184**, 107107. <https://doi.org/10.1016/j.neunet.2024.107107> (2025).



55. Dai, Q., Zhou, X., Yang, J.-P., Du, T. & Chen, L.-F. A mutually supervised heterogeneous selective ensemble learning framework based on matrix decomposition for class imbalance problem. *Expert Syst. Appl.* **271**, 126728. <https://doi.org/10.1016/j.eswa.2025.126728> (2025).
56. Budiman, A., Fanany, M. I. & Basaruddin, C. Adaptive online sequential ELM for concept drift tackling. *Comput. Intell. Neurosci.* **2016**, 20–37 (2016).
57. Wang, S., Minku, L. L., Ghezzi, D., Caltabiano, D., Tino, P., & Yao, X. Concept drift detection for online class imbalance learning in *Proceedings of the 2013 International Joint Conference on Neural Networks (IJCNN)*, Fairmont Hotel, Dallas, Texas, USA, 1–10 (IEEE, 2013).
58. Zhang, H. & Liu, Q. Online learning method for drift and imbalance problem in client credit assessment. *Symmetry* **11**(7), 890 (2019).
59. Bifet, A. & Gavalda, R. Learning from time-changing data with adaptive windowing," in *Proceedings of the 2007 SIAM international conference on data mining* 443–448 (SIAM).
60. Pesaranghader, A. & Viktor, H. L. Fast hoeffding drift detection method for evolving data streams. In *Joint European conference on machine learning and knowledge discovery in databases* 96–111 (Springer, 2016).
61. Berger, V. W. Kolmogorov–smirnov test: Overview (2014).
62. Street, W. N. & Kim, Y. A streaming ensemble algorithm (SEA) for large-scale classification," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining* 377–382 (2001).
63. Al-Khateeb, T. et al. Recurring and novel class detection using class-based ensemble for evolving data stream. *IEEE Trans. Knowl. Data Eng.* **28**(10), 2752–2764. <https://doi.org/10.1109/TKDE.2015.2507123> (2016).
64. Brzezinski, D. & Stefanowski, J. Reacting to different types of concept drift: The accuracy updated ensemble algorithm. *IEEE Trans. Neural Netw. Learn. Syst.* **25**(1), 81–94. <https://doi.org/10.1109/TNNLS.2013.2251352> (2014).
65. Farid, D. M. et al. An adaptive ensemble classifier for mining concept drifting data streams. *Expert. Syst. Appl.* **40**(15), 5895–5906 (2013).
66. Bernardo, A. & Valle, E. D. VFC-SMOTE: Very fast continuous synthetic minority oversampling for evolving data streams. *Data Min. Knowl. Discov.* **35**(6), 2679–2713. <https://doi.org/10.1007/s10618-021-00786-0> (2021).
67. Roseberry, M., Krawczyk, B. & Cano, A. Multi-label punitive kNN with self-adjusting memory for drifting data streams. *ACM Trans. Knowl. Discov. Data* **13**(6), 1–31. <https://doi.org/10.1145/3363573> (2019).
68. Korycki, L. & Krawczyk, B. Concept drift detection from multi-class imbalanced data streams. *IEEE Comput. Soc.* **2021**, 1068–1079. <https://doi.org/10.1109/ICDE51399.2021.00097> (2021).
69. Lu, Y., Cheung, Y. M. & Tang, Y. Y. Adaptive chunk-based dynamic weighted majority for imbalanced data streams with concept drift. *IEEE Trans. Neural Netw. Learn. Syst.* **31**(8), 2764–2778. <https://doi.org/10.1109/TNNLS.2019.2951814> (2020).
70. Usman, M. & Chen, H. Intensive class imbalance learning in drifting data streams. *IEEE Trans. Emerg. Top. Comput. Intell.* **8**(5), 3503–3517 (2024).
71. Chen, Y., Yang, X. & Dai, H.-L. Cost-sensitive continuous ensemble kernel learning for imbalanced data streams with concept drift. *Knowl.-Based Syst.* **284**, 111272 (2024).
72. Brzezinski, D., Minku, L. L., Pewinski, T., Stefanowski, J. & Szumaczuk, A. The impact of data difficulty factors on classification of imbalanced and concept drifting data streams. *Knowl.-Based Syst.* **63**(6), 1429–1469. <https://doi.org/10.1007/s10115-021-01560-w> (2021).
73. Ghazikhani, A., Monsefi, R. & Yazdi, H. S. Online neural network model for non-stationary and imbalanced data stream classification. *Int. J. Mach. Learn. Cybern.* **5**(1), 51–62 (2014).
74. Shen, L., Er, M. J., Liu, W., Fan, Y. & Yin, Q. Population structure-learned classifier for high-dimension low-sample-size class-imbalanced problem. *Eng. Appl. Artif. Intell.* **111**, 104828. <https://doi.org/10.1016/j.engappai.2022.104828> (2022).
75. Raghuwanshi, B. S. & Shukla, S. UnderBagging based reduced Kernelized weighted extreme learning machine for class imbalance learning. *Eng. Appl. Artif. Intell.* **74**, 252–270. <https://doi.org/10.1016/j.engappai.2018.07.002> (2018).
76. Korycki, L. & Krawczyk, B. Concept Drift Detection from Multi-Class Imbalanced Data Streams. *arXiv preprint arXiv:2104.10228* (2021).
77. Palli, A. S. et al. Combined effect of concept drift and class imbalance on model performance during stream classification. *CMC-Comput. Mater. Contin.* **75**(1), 1827–1845 (2023).
78. Palli, A. S., Jaafar, J. & Hashmani, M. A. Performance Degradation of Multi-class Classification Model Due to Continuous Evolving Data Streams. In *International Conference of Reliable Information and Communication Technology* 696–706 (Springer, 2020).
79. Ancy, S. & Paulraj, D. Handling imbalanced data with concept drift by applying dynamic sampling and ensemble classification model. *Comput. Commun.* **153**, 553–560 (2020).
80. Nitish, A., Hanumanthappa, J., Prakash, S. S. & Krinkin, K. Class imbalance and concept drift invariant online botnet threat detection framework for heterogeneous IoT edge. *Comput. Secur.* **141**, 103820 (2024).
81. Bahri, M., Bifet, A., Gama, J., Gomes, H. M. & Maniu, S. Data stream analysis: Foundations, major tasks and tools. *WIREs Data Min. Knowl. Discov.* **11**(3), <https://doi.org/10.1002/widm.1405> (2021).
82. Woźniak, M., Zyblewski, P. & Ksieniewicz, P. Active weighted aging ensemble for drifted data stream classification. *Inf. Sci.* **630**, 286–304 (2023).
83. Wang, H., Fan, W., Yu, P. S. & Han, J. Mining concept-drifting data streams using ensemble classifiers. in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* 226–235 (2003).
84. Brzezinski, D. & Stefanowski, J. Accuracy updated ensemble for data streams with concept drift. In *Hybrid Artificial Intelligent Systems: 6th International Conference, HAIS 2011, Wrocław, Poland, May 23–25, 2011, Proceedings, Part II* 6 155–163 (Springer, 2011).
85. Cano, A. & Krawczyk, B. Kappa updated ensemble for drifting data stream mining. *Mach. Learn.* **109**, 175–218 (2020).
86. Liu, W., Zhang, H., Ding, Z., Liu, Q. & Zhu, C. A comprehensive active learning method for multiclass imbalanced data streams with concept drift. *Knowl.-Based Syst.* **215**, 106778. <https://doi.org/10.1016/j.knosys.2021.106778> (2021).
87. Domingos, P. & Hulten, G. Mining high-speed data streams. in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining* 71–80 (2000).
88. Gomes, H. M. et al. Adaptive random forests for evolving data stream classification. *Mach. Learn.* **106**(9–10), 1469–1495 (2017).
89. Bayram, F., Ahmed, B. S. & Kassler, A. From concept drift to model degradation: An overview on performance-aware drift detectors. *Knowl.-Based Syst.* **245**, 108632. <https://doi.org/10.1016/j.knosys.2022.108632> (2022).
90. Nguyen, H.-L., Woon, Y.-K., Ng, W.-K. & Wan, L. Heterogeneous ensemble for feature drifts in data streams. In *Advances in Knowledge Discovery and Data Mining: 16th Pacific-Asia Conference, PAKDD 2012, Kuala Lumpur, Malaysia, May 29–June 1, 2012, Proceedings, Part II* 16 1–12 (Springer, 2012).
91. Gama, J., Medas, P., Castillo, G. & Rodrigues, P. Learning with drift detection. In *Brazilian symposium on artificial intelligence* 286–295 (Springer, 2004).
92. Wang, B. & Pineau, J. Online bagging and boosting for imbalanced data streams. *IEEE Trans. Knowl. Data Eng.* **28**(12), 3353–3366 (2016).
93. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M. & Bouchachia, A. A survey on concept drift adaptation. *ACM Comput. Surv. (CSUR)* **46**(4), 1–37 (2014).
94. Zyblewski, P., Sabourin, R. & Woźniak, M. Preprocessed dynamic classifier ensemble selection for highly imbalanced drifted data streams. *Inf. Fus.* **66**, 138–154. <https://doi.org/10.1016/j.inffus.2020.09.004> (2021).

95. Bifet, A. et al. Moa: Massive online analysis, a framework for stream classification and clustering. in *Proceedings of the first workshop on applications of pattern analysis* PMLR 44–50 (2010)
96. Liu, W., Zhang, H., Ding, Z., Liu, Q. & Zhu, C. A comprehensive active learning method for multiclass imbalanced data streams with concept drift. *Knowl.-Based Syst.* **215**, 106778. <https://doi.org/10.1016/j.knosys.2021.106778> (2021).
97. Mazurowski, M. A. et al. Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural Netw.* **21**(2–3), 427–436 (2008).

## Acknowledgements

The authors would like to express their sincere appreciation to Universiti Teknologi PETRONAS and Universiti Kebangsaan Malaysia (UKM) for their institutional support and encouragement throughout this project. This research was financially supported by Universiti Kebangsaan Malaysia (UKM) under Grant Code DIP-2023-026, and by Universiti Teknologi PETRONAS through the Graduate Research Assistance Program.

## Author contributions

A.S.P. : Writing—review & editing, Writing—original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Conceptualization. J.J. : Writing—review & editing, Writing—original draft. Mohamad Hanif Md Saad : Methodology, Investigation, Formal analysis, Conceptualization. A.A.M. : Writing review & editing, Formal analysis. H.M.G. : Writing review & editing, Formal analysis. A.A.S. : Formal analysis, Conceptualization. Writing review & editing, Formal analysis. A.R.G. : Writing review & editing, Formal analysis.

## Funding

This research was financially supported by Universiti Kebangsaan Malaysia (UKM) under Grant Code DIP-2023-026, and by Universiti Teknologi PETRONAS through the Graduate Research Assistance Program.

## Declarations

## Competing interests

The authors declare no competing interests.

## Additional information

**Correspondence** and requests for materials should be addressed to A.S.P. or A.A.S.

**Reprints and permissions information** is available at [www.nature.com/reprints](http://www.nature.com/reprints).

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Open Access** This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2025