International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

A New Large Scale SVM for Classification of Imbalanced Evolving Streams

D. Himaja[1*], Dondeti.Venkatesulu[2], Uppalapati. Srilakshmi[3]
Vignan's Foundation for Science and Technology (Deemed to be University),
Guntur, Andhra Pradesh, India.
*Corresponding author E-mail: himajadirsumilli@gmail.com

Abstract
Classification from imbalanced evolving streams possesses a combined challenge of class imbalance and concept drift (CI-CD). However, the state of imbalance is dynamic, a kind of virtual concept drift. The imbalanced distributions and concept drift hinder the online learner's performance as a combined or individual problem. A weighted hybrid online oversampling approach,"weighted online oversampling large scale support vector machine (WOOLASVM)," is proposed in this work to address this combined problem. The WOOLASVM is an SVM active learning approach with new boundary weighing strategies such as (i) dynamically oversampling the current boundary and (ii) dynamic weighing of the cost parameter of the SVM objective function. Thus at any time step, WOOLASVM maintains balanced class distributions so that the CI-CD problem does not hinder the online learner performance. Over extensive experiments on synthetic and real-world streams with the static and dynamic state of imbalance, the WOOLASVM exhibits better online classification performances than other state-of-the-art methods.

Keywords: Online learning, Dynamic class imbalance, Active learning, Support Vector Machines, Oversampling

1 Introduction

In real-life scenarios like credit card fraud, intrusion and spam detections, the data evolves like imbalanced evolving streams where the problem of class imbalance and concept drift coexists. However, the literature addresses both of these problems independently.

The class imbalance learning (CIL) problem on standalone training sets arises when the majority of data belongs to one class, degrading the performance of the minoritized class. This issue has received much attention, and numerous techniques were developed at data and algorithm levels [1]. Oversampling and/or undersampling procedures are used at the data level to balance the class distributions. The algorithm level solutions modify the decision boundary towards the minority class by assigning extra costs or parameter weights that reflect the minority class. The degree of imbalance drives most of these methods.

On the other hand, the concept drift (CD) in evolving streams occurs when the statistical properties of the underlying functions change over time. According to [2], two types of concept drifts such as (i) real and (ii) virtual are possible. Let the training set with predicting and target variables x and y. In real drift, also referred to as posterior drift (i.e., $p(y/x)$), distributions changes lead to decision boundary changes. In contrast, positional shift such as a change in prior probability (i.e., $p(y)$) happens which causes imbalance drift [3] in virtual concept drift. Unlike the conventional problem in the evolving imbalanced stream, the class imbalance is inconsistent and varies from time to time dynamically. It is possible that over time, the minority samples will become the majority, and vice versa. As a result, the learner does not know the status of imbalance in advance. Online learners must adapt to these dynamic changes [3]. Usually, there are three types of drifts. In abrupt drifts, there is a sudden change in concept. The change is gradual in gradual drifts. Drifts recur over time in recurrent drifts.

From the standalone class imbalance problem [4], apart from the data features like degree of imbalance, size of the dataset, and the degree of overlapping, the classifier also implicates performance degradation. For imbalanced evolving streams, solutions like recursive least square adaptive cost perceptron (RLSACP) [5] and online neural network (ONN) [6], and a weighted online sequential extreme learning machine (WOS-ELM) [7] are suggested based on NN. Further online bagging [8] and extreme learning based ensembles [9] are tailored to address this problem.

However, in the case of standalone class imbalance problem, SVM exhibits better generalization abilities than other conventional classification algorithms [4, 10]. Further, it is also extensively used in active learning [11] as the distance between a data point to the hyperplane is calculated directly. SVM active learning methods progressively enhance the model performances by incremental boundary updates. Therefore this work focuses on SVM based active learning solution to the combined problem of class imbalance and

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

virtual concept drift (i.e., imbalance drift *p(y)*) in imbalanced evolving streams. The following are the paper's significant contributions:

1) A weighted oversampling large scale support vector machine to cope with evolving imbalanced streams.
2) A dynamic oversampling strategy to balance the boundary at any time step *t*.
3) A dynamic asymmetric cost weighing strategy for SVM online imbalance stream learning. The remainder of this paper is structured as follows. The related work on the online dynamic class imbalance problem is presented in section 2. Section 3 addresses the motivation for our work and the required background techniques. The proposed algorithm is shown in section 4. Section 5 discusses the obtained results, and finally, section 6 brings the paper to a close.

2 Related Work

Concerned about the combined problem, handling either concept drift or class imbalance prior is a critical issue [3]. However, detecting the *p(y/x)* prior is prone to the impact of class imbalance. The solutions proposed so far indicate that the class imbalance in static and dynamic states should be addressed first.

Drift detection can be handled in two ways [12]: active or passive. In the former methods, the learner first detects the drift and then it is updated to cope with the data changes. Unlike the former techniques, the latter approaches update the model implicitly. Based on the drift handling criteria, the solutions that address the imbalanced evolving streams are arranged under three categories: (1) static class imbalance-no drift, (2) dynamic class imbalance (i.e., imbalance drift (*p(y)*)), and (3) *p(y/x)* drift detection methods.

2.1 Static class imbalance- no drift

Learn++. CDS and Learn++. NIE [13] are two oversampling ensembles extended from Learn++, proposed for handling concept drift. The static state of imbalance is handled with synthetic minority oversampling techniques in these approaches. Other ensemble approaches such as SERA and REA [14, 15] learn from balanced class distributions obtained by propagating informative minority samples from previous to the current batch. These minority samples are selected on their similarity with the current batch minority samples. Here, the concept drift is addressed as the majority voting over-weighted soft-typed hypothesis (i.e., sample's maximum likelihood that belongs to one of the classes) so far learned. In another proposal, HUWRS [16] adapts the random subspace ensemble method for identifying drifting features. Here the drift detection is carried out using hellinger distance. In this approach, whenever a feature drift is triggered, the corresponding classifier in the ensemble only gets reset based on its weights. The extended version of HUWRS [16], HUWRS. IP [17] introduces an instance propagation mechanism using a naive bayes classifier to select old minority samples for achieving more balanced distributions.

Wang et al. [18] proposed k-means clustering undersampling-based ensemble to cope with evolving imbalanced streams with concept drift. However, an additional buffering mechanism is required for batch learning to maintain the samples. Single classifier variants like RLSACP [5] and ONN [6] are based on NN. In this case, the class imbalance is addressed with an adaptive weighing of perceptron error either with classification rate or the degree of imbalance. Mirza et al. [7] proposed WOS-ELM for class imbalance learning. However, the WOS-ELM can address the class imbalance problem in both batches and online fashion. In order to detect *p(y/x)* drift, it is extended to an ensemble of the subset of the online sequential extreme learning machine (ESOS-ELM) [9], where each classifier from a pool of ensemble is learned with a balanced class distribution. Whenever a *p(y/x)* drift gets detected, a new learner is appended to the learner's pool, and the dynamic voting of the majority is used to update the learner's weights. The learners with the least weights in the current ensemble are saved in ELM-store to cope with the recurring concept. Any classifier in the store that performs better than the current ensemble indicates that the concept has occurred again, and the best classifier is added to the current ensemble. Both threshold and hypothesis tests [12] based drift detection methods are used subsequently to cope with abrupt and gradual drifts. However, this method handles the static imbalance with the concept drift problem (CI-CD).

The cost-sensitive adaptive random forest (CSARF) proposed in [19], is a variant of ARF. In CSARF, matthews correlation coefficient (MCC) assigns weights to each tree in the ensemble instead of accuracy. A sliding window was introduced to eliminate the problem of not presenting minority class instances for some ensemble classifiers. The learning process is modified so that each tree is trained with minority class samples. Cost sensitivity assignments can be done with local (CSARF-local) or global (CSARF-global) strategies. CSARF-local uses misclassification costs to influence the output of the base classifiers before the combination of votes is considered. In contrast, CSARF-global uses the final mix of votes from all the decision trees but not individually.

2.2 Dynamic class imbalance (i.e., imbalance drift)

Wang et al. [8] proposed several resampling-based ensemble methods by extending online bagging, such as weighted online bagging, undersampling, and oversampling online bagging (OOB and UOB), to cope with the CI-CD problem. Here the oversampling and undersampling rates for

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

online bagging are directed by time decay class size. Inline, sukarna et al. [20] proposed a generalized oversampling-based online imbalanced learning framework (GOS-IL) for online learners to handle *p(y)* drift.

2.3 *p(y/x)* drift detection methods

From [3, 21], as the *p(y/x)* drift is prone to class imbalance problem, several approaches to detect *p(y/x)* were proposed. Without prior imbalanced treatment, the drift is biased towards the majority class [3, 21]. The solutions that are proposed are the drift detection method online class imbalance (DDM-OCI) [21] is a modification to DDM [22]. Unlike DDM, whose concentration is on change detection on overall error rate, DDM-OCI tracks true positive rate (TPR) changes. Instead of only tracking changes in TPR, a linear four rate tracking mechanism for drift detection is proposed [23]. The drift signal is activated if a significant change in any rate updates is detected. Brzezinski and stefanowski [24] proposed a prequential AUC-based drift detection mechanism that recognizes the drift in prequential AUC by the page-hinkley test. In [25], a two-layer drift detection method is proposed. Layer 1 incorporates LFR, layer 2 is based on the permutation test, and both are supervised. However, the high degree of imbalance nullifies the target class performance, making it challenging to identify the drift.

Wang et al. extended AUC as prequential multi-class AUC (PMAUC), weighted AUC (WAUC), and equal weighted AUC (EWAUC) for multi-class classification [26]. A heterogeneous dynamic weighted majority (HDWM) [27] is proposed to add a new base learner to the current ensemble whenever performance degradation is observed. This base learner selection is made automatically from the pool of existing learners. It supports both active and passive approaches. Recently, in SDDM [28], statistical methods that identify distribution differences are used for drift detection. In [29], cluster-based distance methods detect recurring concept drifts.

Except wang et al. [8], the rest of the methods assume the stream evolves with the static imbalance and *p(y/x)* drift. However, this is an online bagging based ensemble that consumes more computational cost. On the other hand, the support vector machine (SVM) proved less sensitive to the class imbalance problem [20]. As imbalanced evolving data streams are considered, the SVMs are not well investigated. Therefore, this work has developed an online active learning-based oversample hybrid SVM to cope with evolving imbalanced streams. However, undersampling is not studied. A shift in decision boundary [30] happens due to information loss [31]. Table 1 shows the summary of all the related works. As there are limited contributions and primarily ensemble approaches, this work focuses on a single classifier based solution to address the combined problem (CI-CD).

Table 1 Summary of all the related works

Method	Type of Learning		State of Imbalance		Drift		Drift Detection	
	Batch	Online	Static	Dynamic	*p(y/x)*	*p(y)*	Active	Passive
[13]	✓		✓		✓			✓
[14,15]	✓		✓		✓			✓
[16]	✓		✓		✓		✓	
[17]	✓		✓		✓		✓	
[18]	✓		✓		✓			✓
[5,6]		✓	✓		✓			✓
[7]	✓	✓	✓		-	-		✓
[9]	✓	✓	✓		✓		✓	
[12]	✓	✓	✓		✓	✓	✓	
[19]	✓		✓		-	-		✓
[8]		✓	✓	✓		✓		✓
[20]		✓	✓	✓		✓		✓
[21]		✓	✓		✓		✓	
[23]	✓	✓		✓	✓	✓	✓	
[24]		✓	✓	✓	✓	✓	✓	
[26]		✓	✓	✓	✓	✓	✓	
[27]	✓		✓		✓		✓	✓
[28]	✓		✓		✓		✓	
[29]	✓		✓		✓		✓	

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

3 Motivation and Background

3.1 Motivation

Generally, active learning methods are used to reduce annotation costs [32]. Unlike active learning with other learners, active learning with SVM (AL-SVM) selects the informative samples closest to the current model's hyperplane [11, 33] incrementally. Due to this behavior, AL-SVM is used as an informative undersampling criterion [34] for solving standalone class imbalance problems.

However, the SVM active learning with standard SVM tools [35, 36] is computationally expensive as it is needed to solve the optimization from the beginning for each boundary update. In contrast, the LASVM (large scale support vector machine) [37] speeds up this process by simply extending from the previous α values. In this paper, the LASVM is tailored to address the following issues in evolving streams:

i) Static class imbalance: The rate of change of imbalance is static throughout the stream.

ii) Dynamic class imbalance (i.e., imbalance drift *p(y)*): The rate of change of imbalance varies with time, and sometimes the minority class becomes the majority and vice versa. However, this dynamic state of change of imbalance undergoes abrupt and gradual speeds. Here it is referred to as imbalance drift.

To address the above issues, the work proposes hybrid oversampling-based asymmetrically weighted LASVM (WOOLASVM) for solving the CI-CD problem.

3.2 Background

Given a training set $T(x_i,\ y_i)$, where $x_i \in X, y_i \in Y\{Y = 1\ or\ -1\}$, the support vector machine (SVM) [38] describes an optimal hyperplane as described in equation 1.

$$\min_{w,b,\xi_i} \frac{1}{2} w.w^T + C\sum_{i=1}^{N} \xi_i \qquad (1)$$

Subject to

$$\begin{cases} \forall i \quad y_i\left(w^T x_i + b\right) \geq 1 - \xi_i \\ \forall i \quad \xi_i \geq 0 \end{cases}$$

Where *w* is the norm of the hyperplane, *b* is the intercept of hyperplane from the origin, x_i is the input vector, y_i is the corresponding class label, ϵ is the slack variable for handling nonlinearity, and *C* is the tuning parameter for corresponding loss function of misclassification cost. Generally, equation 1 is solved by convex quadratic programming (*QP*). The sequential minimum optimization (*SMO*) [39] is widely used for solving this *QP* and works based on the principle of directional search in identifying a τ - violating pair. Here τ is the tolerance on the gradients of the pair *(i, j)*, $g_i - g_j > \tau$ where *g* represents the corresponding gradient. The SMO halts when there is no such τ - violating pair, and the final equations after solving *QP* are:

$$w = \sum_{i=1}^{N} \alpha_i y_i \Phi(x_i) \qquad (2)$$

and the test samples are predicted with

$$y(x) = sign[\sum_{i=1}^{N} \alpha_i y_i k(x_i, x) + b] \qquad (3)$$

α *'s* with non-zero values formulate the boundary for SVM and are called support vectors (*SVs*). In the case of incremental learning, methods such as active learning [11] take out this as an advantage and incrementally update the boundary by querying the unseen samples that are exactly nearer to the current decision boundary. The queried samples become the support vector for the new boundary. For each of this updation, using SVM tools based on *SMO* optimizer solves the *QP* from the beginning by assuming the training set as the new. Hence, these tools are not suitable to scale large data sets. However, *LASVM* [37] *Process* the new sample by adding it to the current support vector set *S*. Next, it *ReProcesses* by eliminating some blatant nonsupport vectors from *S*.

For every occurrence of the new sample, processing and reprocessing of the boundary, i.e., updating the support vector set *S* occurs. The optimization phase extends the gradient and α computations from previous α values and *S*, which *SMO* computes. In the *Process* step of LASVM, at first, the new instance *k* is added to *S*. Then search is being carried out for *(i, j)* in *S*, which could satisfy the τ - violating pair condition with maximum gradient and using directional search, $\alpha\ 's$, g_s are updated. If *(i, j)* is not a τ -violating pair, it is bailed out from *S*. Equations 4 and 5 depict the new gradient initialization and τ -violating pair identification.

$$g_k = y_k - \textstyle\sum_{s\in S} \alpha_s K_s \qquad (4)$$

$$\text{i} = \begin{cases} k,\ j = argmin_{s\in S} g_s\ with\ \propto_s > A_s\ \text{y}_\text{k} = +1 \\ k,\ j = argmin_{s\in S} g_s\ with\ \propto_s < B_S \qquad otherwise \end{cases} \qquad (5)$$

The *ReProcess* step of *LASVM* removes any other nonsupport vectors from *S*. It initially searches for τ - violating pair from *S* with maximum gradient. Then, the directional search is carried out to update $\alpha\ 's$, g_s. During this process, the identified blatant nonsupport vectors are removed.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

$$\begin{cases} \text{i} = argmin_{s\in S} g_s \; with \; \propto_s < B_z, \text{j} = argmin_{s\in S} g_s \; with \; \propto_s > A_s \\ \forall s \in S : \alpha_s = 0 \\ S = \begin{cases} S - \{s\} y_s = -1 \bigcap g_s \geq g_i \\ S - \{s\} y_s = +1 \bigcap g_s \leq g_j \end{cases} \end{cases}$$

(6)

The final computation of bias b, the gradient of most τ - violating pair in S, Δ are calculated as

$$b = \frac{(b_i - b_j)}{2}, \; \Delta = (b_i - b_j) \quad (7)$$

Therefore the learning is faster, and the boundary adaptively changes with the new data samples. Usually, in standalone training sets, to cope with imbalance, the boundary of the SVM is weighted with costs to prioritize better minority class prediction. The cost–sensitive SVM formulation of class imbalanced problems is in [40]. The cost parameters 'C' of SVM (equation 1) is further subdivided into C_+ and C_- are weighted asymmetrically. Based on the imbalance ratio, higher weights are given to the cost parameters C_+ than C_- (Equation 8)

$$\min_{w,b,\xi_i} \frac{1}{2} w.w^T + C_+ \sum_{i=1}^{N} \xi_i + C_- \sum_{i=1}^{N} \xi_i \quad (8)$$

$$\text{Subject to} \begin{cases} \forall i \quad y_i \left(w^T x_i + b \right) \geq 1 - \xi_i \\ \forall i \quad \xi_i \geq 0 \end{cases}$$

4 WOOLASVM

However, from wu et al. [41], the SVM on highly imbalanced datasets results in a skewed boundary. As the data becomes more skewed, so does the ratio of positive to negative support vectors. This leads to poor minority class performance. However, these conclusions are still valid for SVM online learning with imbalanced evolving streams. In addition to this, there are other challenges with evolving streams. The entire distribution parameters may not be available prior. Whenever the underlying distribution changes with time, i.e., the data at a time *(t-1)* may not be valid at *t* (as per *p(y)* change). To cope with this combined issue, a weighted online oversampling based large scale support vector machine (WOOLASVM) is proposed in this paper. For WOOLASVM, the boundary of the LASVM is weighted in two folds:

(1) Oversample the SVM boundary with respect to the imbalance ratio of current time *t*.

(2) The asymmetric cost parameters of the SVM objective function (equation 8) C_+ and C_- are weighted with dynamic cost weighing factors w_+, w_-. These weighing factors reflect the current state of imbalance in the stream at time *t*.

Figure 1 illustrates a sample scenario of WOOLASVM, and algorithm 1 shows its pseudo-code. Initially, an imbalance boundary is assumed with at most a single minority and few majority samples for the learner (see figure 1(a)). To classify the class y_t of a test sample $< x_t, y_t >$, the initial training set $< x_{t-1}, y_{t-1} >$ is balanced by oversampling [42] half the pathway between minority class and majority class samples using *half_generate_synthetic (x_{min}, y_{min}, x_{kmaj} , y_{kmaj} , d)* function where *d* indicates the current degree of imbalance (i.e., $\frac{No.of\ positives}{No.of\ Negatives}$). This formulates a balanced boundary (see figure 1(b)). The following incoming sample is predicted on this newly formulated balanced boundary (see figure 1 (c)) and then appended to the support vector set using the *Process* step (see figure 1(d)). Then the latest boundary is polished by pulling out some blatant nonsupport vectors using *ReProcess* step (see figure 1(e)).

Due to these repeated steps of *Process* and *ReProcess*, the boundary goes imbalanced (see figure 1(e)). To alleviate this problem, the boundary is further asymmetrically weighted (C_+, C_-) with the class-specific weighing factor w_k (i.e., w_+, w_- *a*nd +, – are related to positive and negative class) with respect to current time *t*. w_+, w_- are calculated using equation 9.

$$w_k^t = \theta w_k^{t-1} + (1 - \theta)[(y_t, c_k)] \quad k=1,2...n \quad (9)$$

Here $[< y_t, c_k >] = 1$ when $y_t = c_k$ otherwise 0, *k* represents the corresponding class. The θ acts as the forgetting factor ($0 < \theta < 1$) that enforces more weight on the samples of current significance than the older ones. These weights are used to weigh the cost parameters of equation 8. Based on imbalance drift, if $w_+ < w_-$ then C_+ is weighted with $\frac{w-}{w+}$, or else C_- is weighed with $\frac{w+}{w_-}$ and counterpart becomes 1. This sort of weighing minimizes the impact of class imbalance related to past data on equation 8.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

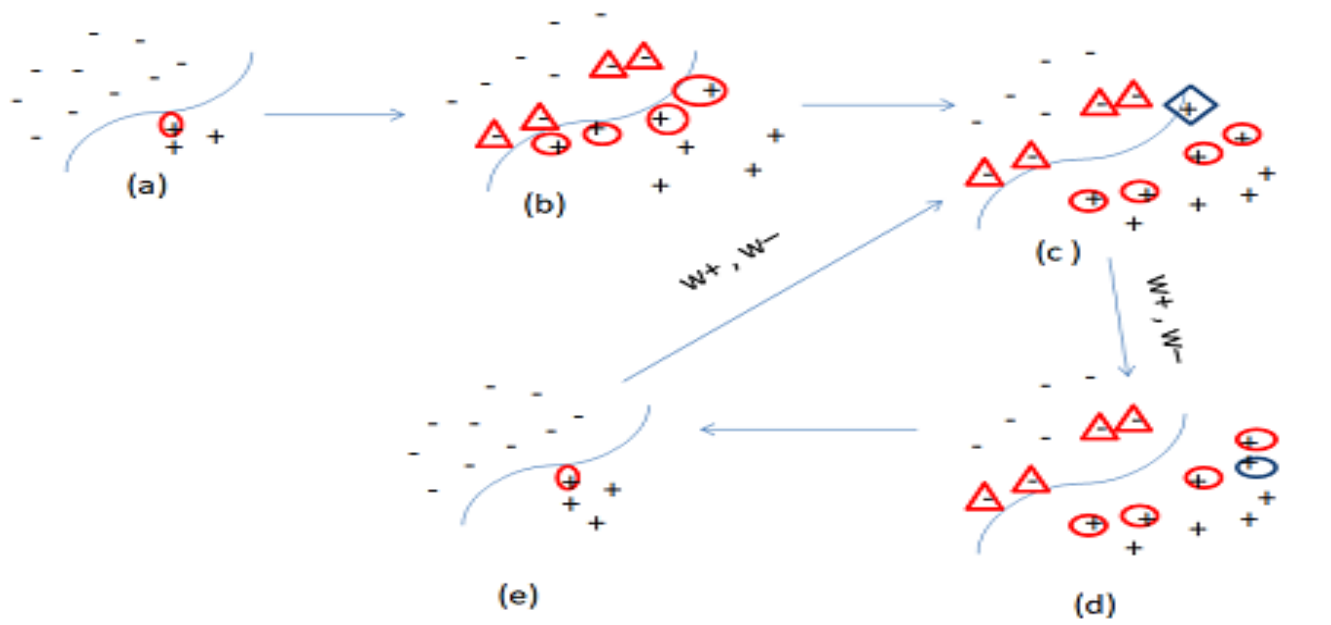

Figure 1 Example scenario with WOOLASVM

For further boundary refinements, apart from the asymmetric weighing SVM objective function, the dynamic oversampling is based on the current imbalance drift in the stream. If $nsvt_- > nsvt_+$ or vice versa happens where nsv_t contains the total support vectors at a time *t*, then *generate_synthetic* (x_{min}, y_{min}, x_{kmaj}, y_{kmaj}, $_d$) is called. This function generates synthetic minority samples around the current support vectors to bridge the difference between $nsvt_-$, $nsvt_+$. WOOLASVM's operational model is as follows:

Step 1: To avoid skewness (see Figure 1(a)), the boundary is balanced by oversampling minority samples half the pathway between minority and majority samples. The *Process* (*New TrainingSet*) adds these generated samples to the existing support vector set. Figure 1(b) shows the formulation of the new balanced boundary. Using training set samples, a confusion matrix is created.

Step 2: The new samples in the evolving stream are predicted over this balanced boundary (see Figure 1(c)). Performance evaluation indicators are updated accordingly. Based on the imbalance drift, cost parameters (i.e., C_+ and C_-) are calculated from cost weighing factors (i.e., w_+ and w_-). If $w_+ < w_-$ then C_+ is weighed with $\frac{w-}{w+}$ otherwise C_- is weighed with $\frac{w+}{w-}$ and counterpart becomes 1. The current sample is now appended to the *SVs* set through *Process* (<*NewSample* >, < C_+, C_->) which results in the next model (figure 1(d)).

Step 3: The new model is updated using *ReProcess* ((< *NewModel* >, <C_+, C_->)) *function* to remove blatant nonsupport vectors (See Figure 1(e))). After removing the blatant nonsupport vectors, the new boundary may become imbalanced (see Figure 1(e)). As it is online learning on the imbalance stream, there may not be enough minority samples. Pulling out a single minority by *ReProcess* has a major impact.

Step 4: At current time *t*, the positive class may become the majority or minority as the stream emerges with a dynamic degree of imbalance *d*. This is addressed by creating *d* artificial *SVs* focused around minority class *SVs* (positive or negative). If ($nsv_+ < nsv_-$) then artificial samples are created across positive class else across the negative class. At the end of each *ReProcess*, the positive class and negative class *SVs* (nsv_+, nsv_-) are updated and can be obtained by [nsv_+, nsv_-]=*get_sv(New Model)*.

Step 5: Calculate new <C_+, C_->. As the minority samples are generated synthetically, to manage the penalty, the *SVs* imbalance ratio is assigned for the cost parameter. The latest boundary is generated with the artificial samples by *NewModel = Process* (< *synth_data* >, <C_+, C_->) (See Figure 1(e) to 1(c)). Steps 2 through 5 are repeated until the last sample is obtained.

In LASVM, the *Process* and *ReProcess* steps end up with an imbalanced boundary. In contrast, the initial boundary is balanced through generating synthetic samples half the pathway between minority and majority *SVs* and in further boundary updates, based on the current state of imbalance (both static and dynamic) oversampling around the *SVs* as well as the asymmetric weighing of the objective function takes place to enrich the minority class prediction in WOOLASVM. Thus WOOLASVM enables better minority class prediction over balanced boundaries than LASVM.

Algorithm 1 WOOLASVM

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
Article Received: 10 June 2022 Revised: 15 July 2022 Accepted: 30 July 2022 Publication: 31 August 2022

Input: $< x_i , y_i >$ training set with degree of imbalance *d*,
$< x_{min}, y_{min} >$ are corresponding minority samples,
$< x_{maj} , y_{maj} >$ are corresponding majority samples,
+ is positive class and - is negative class
Output: *GMean* and *Recall*

1: for each minority sample in $< x_i, y_i >$
2: synth_data = half_generate_synthetic ($< x_{min}, y_{min} >$, $< x_{kmaj}, y_{kmaj} >$, *d)*
3: *New Traningset*= [$< x_{kmaj}, y_{kmaj} >$, $< x_{min}, y_{min} >$, < *synth_data* >]
4: *New Model=Process* (New *TraningSet*)
5: *[nsv+, nsv–] = get_sv (New Model)*
6: **while** samples **do**
7: Obtain $< x_t, y_t >$ from the evolving stream
8: Classify $< x_t, y_t >$ on *New Model*
9: Compute *Recall* and *GMean*
10: Compute $< w_+, w_- >$ using Equation 9
11: **if** ($w_+ < w_- >$ then
12: $C_+ = \frac{w-}{w+}$ and $C_-=1$
13: **else if** ($w_- < w_+$) then
14: $C_- = \frac{w+}{w-}$ and $C_+=1$
15: **else**
16: $C_+=1$ and $C_-=1$
17: **end if**
18: *New Model=Process* ($< x_t, y_t >$, $< C_+, C_- >$)
19: Update *New Model=ReProcess (New Model*, $< C_+, C_- >$*)*
20: Repeat *ReProcess* until $\delta \leq \tau$
21: *[nsv+, nsv–] = get_sv (New Model)*
22: **if** ($nsv_+ < nsv_-$) then
23: $d = \frac{nsv-}{nsv+}$
24: *synth_data = generate_synthetic* ($< x_{(sv+)}, y_{(sv+)} >$, $< x_{(ksv+)}, y_{(ksv+)} >$, *d)*
25: $C_+ = \frac{nsv-}{nsv+}$ and $C_-=1$
26: *New Model=Process* ($< synth\_data >$, $< C_+, C_- >$)
27: **end if**
28: **if** ($nsv_- < nsv_+$) then
29: $d = \frac{nsv+}{nsv-}$
30: *synth_data = generate_synthetic* ($< x_{(sv-)}, y_{(sv-)} >$, $< x_{(ksv-)}, y_{(ksv-)} >$, *d)*
31: $C_- = \frac{nsv+}{nsv-}$ and $C_+=1$
32: *New Model=Process* ($< synth\_data >$, $< C_+, C_- >$)
33: **end if**
34: **end while**

5 Experimental Results and Discussion

The proposed algorithms are validated on the four cases of imbalance with *p(y)* and *p(y/x)* drifts over synthetic and real-world streams. The synthetic data generation procedure for four cases of imbalance is described below:

5.1 Data set Preparation

Case (i): Static class imbalance streams, generated by constant imbalance ratio throughout the stream.

Case (ii): Dynamic class imbalance streams, generated by minority class percentages that vary with respect to time, i.e., imbalance drift (*p(y)*). The sudden imbalance drift starts exactly in the middle of the stream, and the gradual drift continues after the middle of the stream till 300 time steps of stream size 1000.

Case (iii): Static class imbalance streams with *p(y/x)* drift, generated by constant imbalance ratio throughout the stream, abrupt concept drift starts exactly in the middle of the stream and gradual starts in the middle of the stream, and continues after the middle of the stream till 250 time steps of stream size 1000.

Case (iv): Dynamic class imbalance streams (i.e., *p(y)* with *p(y/x)* drift), generated with varied imbalance ratios, both the imbalance and concept drift start exactly in the middle of the stream, and in gradual case, drift (both imbalance and concept drift) start at the middle of the stream and continues till 250 time steps of stream size 1000.

Different synthetic data streams are generated for the above four cases with different minority class percentages, say 10%, 20%, 30%, 40%, and 50%. The total number of data streams generated for experimentation in this work is 141. The size of each synthetic data stream is fixed to 1K. The real-world data sets are COVER TYPE, SHUTTLE, SMART BUILDING, and ELECTRICITY. The size of COVER TYPE is 5, 81,012 with 1% imbalance percentage, and it is a multi-class dataset. Binary class classification problem is considered in this work. Multi-class datasets are converted to binary class by selecting the category with the smallest number of instances as minority and the rest as majorities. Table 2 depicts the characteristics of the datasets used in this work. The synthetic data streams are generated using a framework such as MOA [43] and minku et al. [44].

5.2 Experimental Settings

The evaluation prequential [45] of individual class *Recall* and classifier *GMean* are used as performance indicators. As two-class information is necessary, the WOOLASVM is initially given an input with nine instances where eight instances are negative, and one is positive, assuming that there is only a sample available at initial training. Proposed WOOLASVM is compared with the online bagging method OOB [8], which handles the online dynamic CIL problem. However, the dynamic CIL problem

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

is concerned, OOB is a benchmark approach. In WOOLSAVM w_+, w_- are set to corresponding class weights at time *t*, and the forgetting factor θ is set to 0.9. According to [3, 21], an explicit drift detection method is required for OOB to cope with the CI-CD learning problem. However, to compare the *p(y)* and *p(y/x)* drift adaptation between the algorithms, the drift detection methods [2] are not considered in this study. Further, the proposed approach is compared with baseline LASVM, hoeffding adaptive tree (HAT), and leverage bagging (LB) methods. For LB and HAT, hoeffding trees are used as base classifiers and are considered from the MOA [43]. Evaluation prequential of individual class *Recalls* and *GMean* are taken from the MOA. To represent individual class performance, the evaluation prequential *Recall* is presented in graphs. For overall classifier performance, the *average±standard* deviation of *GMean* for each data stream is reported in tables.

Table 2 Datasets description

Dataset	Size	No.of Attributes	% of Minority Class
CIRCLE	1K	2	10%,20%,30%,40%,50%
LINE	1K	2	10%,20%,30%,40%,50%
SEA	1K	3	10%,20%,30%,40%,50%
SINEV	1K	2	10%,20%,30%,40%,50%
SINEH	1K	2	10%,20%,30%,40%,50%
STAGGER	1K	3	10%,20%,30%,40%,50%
AGARWAL	1K	9	10%,20%,30%,40%,50%
COVERTYPE	581K	54	1%
SHUTTLE	58K	9	10%
SMART BUILDING	5K	14	1%
ELECTRICTY	45k	4	40%

5.3 Results and Discussion

Case 1:

Figure 2 presents the impact of class imbalance on WOOLASVM performance using the STAGGER stream by varying the percentages of imbalance ratio ranging from 10% to 50%. Here, the performance is measured in terms of evaluation prequential of minority class *Recall* and *GMean* and compared with the other methods considered. From Figure 2, it is observed that irrespective of the imbalance ratio, the WOOLASVM exhibits better performance compared with other methods. As the percentage of the minority class increases, i.e., from 10-50%, the performance of all the considered algorithms increases significantly from the ground. The WOOLASVM exhibited more than 70% minority class *Recall*. The *GMean* is close to 90%, even at a degree of higher imbalance ratios such as 10%. Similar kinds of results are observed for the remaining datasets. Figure 3 depicts the WOOLASVM's performance on real-world imbalance streams such as SHUTTLE, which is highly imbalanced (10%). In this extreme case of imbalance stream, the WOOLASVM wins over other considered methods. From Figure 3, it is observed that OOB and HAT performance is grounded and not able to rise as the stream progresses with time. The performance of LB is nearly zero at the beginning, and later continuous improvement in the performance is observed. The LASVM families of algorithms are not more sensitive to evolving class imbalance than other methods.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

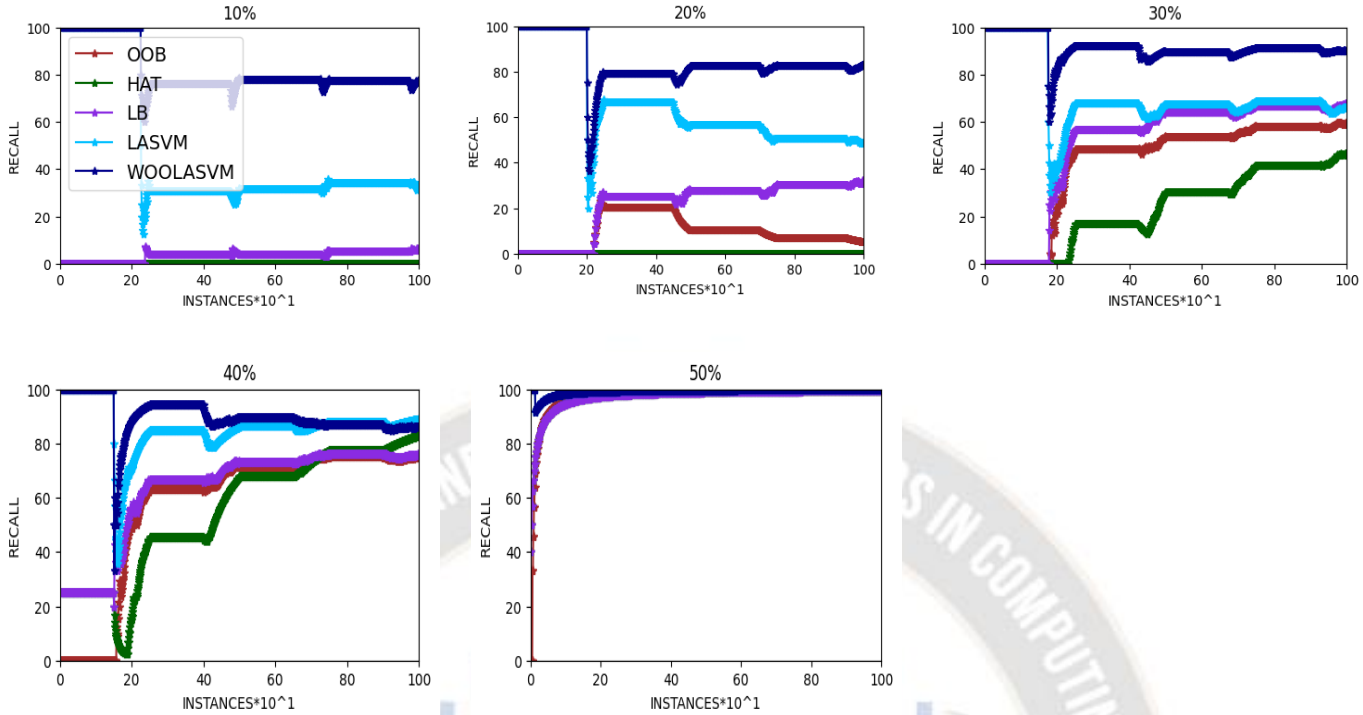

Figure 2 Minority class *Recall* prequential for the stream size 1K on static imbalance streams for STAGGER dataset.

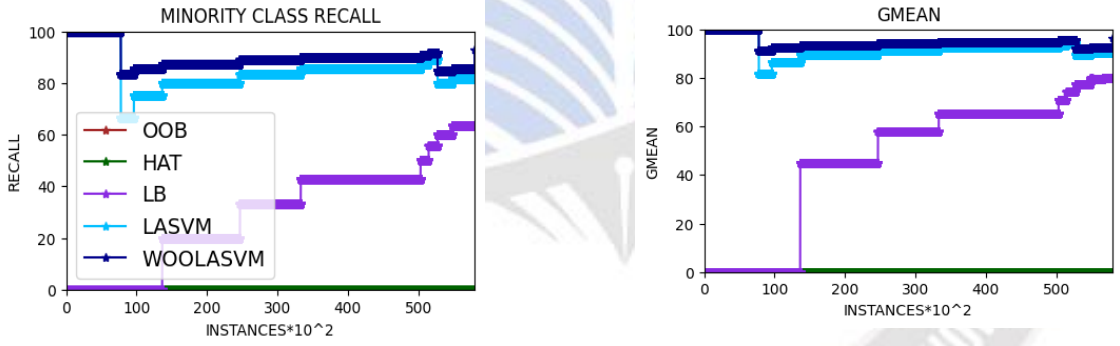

Figure 3 Minority class *Recall* prequential and *GMean* for the stream size 58K for SHUTTLE dataset.

Table 3 shows the comparison of WOOLASVM with other algorithms on various synthetic and real-world streams. The winning performance is boldfaced. The OOB, LB, and HAT underperformed compared on static imbalanced data streams to LASVM, and WOOLASVM. On synthetic streams, at a higher degree of imbalance such as 10% and 20%, and moderate degrees of imbalance such as 30% and 40%, WOOLASVM performance is higher than that of LASVM. In most static imbalanced cases, the WOOLASVM performance is better than the rest. On balanced class percentages, LB better performed than the other algorithms.

As the real-world datasets are concerned, on SHUTTLE and SMART BUILDING, WOOLASVM exhibited superior performance. However, on COVER TYPE, LASVM has been the winner. This might be due to the large stream size, which could be enough to formulate a well-separated boundary. The other two online learners, such as HAT and LB, also exhibited a similar trend of acceptable performances. To rank the considered online learners based on the overall performance of synthetic and real-world data streams, a statistical nemenyi test is carried out. Figure 4 shows the nemenyi post hoc test with a critical distance=1.062. The WOOLASVM achieves the highest performance rank of 4.23, as shown in the figure. However, OOB intended to solve the dynamic class imbalance problem secured the lowest rank of 1.56 and was significantly distinct from WOOLASVM in *GMean* performance.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

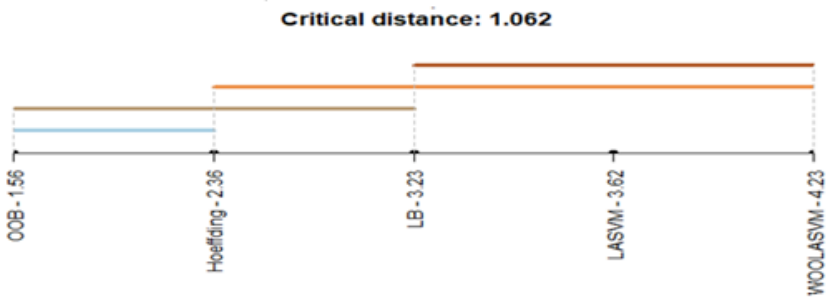

Figure 4 Nemenyi test for static class imbalance streams

Main observations:

1. The WOOLASVM performance is better than other algorithms irrespective of the degree of imbalance on synthetic and real-world static imbalanced data streams.
2. With other algorithms, the performance increases as the minority class percentage increases. The same is observed with LASVM and even for OOB.

Case 2:

Figure 5 shows the WOOLASVM performance in the case of dynamic class imbalance streams (imbalance drift, i.e., *p(y)*). The performance is reported in class-specific *Recalls* such as positive (1), negative (-1), and their corresponding *GMean* for varying percentages of minority class distribution in the evolving stream because both classes change their state of imbalance over time. Here it is assumed that the stream evolves with dynamic change in minority class percentages but is prone to imbalance drift in the middle of the stream. Both abrupt *(a)* and gradual *(g)* speeds of imbalance drifts are considered here.

Compared to the negative class, the positive class performance is more prone to imbalance drift problems (figure 5). As the minority class percentage increases in the distribution, the impact of imbalance drift decreases in terms of a learner's performance improvement. As the minority becomes the majority in imbalance drift cases, there is a sudden increase in performance. This change is significant for a higher degree of imbalance of minority class cases such as [10%, 20%, and 30%]. Concerning the negative class, the impact of dynamic CI is less due to prior good training. As *GMeans* is considered, the impact of imbalance drift is more on a higher degree of imbalance of minority class than moderate and balanced cases. This trend is observed the same for all considered data streams.

Table 3 Average *GMean* for static class imbalance streams

Dataset	(a)	(b)	(c)	(d)	(e)	(f)
CIRCLE	10%	0±0	0±0	0±0	29.8±10.9	**58.1±9.8**
	20%	0±0	30.9±17.4	15.9±9.2	45.9±9.9	**63.7±5.1**
	30%	57.4±2.6	50.6±27.1	43.3±23	**72.2±4.6**	61.3±7.3
	40%	81.7±0.9	73.5±15.2	76.8±7.08	**82.4±2.8**	74.13±3.7
	50%	92.6±10.6	95.4±5.02	93.8±7.4	**96.6±1.8**	85.5±2.6
LINE	10%	0±0	11.2±7.1	3.1±5.1	35.4±15.3	**59.9±7.8**
	20%	1.2±3.8	41.4±22.7	26.6±15.1	50.1±9.6	**70.1±6.7**
	30%	43.5±3.2	60.4±30.6	52.9±27.2	69.7±4.4	**71.5±3.8**
	40%	56.9±0.6	79.2±15.6	76.7±16	**84.6±2.4**	74.9±3.3
	50%	67.9±1.02	97±3.1	96.4±2.9	**96.6±1.8**	94.2±2.9
SEA	10%	0±0	11.9±7.8	11.9±7.8	39.9±33.1	**92.1±5.2**
	20%	0.4±0.04	38.1±19.8	23.3±13.4	41.8±30	**96.3±3.1**
	30%	1.9±0.1	48.7±24	40.3±21	42±28	**96.1±3.3**
	40%	5.5±0.11	70.2±12.5	46.3±11.8	43.9±25.4	**96.8±4.7**
	50%	19.9±0.4	**90.6±4.9**	90.4±3.8	82.3±9.3	78.2±6.9
SINEV	10%	0±0	11.9±7.8	12±7.8	14.7±15.1	**38.9±12.6**
	20%	0±0	39.5±20.7	29.8±16	23.5±13.8	**56.8±7.3**
	30%	13.3±4.2	55.2±28.8	42.3±24.5	35.2±12.3	**62.4±6.6**

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

	40%	34.5±1.1	75.1±13.9	72.3±15.3	50.9±11.7	**77.5±4.5**
	50%	64.3±21.2	**95.5±3**	93.8±4	60.1±16.8	68.9±11.23
STAGGER	10%	10.4±3.5	15.7±8.9	0±0	65.7±18.4	**88.7±5.8**
	20%	26.1±0.9	38.8±20.7	0±0	78.9±11.5	**88.7±6.05**
	30%	56.2±0.9	59.1±28	39.8±23.4	82.1±8.8	**93.8±3.2**
	40%	67.3±0.04	73.8±11.7	70.5±17.1	89.9±5.6	**92.8±4.3**
	50%	97.03±0.7	92.3±5.1	96±4.8	**99.6±0.5**	**99.6±0.5**
AGARWAL	10%	0±0	5.3±8.7	15.8±15.5	75±15.3	**88.±6.7**
	20%	13.1±3.1	28.4±15.3	11.3±6.5	77.7±12.8	**87.9±6.7**
	30%	30.3±2.1	41.5±22	38.1±19.6	73.4±12.9	**91.4±4.4**
	40%	50.8±1.6	59.4±28.2	57.8±27	75.6±10.7	**92.2±4.3**
	50%	80.7±1.8	**85.4±6.3**	82.2±7	71.7±5.1	63.9±5.9
COVER TYPE	1%	26.5±0.3	82.3±5.1	90.6±6.1	**93.7±1.2**	91.9±1.3
SHUTTLE	10%	0±0	46.5±27.5	0±0	91.8±3.9	**94.7±2.3**
SMART BUILDING	1%	0±0	56.2±17.9	16.2±19.3	30.9±11.1	**47.3±15.1**

Note: (a): % of Minority Class (b): OOB (c): LB (d): HAT (e): LASVM (f): WOOLASVM

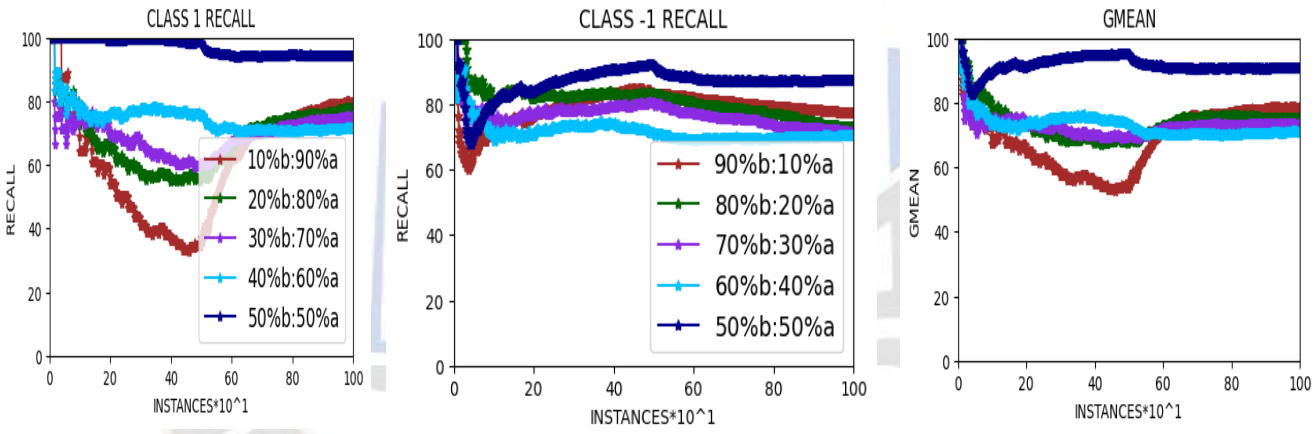

Figure 5 *Recall* prequential and *GMean* for the stream size 1K on dynamic class imbalance streams for LINEa dataset.

Figure 6, 7 shows the WOOLASVM performance on imbalance drift problems compared to considered algorithms on SINEV*a*, *g* streams at a high degree of imbalance (10%). In this scenario, it is observed that WOOLASVM better copes with imbalanced drift compared to the other considered algorithms. From class 1 (minority before the imbalanced drift and majority after) in terms of minority class *Recall*, except for the LASVM family, the rest of the performance of the other algorithms is grounded before the imbalance drift and raised afterward. The same trends are observed for both abrupt and gradual drifts. However, regardless of the change from majority to minority, the influence of the imbalance drift on all algorithms is negligible for class -1. The ability of all algorithms to deal in later stages was aided by sufficient learning of samples relevant to class -1 prior to the change. However, OOB is still observed to be grounded after the imbalance drift. Similarly, the *GMean* performance curve was similar to the class 1 *Recall*.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

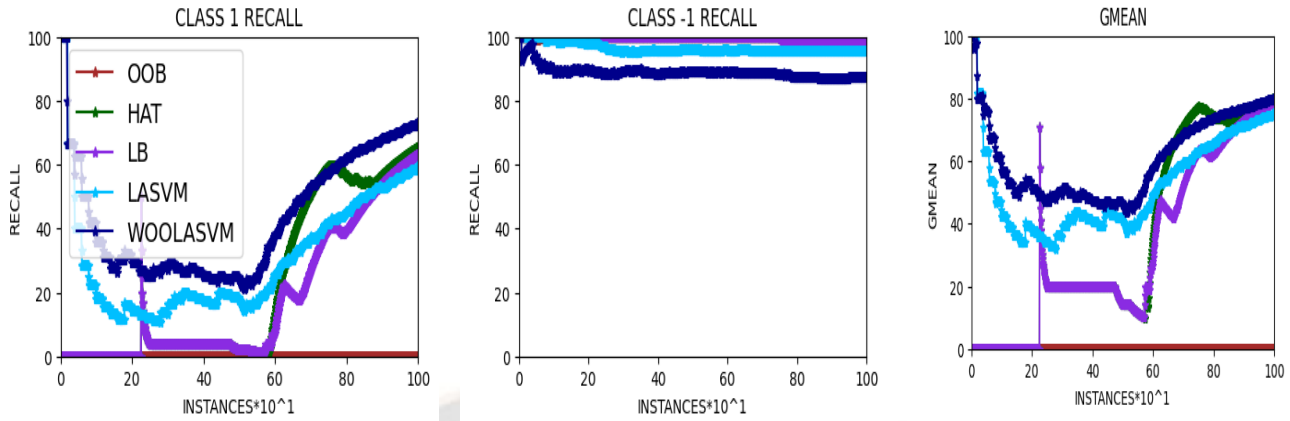

Figure 6 *Recall* prequential and *GMean* for the stream size 1K on dynamic class imbalance streams for SINEVa 10% dataset.

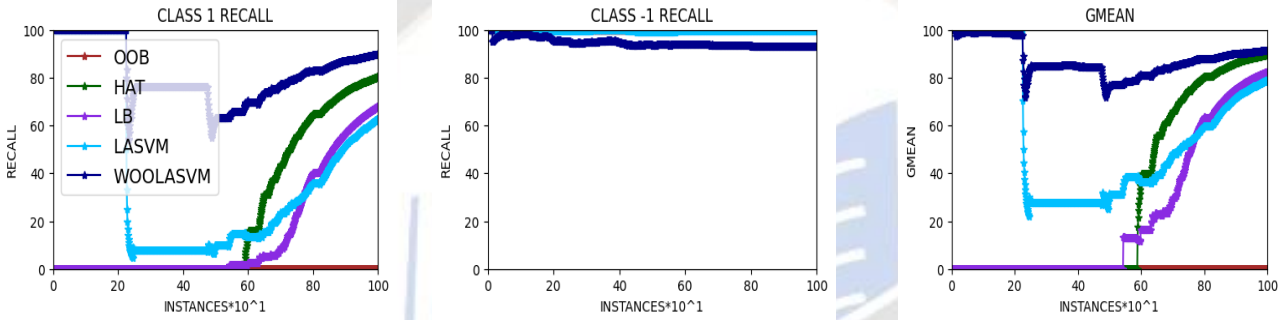

Figure 7 *Recall* prequential and *GMean* for the stream size 1K on dynamic class imbalance streams for SINEVg 10% dataset.

Table 4 shows the findings on seven synthetic streams with the imbalance drift problem. Here the validation is carried out only with synthetic data streams, as it is hard to find the real-world stream in this setting. At higher degrees of imbalance (i.e., 10% and 20%), WOOLASVM performed well compared to other algorithms. The LASVM is the second performer. Similar trends, like SINEV*a, g* are observed for all these considered data streams. To identify the statistical significance of all data streams over the considered algorithm nemenyi test is conducted over *GMeans*. From this test (figure 8), WOOLASVM yielded the best mean rank of 4.45. Here OOB secured a ranking of 1.61 among all, which is intended to solve the imbalance drift problem and is significantly different from WOOLASVM with a critical distance of 1.096, with better results.

Main observations:

1. WOOLASVM handles imbalance drift better than other algorithms.
2. The dynamic imbalance treatment of WOOLASVM can adapt the changes to drift caused by imbalance drift.

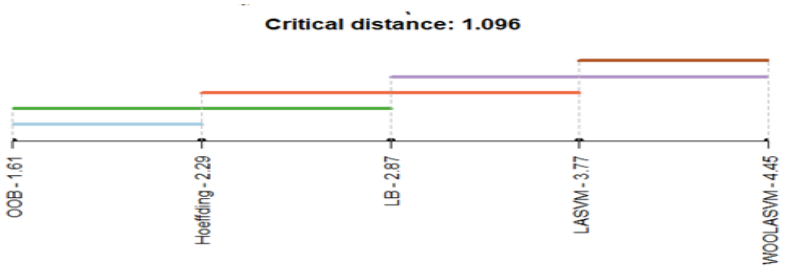

Figure 8 Nemenyi test for dynamic class imbalance streams

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

Table 4 Average *GMean* for dynamic class imbalance streams

Dataset	(a)	(b)	(c)	(d)	(e)	(f)
$CIRCLE_a$	10%	0.7±0.3	27.1±32.9	8.3±9.8	49.4±22.7	**68.1±9.7**
	20%	0.4±0.1	39.1±24.9	19±11.04	57.9±13.9	**66.09±5.2**
	30%	48.7±2.7	50.2±26.7	48.7±7.8	**73.1±4.9**	64.9±7.1
	40%	69.9±0.9	67.8±12	65.5±10.5	**80.8±2.4**	74.4±3.6
	50%	78.6±0.5	84.3±12.1	80.2±14.9	**95.4±1.3**	83.5±2.3
$LINE_a$	10%	0±0	32.7±28.5	29.3±35.6	56.4±18.9	**68.9±9.3**
	20%	1.1±3.4	47.5±26.7	40±18.8	62±11.05	**74.2±5.7**
	30%	40.8±3.03	56.2±28.3	43.1±26.6	69.6±4.4	**72.9±3.5**
	40%	55.4±0.6	74.1±12.2	60.6±8.7	**82.02±2.5**	73.1±4.1
	50%	67.9±1.02	85.7±11.2	85.9±9.1	**94.7±2.03**	91.7±2.4
SEA_a	10%	2.7±0.4	31.2±27	14±9.3	45±30.9	**90.9±5.8**
	20%	20.3±1.1	42.1±23.3	16.6±10.2	50.3±26.1	**91±5.3**
	30%	40.7±0.2	48±23.8	35.7±19.1	56.7±20.7	**93.7±4.2**
	40%	51.9±0.3	65.9±10.6	59.4±11	64.5±16	**93.8±4.4**
	50%	74.9±0.6	**79.6±11.6**	78±13.7	72.6±11.6	72.8±6.18
$SINEV_a$	10%	0±0	32±27.1	35.6±30.7	52.6±15.3	**61.2±13.2**
	20%	7.3±4.2	43.5±23.9	28.5±18	56.4±8.9	**70.6±6.3**
	30%	51.04±1.2	51.9±26.8	36.8±20.8	66.7±6.3	**73.6±4.2**
	40%	58.9±0.6	70.8±11.8	66.6±12.1	**76.7±3.8**	74.4±4.6
	50%	68.5±0.9	**84.6±11.3**	84±9.6	77.5±9.5	77.6±5.6
$STAGGER_a$	10%	13.7±2.6	33±27.3	9.1±11.8	75.2±17.9	**90.07±5.8**
	20%	25.8±1.24	41.9±23.4	1.4±3.6	85.5±8.2	**90.4±4.9**
	30%	49.1±0.8	56.2±26.5	34.3±19.5	82.3±8.8	**93.6±3.3**
	40%	57.7±0.2	70.1±10	60.2±12.4	85.8±7.6	**92.8±4.2**
	50%	82.7±0.7	78.7±12.7	81.8±14.9	96.03±3.6	**96.6±3.04**
$AGARWAL_a$	10%	5.1±0.8	24.4±30.7	7.5±10.1	56.8±25.7	**88.8±8.1**
	20%	20.7±2.6	45.9±26.7	26.6±14.7	57.2±22.3	**92.7±4.4**
	30%	34.6±1.9	51.6±27.2	33.9±16.8	62.6±17.7	**92.7±4.2**
	40%	44.8±1.2	57.8±27.6	48.3±22.5	64.1±15.6	**93.3±4.4**
	50%	71.5±0.92	76.3±8.5	**77.6±6.5**	58.4±7.3	59.8±5.7
$SINEV_g$	10%	0±0	22.3±29.9	29.9±37.4	57.1±27.8	**87.8±6.7**

Note: (a): % of Minority Class (b): OOB (c): LB (d): HAT (e): LASVM (f): WOOLASVM

Case 3:

Figures 9, 10 depict WOOLASVM's performance compared to other algorithms on static CI-CD problem illustrated using a STAGGER*a* stream at a high and balanced degree of imbalanced cases. Here, the stream is assumed to be evolving with static imbalance with concept drift in the middle of the stream. However, it is identified that at a high degree of imbalance cases such as 10%, the impact of the degree of imbalance is high on classifier performance. In more balanced cases, the impact of the concept drift looks vibrant. However, in both cases, WOOLASVM exhibits better performance than other considered methods. A similar trend is observed with eight synthetic and one real-world streams and with gradual and abrupt drifts.

Table 5 shows the average *GMean* performance in this scenario over eight synthetic streams and one real-world stream such as electricity for concept drift validation. On synthetic streams with various percentages of minority class at 10% and 20% cases, WOOLASVM performed well compared to other algorithms. At moderate minority class percentages, every classifier performed equally well. Considering the real-world dataset electricity, LASVM is the winner. The performance of WOOLASVM is next to LASVM. The HAT and LB performed equally well. In a few cases, OOB performed well than LB. Altogether, OOB

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

underperformed compared with other methods. However, though there is a performance drop at drifted positions, the *Process* and *Reprocess* steps through active learning enable the WOOLASVM and LASVM towards early adaption to concept drift. Again, the statistical significance of WOOLASVM concerning other algorithms is studied through the nemenyi post hoc test (see Figure 11). At critical distance CD=0.953, the WOOLASVM secured a 4.29 ranking, significantly different from OOB.

Main observations:

1. Streams at a higher degree of imbalance are prone to a static state of imbalance.
2. The more balance the stream, the impact of *p(y/x)* drift is more.
3. The active learning and imbalance treatment of WOOLASVM leads to early convergence from static CI-CD problems compared to other algorithms.

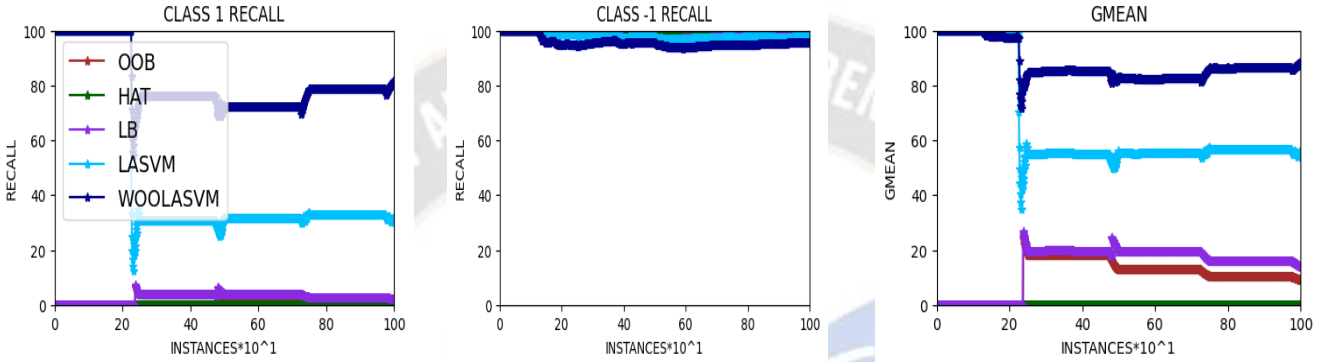

Figure 9 *Recall* prequential and *GMean* for the stream size 1K on static class imbalance with *p(y/x)* drift for STAGGERa 10% dataset.

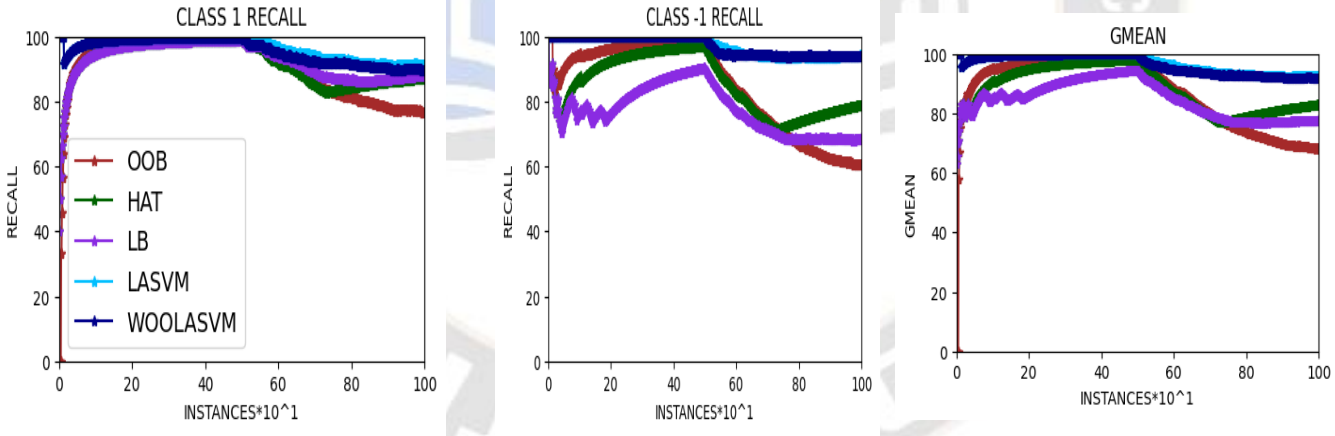

Figure 10 *Recall* prequential and *GMean* for the stream size 1K on static class imbalance with *p(y/x)* drift for STAGGERa 50% dataset.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

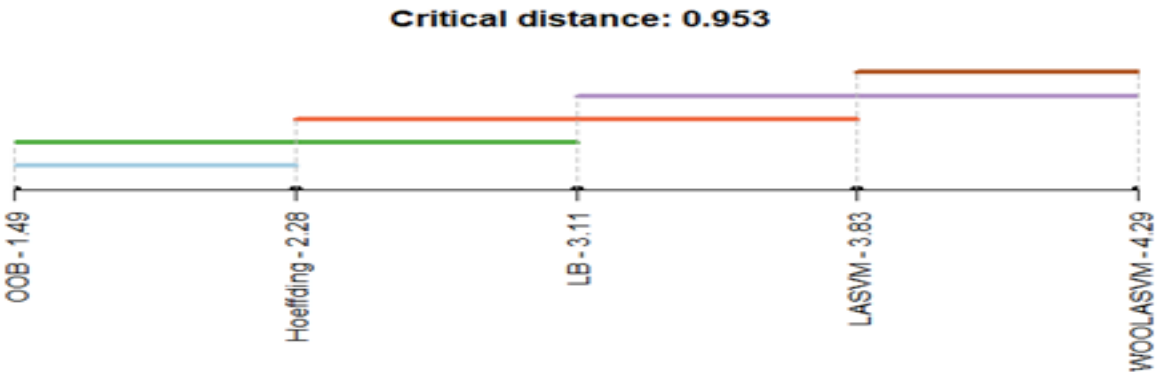

Figure 11 Nemenyi test for static class imbalance streams with *p(y/x)* drift

Case 4:

Figures 12 and 13 demonstrate the *Recall* and *GMean* performance of the WOOLASVM algorithm in coping with dynamic CI-CD problems using the $STAGGER_a$ stream compared to other algorithms at a high and balanced degree imbalanced cases. Both concept and imbalanced drifts are positioned at the same instance (i.e., exactly in the middle of the stream). However, it is identified that at a high degree of imbalance cases such as 10%, the influence of the imbalance drift is high on learner's performance than *p(y/x)* drift. In balanced cases, the impact of *p(y/x)* drift is vital. Compared to other algorithms, WOOLASVM better performed in coping with dynamic CI-CD problem and adapted early towards the *p(y/x)* drift. A similar trend is observed on nine synthetic streams with abrupt imbalance, gradual and abrupt *p(y/x)* drifts (See Table 6).

Table 5 Average *GMean* for static class imbalance streams with *p(y/x)* drift

Dataset	(a)	(b)	(c)	(d)	(e)	(f)
$CIRCLE_a$	10%	0±0	0±0	0±0	30.9±10.8	**56.8±9.6**
	20%	0±0	28.9±16.2	15.9±9.2	45.4±10.2	**63.1±5.3**
	30%	53.1±2.7	48.6±25.6	52.3±6.6	**70.6±4.9**	61.2±7.3
	40%	77±0.95	70.7±13.5	71.6±7.4	**80.7±2.3**	72.5±3.7
	50%	87.9±0.5	91.7±5.6	89.7±6.6	**94.7±1.5**	82.8±2.7
$LINE_a$	10%	0±0	33.1±18.7	25.3±15.5	21.9±13.2	**60.5±8.3**
	20%	22.7±4.8	54.2±29	43.6±23.3	56.8±11.2	**66.9±6.2**
	30%	52.4±2.1	63.1±30.5	52.5±27.8	**70.7±5.03**	69.1±6.2
	40%	60.9±0.5	79.3±13.6	78±14.5	**83.9±1.8**	78.2±2.6
	50%	69.2±0.9	92.6±5	91.3±5.7	**96.4±0.8**	94.4±1.3
SEA_a	10%	0±0	11.1±7	0±0	40.3±32.6	**90.8± 5.3**
	20%	20.2±1.9	31.5±16.9	20.2±12	50.5±25.1	**95.4± 2.8**
	30%	45.1±0.9	50.1±24.6	45.2±22.9	58±20.21	**95.5±2.9**
	40%	61.8±0.7	71.8±11	70.3±11.4	73.5±13.4	**95.9±2.6**
	50%	88.8±0.5	**90.6±1.6**	90.3±2.3	84.3±4.9	79.9±3.7
$SINEV_a$	10%	0±0	26.4±15.2	15.8±10	17.4±14.3	**29.3±13.1**
	20%	1.4±1.1	**54.7±29.4**	36±20.2	28.8±11.7	49.8±7.2
	30%	34.9±1.7	**63.5±31.6**	52.9±27.7	41.1±9.6	58.9±6.3
	40%	42.4±0.8	**78.6±15.8**	15.8±10	59.6±8.1	67.7±6.2
	50%	65.3±20.8	**88.9±5.3**	86.7±6.3	70.03±9.1	76.9±4.9
$STAGGER_a$	10%	10.4±3.5	14.01±8	0±0	65.3±18.6	**87.8±6.4**
	20%	26.1±0.9	35.2±19.1	0.08±0.9	82.2±9.9	**92.4±3.9**
	30%	50.9±0.8	53.8±25.6	33.8±19.2	81.5±9.03	**93.9±3.2**

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

	40%	59.5±0.2	67.9±9.6	66.4±14.1	87.5±6.4	**93.2±4.1**
	50%	86.9±0.8	84.5±6.2	88.2±7.5	**96.7±3**	96.4±3.3
AGARWAL$_a$	10%	0±0	0±0	11.5±7.1	46.5±29.1	**91.7±5.2**
	20%	0±0	10.1±5.5	8.3±5	53.2±23.8	**92.9±3.9**
	30%	16.7±0.9	23.3±12.3	11.6±6.5	57.8±19.4	**94.9±2.9**
	40%	46.9±0.8	49.9±22.8	47.4±21.9	63.8±15.9	**94.6±3.6**
	50%	**78.2±0.83**	**78.2±11.2**	75.8±7.9	61.5±6.4	62.5±4.9
SINEH$_g$	10%	0±0	0±0	0±0	56.2±23.8	**89.4±6.1**
	20%	11±1.7	73.3±13.7	3.7±6.3	64.3±18	**89.5±5.9**
	30%	29.2±1.1	47.4±25.1	34.3±18.4	71.5±13.8	**91.5±4.9**
	40%	37.6±1.7	59.9±28.2	62.3±12.9	79.1±9.7	**94.4±4.2**
	50%	63.2±4.9	81.1±2.3	80.6±2.2	80.4±11.7	**82.8±5.9**
LINE$_g$	10%	22.3±2.6	33±18.7	25.3±15.5	78.1±12.3	**81.8±8.4**
	20%	36.2±0.5	52.7±28.5	42.3±23.2	78.8±10.7	**93.7±3.5**
	30%	26.5±6.9	63.4±30.6	53.3±28.1	**94.03±4.1**	90.5±3.1
	40%	35.03±1.4	80.2±13.8	79±14.8	**90.7±4.7**	90.1±2.2
	50%	69.6±0.9	94.5±4	93.2±5	**96.6±1.8**	95.5±2.5
Electricity	40%	38.6±0.6	65.9±3.8	65.5±2.4	**71.4±3.3**	67.7±4.3

Note: (a): % of Minority Class (b): OOB (c): LB (d): HAT (e): LASVM (f): WOOLASVM

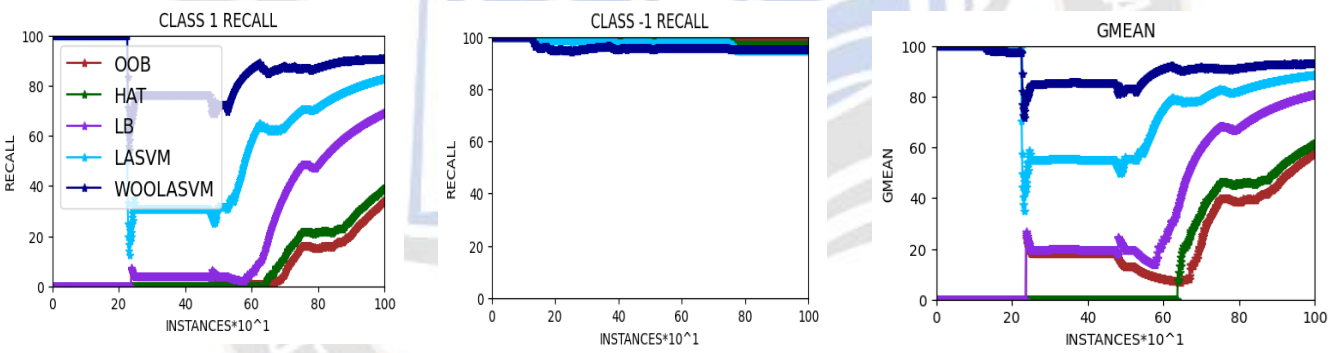

Figure 12 *Recall* prequential and *GMean* for the stream size 1K on dynamic class imbalance streams (i.e., *p(y)* with *p(y/x)* drift) for STAGGERa 10% dataset.

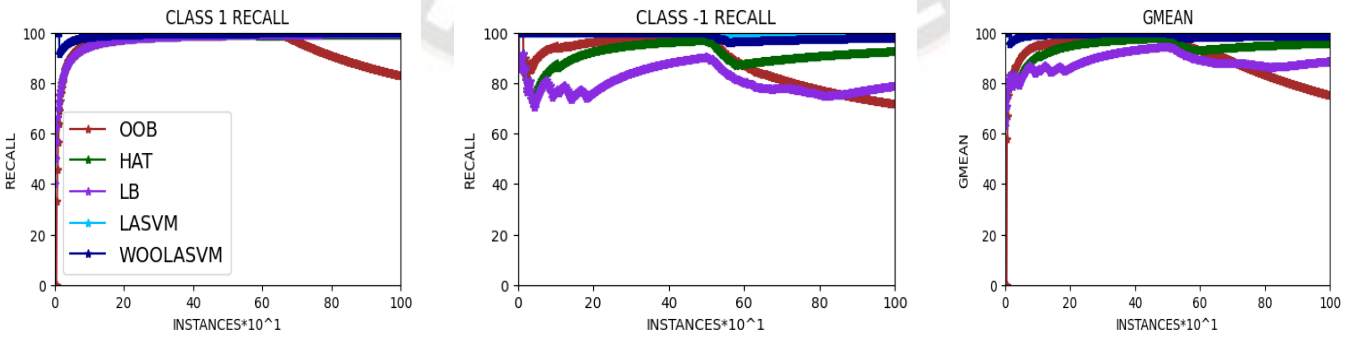

Figure 13 *Recall* prequential and *GMean* for the stream size 1K on dynamic class imbalance streams (i.e., *p(y)* with *p(y/x)* drift) for STAGGERa 50% dataset.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

From figure 14, it is clear that WOOLASVM is the winner with the 4.33 best ranking in the nemenyi post hoc test. However, the proposed WOOLASVM differs greatly from OOB in critical distance=0.964.

Main Observations:

1. On an imbalanced evolving stream, imbalanced drift has a significant impact.
2. On balanced evolving streams, concept drift has a significant impact.
3. Due to dynamic sampling and weighing, active learning WOOLASVM better copes with dynamic CI-CD problems than other methods.

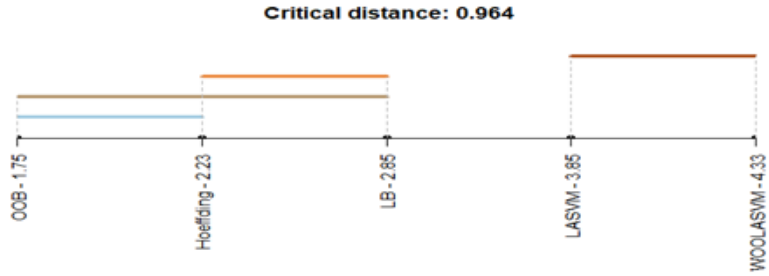

Figure 14 Nemenyi test for on dynamic class imbalance streams (i.e., *p(y)* with *p(y/x)* drift)

6 Conclusion

This paper introduces the SVM active learning-based weighted online learner WOOLASVM to address the combined problem of CI-CD. The main focus is dealing with the imbalance drift part of concept drift. Throughout the stream learning process, the WOOLASVM algorithm is proposed to maintain a balanced boundary. Here the balanced boundary is assigned in two folds.

1. The initial boundary is balanced by creating artificial samples that are half the pathway towards minority class samples between minority and majority class SVs.
2. The next formulating boundaries are again balanced in two ways. At first, asymmetrically weighing the cost functions based on the weights of the current data. This is to push the border towards the majority class to create a better space for minority class prediction. Further balanced boundary formulation, the minority class SVs are oversampled dynamically. Here the imbalance drift is handled by adjusting the boundary based on the weight difference between the minority to the majority or vice versa for dynamic oversampling between the SVs.

The experimental results were demonstrated on synthetic and real-world data streams in four different imbalance and concept drift settings.

The main observations are:

1. At a higher static degree of imbalance, the WOOLASVM converges to acceptable performance compared to the base LASVM active learning, LB, HAT.
2. Proposed WOOLASVM exhibited better *Recall* and *GMean* performance in coping with imbalance drift (*p(y)*) than OOB.
3. In a static state of imbalance, the WOOLASVM outperformed the LB and HAT in dealing with *p(y/x)* drift.
4. The WOOLASVM handled static and dynamic imbalance better than LB, HAT, and OOB with *p(y/x)* drift.

Table 6 Average *GMean* for dynamic class imbalance streams (i.e., *p(y)* with *p(y/x)* drift)

Dataset	(a)	(b)	(c)	(d)	(e)	(f)
$CIRCLE_a$	10%	11.6±0.6	23.5±32.3	9.02±14.9	53.6±25.9	**68.1±9.9**
	20%	7.3±0.5	37.8±24.6	18±12.2	58.9±13.9	**66.8±5.5**
	30%	49.7±2.5	49.7±26.4	49.4±7.4	**72.1±4.6**	64.1±6.9
	40%	32.9±0.9	67.9±12.4	68.1±8.9	**80.5±2.3**	74.6±3.7
	50%	81.6±0.6	87.2±9.3	82.5±12.9	**94.6±1.5**	84.2±2.3
$LINE_a$	10%	8.2±13.8	44.1±28.7	32.8±21.8	49.1±28.5	**71.8±8.5**
	20%	21.2±10.6	54±28.9	42.8±24.1	67.1±10.7	**71.9±5.8**
	30%	49.2±16.9	61.5±29.7	50.3±26.8	**72.7±5.4**	71.4±5.8
	40%	59.4±18.7	77.7±13.2	75.5±13.9	**83.7±1.7**	78.1±2.6
	50%	74.3±14.3	91.6±5.6	93.4±3.5	**96±0.8**	94.9±0.9

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

SEA_a	10%	9.5±0.7	31.8±28.5	9.02±12.2	48.1±29.1	**93.4±4.2**
	20%	21.7±1.8	38.4±22.9	23.3±13.2	54.3±23.6	**95.5±2.71**
	30%	39.8±0.95	50.8±25.2	39.6±20.7	57.9±20.2	**94.9±3.3**
	40%	51.9±0.6	68.8±9.4	60.6±11	69.6±13.8	**95.03±3.2**
	50%	76.1±0.5	**79.7±12.4**	78.1±14.2	75.1±9.6	75.1±5.5
$SINEV_a$	10%	0±0	37±26.6	27.4±23.4	47.9±13.3	**63.3±14.5**
	20%	7.3±5.6	56±30	38.4±22.1	55.3±8.8	**71.5±6.5**
	30%	49.8±1.2	62±30.9	58.5±30.9	63.6±6.1	**73.3±3.9**
	40%	58.2±0.6	**77.3±15.3**	74.6±15.2	75.5±3.9	73.3±4.1
	50%	68.5±0.9	88.1±5.6	**90.3±5.8**	76.8±9.2	76.7±5.4
$STAGGER_a$	10%	19.6±2.4	32.9±28.8	16.2±22.6	76.7±17.5	**90.8±5.6**
	20%	34.9±0.9	43.3±24.7	17.3±23.9	85.6±8.14	**93.5±3.9**
	30%	54.7±0.9	57.9±27.3	41.4±25	82.9±8.7	**94.5±3.1**
	40%	59.1±0.13	73.1±11.4	69.8±16.5	89.08±5.8	**93.7±3.9**
	50%	90.5±0.9	88.3±3.6	94±4.1	**99.4±0.5**	98.9±0.7
$AGARWAL_a$	10%	4.2±1.8	22.4±27.2	4.6±10	55.9±25.4	**95.04±3.7**
	20%	14.5±1.6	34.4±27.3	22±15	58±22.17	**94.8±3.11**
	30%	24.7±1.8	36.4±27.8	22±17.3	62.1±18.4	**95.9±2.3**
	40%	48.4±0.93	55.6±27.6	47.1±23.6	67.9±14.2	**95.2±3.5**
	50%	76.4±0.8	**76.5±11.3**	75.1±8	59.8±6.3	62.6±4.9
$SINEH_g$	10%	5.7±0.3	24.9±32.3	7.8±13.1	62.9±21.5	**93.3±4.7**
	20%	18.3±1.4	34.4±25.4	13.4±18.1	65.3±17.6	**91.7±4.7**
	30%	27.5±2.2	47.8±25.3	37.5±21	69.7±14.3	**91.9±4.6**
	40%	36.2±1.6	57±26.7	59.6±10.9	76.3±10.5	**94.1±4.3**
	50%	63.2±4.9	77.3±5	75.8±6.3	75.3±9.7	**79.4±4.8**
$LINE_g$	10%	20.5±2.1	44.1±28.8	24.4±17	84.9±9.7	**85.5±6.6**
	20%	30.9±0.4	55±29.4	40.6±23	82.5±9.8	**93.1±3.4**
	30%	24.4±6.4	61.3±29.7	52.4±27.7	**94.1±3.9**	90.9±2.8
	40%	33.9±1.4	78.2±13.4	77±14.2	**90.7±4.7**	90.1±2.1
	50%	69.5±0.9	93±5.1	94.1±3.4	**96.5±1.9**	95.3±2.6

Note: (a): % of Minority Class (b): OOB (c): LB (d): HAT (e): LASVM (f): WOOLASVM

Acknowledgments

India's defense research and development organization (DRDO) sponsored this research under the sanction code: ERIPR/GIA/17-18/038. Center for artificial intelligence and robotics (CAIR) served as the work's reviewing lab. We want to acknowledge the support and help of the late Dr. T. Maruthi Padmaja in this work, and she is the grant recipient.

References

[1]. He. Haibo, Edwardo A. Garcia, Learning from imbalanced data, IEEE Transactions on Knowledge and Data Engineering, 21(9) (2009) 1263–1284.

[2]. J. Gama, I. Zliobait˙e, A. Bifet, M. Pechenizkiy, A. Bouchachia, A survey on concept drift adaptation, ACM Computing Surveys (CSUR), 46 (2014) 1–37.

[3]. S. Wang, L. L. Minku, X. Yao, A Systematic Study of Online Class Imbalance Learning With Concept Drift, IEEE Transactions on Neural Networks and Learning Systems, 29 (10) (2018) 4802–4821

[4]. N. Japkowicz, S. Stephen, The class imbalance problem: A systematic study, Intelligent Data Analysis, 6(5) (2002) 429–449

[5]. A. Ghazikhani, R. Monsefi, H. S. Yazdi, Recursive least square perceptron model for non-stationary and imbalanced data stream classification, Evolving Systems, 4 (2013) 119–131.

[6]. A. Ghazikhani, R. Monsefi, H. S. Yazdi, Online neural network model for non-stationary and imbalanced data stream classification, International Journal of Machine Learning and Cybernetics, 2 (2014) 51-62.

[7]. B. Mirza, Z. Lin, K.A. Toh, Weighted Online Sequential Extreme Learning Machine for Class Imbalance Learning, Neural Process Lett, 38 (2013) 465–486.

[8]. S. Wang, L. L. Minku, X. Yao, Resampling-Based Ensemble Methods for Online Class Imbalance Learning,

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

IEEE Transactions on Knowledge and Data Engineering, 27 (5) (2015) 1356–1368.

[9]. B. Mirza, Z. Lin, N. Liu, Ensemble of subset online sequential extreme learning machine for class imbalance and concept drift, Neurocomputing, 149 (Part A) (2015) 316–329.

[10]. Y. Tang, Y. Q. Zhang, N. V. Chawla, S. Krasser, SVMs Modeling for Highly Imbalanced Classification, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 39 (1) (2009) (281-288).

[11]. Chaudhary, D. S. . (2022). Analysis of Concept of Big Data Process, Strategies, Adoption and Implementation. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 8(1), 05–08. https://doi.org/10.17762/ijfrcsce.v8i1.2065

[12]. J. Kremer, K. Steenstrup, C. P. Igel, Active learning with support vector machines, Wires data mining and knowledge discovery, (2014).

[13]. SULEIMAN, S. M., SHUAIBU, D. S., & BABALE, S. A. (2022). Investigating the effect of High Altitude Platform Positioning on Latency and Coverage of 4G Cellular Systems. International Journal of Communication Networks and Information Security (IJCNIS), 14(1).

[14]. G. Ditzler, M. Roveri, C. Alippi, R. Polikar, Learning in Nonstationary Environments: A Survey, IEEE Computational Intelligence Magazine, 10 (4) (2015) 12-25.

[15]. J. Liao, J. Zhang, W. Y. Ng. Wing, Effects of different base classifiers to Learn++ family algorithms for concept drifting and imbalanced pattern classification problems, 2016 International Conference on Machine Learning and Cybernetics (ICMLC), 1 (2016) 99-104.

[16]. 2009 Sheng Chen and He. Haibo, SERA: Selectively recursive approach towards nonstationary imbalanced stream data mining, International Joint Conference on Neural Networks, (2009) 522-529.

[17]. S. Chen, He. Haibo, Towards incremental learning of nonstationary imbalanced data stream: a multiple selectively recursive approach, Evolving Systems, (2) (2011) 35-50.

[18]. T. R. Hoens, N. V. Chawla, R. Polikar, Heuristic Updatable Weighted Random Subspaces for Non-stationary Environments, 2011 IEEE 11th International Conference on Data Mining, (2011) 241-250.

[19]. T. R. Hoens, N. V. Chawla, Learning in non-stationary environments with class imbalance, 18th ACM SIGKDD international conference on knowledge discovery and datamining, (2012) 168-176.

[20]. Y. Wang, Y. Zhang, Y. Wang, Mining Data Streams with Skewed Distribution by Static Classifier Ensemble,Opportunities and Challenges for Next-Generation Applied Intelligence, Chien, Been-Chian and Hong, Tzung-Pei, Springer Berlin Heidelberg, Berlin, Heidelberg, (2009), 65–71.

[21]. L. Loezer, F. Enembreck, J. P. Barddal, A. S. Britto, Cost-sensitive learning for imbalanced data streams, Proceedings of the 35th Annual ACM Symposium on Applied Computing,(2020)

[22]. S. Barua, M. M. Islam, K. Murase, GOS-IL: A Generalized Over-Sampling Based Online Imbalanced Learning Framework, ICONIP, (2015).

[23]. N. A. Libre. (2021). A Discussion Platform for Enhancing Students Interaction in the Online Education. Journal of Online Engineering Education, 12(2), 07–12. Retrieved from http://onlineengineeringeducation.com/index.php/joee/article/view/49

[24]. S. Wang, L. L.Minku, D. Ghezzi, D. Caltabiana, P. Tino, X. Yao, Concept drift detection for online class imbalance learning,The 2013 International Joint Conference on Neural Networks (IJCNN), (2013) 1–10.

[25]. Tutuncu, K., & Hacimurtazaoglu, M. (2021). KBM Based Variable Size DCT Block Approaches for Video Steganography: KBM Based Variable Size DCT Block Approaches for Video Steganography. International Journal of Intelligent Systems and Applications in Engineering, 9(4), 220–231. https://doi.org/10.18201/ijisae.2021473643

[26]. J. Gama, P. Medas, G. Castillo, P. Rodrigues, Learning with Drift Detection, Advances in Artificial Intelligence – SBIA 2004, Bazzan, Ana L. C. and Labidi, Sofiane, Springer Berlin Heidelberg, Berlin, Heidelberg, (2004) 286–295.

[27]. H. Wang, A. Zubin, Concept drift detection for streaming data, 2015 International Joint Conference on Neural Networks (IJCNN), (2015) 1–9.

[28]. D. Brzezinski, J. Stefanowsk, Properties of the area under the ROC curve for data streams with Concept drift, Knowledge and information systems, 52 (2017) 531–562.

[29]. Yu. Shujian, Z. Abraham, H. Wang, S. Mohak, J.C. Prinicipe, Concept drift detection and adaptation with hierarchical hypothesis testing, Journal of the franklin institute, 356 (5) (2019) 3187–3215.

[30]. S. Wang, L. L. Minku, AUC Estimation and Concept Drift Detection for Imbalanced Data Streams with Multiple Classes, 2020 International Joint Conference on Neural Networks (IJCNN), (2020) 1–8.

[31]. M. M Idrees, L. L. Minku, F. Stahl, A. Badii, A heterogeneous online learning ensemble for non-stationary environments, Knowledge-Based Systems, 188 (2020) 104983.

[32]. S. Micevska, A. Awad, S. Sakr, SDDM: An interpretable statistical concept drift detection method for data streams, Journal of intelligent information systems, 56 (2021) 459–484.

[33]. Li. Peipei, W. Man, H. Jun, H. Xuegang, Recurring Drift Detection and Model Selection-Based Ensemble Classification for Data Streams with Unlabeled Data, New Generation Computing, 39 (2021) (341-376)

[34]. Alabdulrazzaq, H., & Alenezi, M. N. (2022). Performance Evaluation of Cryptographic Algorithms: DES, 3DES, Blowfish, Twofish, and Threefish. International Journal of Communication Networks and Information Security (IJCNIS), 14(1).

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 8
DOI: https://doi.org/10.17762/ijritcc.v10i8.5665
***Article Received**: 10 June 2022 **Revised**: 15 July 2022 **Accepted**: 30 July 2022 **Publication**: 31 August 2022*

[35]. G. Ditzler, R. Polikar, Incremental Learning of Concept Drift from Streaming Imbalanced Data, IEEE Transactions on Knowledge and Data Engineering, 25 (10), (2013) 2283-2301.

[36]. R. Akbani, S. Kwek, N. Japkowicz, Applying Support Vector Machines to Imbalanced Data Sets, ECML 2004,15th European confernece on machine learning, (2004) 39–50.

[37]. I. Zliobaite, A. Bifet, B. Phfranger, G. Holmes, Active Learning With Drifting Streaming Data, IEEE Transactions on Neural Networks and Learning Systems, 25 (1) (2014) 27–39.

[38]. S. Tong, D. Koller, Support vector machine active learning with applications to text classification, The Journal of Machine Learning Research, 2 (2005) 45–66.

[39]. S. Ertekin, J. Huang, L. Bottou, C. L. Giles, Learning on the border: Active learning in imbalanced data classification, International conference on information and knowledge management proceedings, (2007) 127-136.

[40]. C. C. Chang, C. J. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology, 2 (3) (2011) 1–27.

[41]. T. Joachims, SVM Light: Support Vector Machine, (2002).

[42]. A. Bordes, S. Ertekin, J. Weston, L. Bottou, Fast Kernel Classifiers with Online and Active Learning, Journal of Machine Learning Research, 6 (2005) 1579-1619.

[43]. C. Cortes, V. Vapnik, Support-vector networks, Machine Learning, (1995) 273-297.

[44]. J. Platt, Sequential Minimal Optimization : A Fast Algorithm for Training Support Vector Machines, Microsoft Research Technical Report, (1998).

[45]. Demirci, I., & Saritas, I. (2021). Damage Detection of Carbon Face Sheet Nomex Sandwich Composites with Image Processing Technique. International Journal of Intelligent Systems and Applications in Engineering, 9(4), 282–287. https://doi.org/10.18201/ijisae.2021474023

[46]. K. Morik, P. Brockhausen, T. Joachims, Combining Statistical Learning with a Knowledge-Based Approach - A Case Study in Intensive Care Monitoring, ICML, (1999).

[47]. W. Gang, E.Y.Chang, Class-Boundary Alignment for Imbalanced Dataset Learning, (2003).

[48]. N. V. chawla, K. W. Bowyer, L. O. Hall, SMOTE: synthetic minority over-sampling technique, Journal of Artificial Intelligence Research, 16 (1) (2002) 321-357.

[49]. A. Bifet, G. Holmes, R. Kirkby, MOA: Massive Online Analysis, Journal of Machine Learning Research, 11 (2010) 1601-1604.

[50]. L. L. Minku, A. P. White, X. Yao, The Impact of Diversity on Online Ensemble Learning in the Presence of Concept Drift, IEEE Transactions on Knowledge and Data Engineering, 22 (5) (2010) 730-742.

[51]. J. Gama, R. Sebastiao, P. P. Rodrigues, On evaluating stream learning algorithms, Machine Learning, (90) (2012) 317-346