International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

Skewed Evolving Data Streams Classification with Actionable Knowledge Extraction using Data Approximation and Adaptive Classification Framework

Rahul A Patil[1], Dr Pramod D Patil[2]
[1]Research Scholar, Dr.D.Y.Patil Institute of Technology, Pimpri, Pune-411018 &
Assist. Prof. Pimpri Chinchwad College of Engineering, Pune-411044, India
patil.rahul3068@gmail.com
[2]Professor, Computer Engg. Dr. D. Y. Patil Institute of Institute of Technology, Pimpri, Pune-411018, India
pdpatiljune@gmail.com

Abstract—Skewed evolving data stream (SEDS) classification is a challenging research problem for online streaming data applications. The fundamental challenges in streaming data classification are class imbalance and concept drift. However, recently, either independently or together, the two topics have received enough attention; the data redundancy while performing stream data mining and classification remains unexplored. Moreover, the existing solutions for the classification of SEDSs have focused on solving concept drift and/or class imbalance problems using the sliding window mechanism, which leads to higher computational complexity and data redundancy problems. To end this, we propose a novel Adaptive Data Stream Classification (ADSC) framework for solving the concept drift, class imbalance, and data redundancy problems with higher computational and classification efficiency. Data approximation, adaptive clustering, classification, and actionable knowledge extraction are the major phases of ADSC. For the purpose of approximating unique items in the data stream with data pre-processing during the data approximation phase, we develop the Flajolet Martin (FM) algorithm. The periodically approximated tuples are grouped into distinct classes using an adaptive clustering algorithm to address the problem of concept drift and class imbalance. In the classification phase, the supervised classifiers are employed to classify the unknown incoming data streams into either of the classes discovered by the adaptive clustering algorithm. We then extract the actionable knowledge using classified skewed evolved data stream information for the end user decision-making process. The ADSC framework is empirically assessed utilizing two streaming datasets regarding classification and computing efficiency factors. The experimental results shows the better efficiency of the proposed ADSC framework as compared with existing classification methods.

Keywords-Adaptive classification, actionable knowledge extraction, adaptive clustering, data approximation, concept drift, class imbalance, skewed evolving data stream.

I. INTRODUCTION

Data stream categorization [1]-[3] is still one of the most vital data mining strategies. It entails utilizing the design built from chronological data by learning to construct a forecast regarding the fresh observations/data class label. Formally, it is a learning assignment that maps all attribute sets to a set of predetermined class labels [4]. In recent years, the research community has focused on the challenge of data stream categorization. One of the most distinguishing features of data stream mining is that classification is a continuous process; therefore, the quantity of training data may be deemed endless [5]. As a result, storing all the examples of training the classifiers is nearly impossible. To solve this issue, specific incremental learning strategies are presented. Classification of stream data has been actively investigated in recent years, with many intriguing methods designed since it may aid decision-making by anticipating class labels for given data based on prior records. Most stream mining experiments are generally balanced, resulting in consistent data streams. Many applications, however, may include concept-drifting data streams with skewed distributions (SDs). Each data chunk in SDs has numerous fewer good examples. Simultaneously, the loss functions associated with the positive and negative classes are imbalanced. Positive cases misclassified can have severe consequences for particular applications, such as a series of financial transactions. Data redundancy, in addition to concept drift and class imbalance, is difficult for SEDS categorization.

Concept drift, which refers to goal conceptions of streams shifting over time, is a prevalent aspect of data streams [6]-[8]. Since the model trained on old notions may not be sufficient for

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

contemporary concepts, concept drift might degrade classification performance. For example, consumer behavior may impact fashion trends in suggest systems, and the weather forecast model may no longer be useful when the season changes. As a result, an effective data stream learning model should be capable of capturing drifts quickly and updating the model accordingly [9]. Several approaches for coping with concept drift have been developed [10]. The window-based approach among these uses a natural forgetting mechanism to add new instances and erase outmoded ones. The most common window technology is the sliding windowIt shifts processed instances using the first-in-first-out structure, ensuring that the most recent instances are displayed in the current window. Ensemble algorithms are the most often used approaches for dealing with concept drift because they are modular and can swiftly adjust to changes.

Although significant research has been conducted on concept drift, the class imbalance problem [11] makes tackling concept drift even more challenging. In the actual world, class imbalances are widespread. Financial fraud detection, geological disaster prediction, and cancer diagnosis are a few examples. In binary classification, the majority class is the one with the most occurrences, while the minority class is the one with the fewest. For example, in the online fraud identification of automotive insurance, dishonest clients accounted for barely 1% of the total customers in 100,000 incidents. Finding a method to reliably identify just 1% of fraudulent instances will greatly reduce economic damage. Several common approaches to dealing with class imbalance [12-15] have been proposed. The existing class imbalance solutions are categorized into data-level techniques, cost-sensitive learning, and ensemble methods. Cost-sensitive learning strategies seek to reduce total costs. According to some experts, the cost-sensitive approach is the most successful and widely used tool for dealing with class imbalance.

Data approximation and preprocessing continuously streaming data is yet another challenge for skewed evolving streaming data classification [16]. There is a scarcity of efficient approaches for performing data approximation. On the internet, more than 2.5 quintillion bytes of data are generated each day. To evaluate this data, it must first be collected, stored in a secure location, cleaned, and analyzed. Big data developers face one of the biggest challenges when dealing with worthless or redundant data [16]. Collecting and evaluating this extra data takes significant time and finances, yet it is all for naught. Therefore, removing duplicate data is essential for reducing the cost and frequency of analysis. Data cleaning may be accomplished in several methods [17-20], but first, you must establish how much useable data is included in the dataset. As a result, it's critical to decide unique data before deleting duplicate data from an evolving data stream. Aside from that, noisy data in online streaming data might lead to erroneous knowledge discovery and decision-making. As a result, adequate methods are necessary to address approximation, concept drift, and class imbalance challenges before performing the classification of skewed evolving streaming for knowledge discovery and decision making.

To address these issues, we propose a Adaptive Data Stream Classification (ADSC) framework to perform the SEDS classification using timestamp-based techniques for the data approximation, data clustering, supervised classification, and actionable knowledge extraction. To the best of our knowledge, ADSC is the first integrated mechanism that addresses the various challenges while performing the data stream classification. The challenges are mainly related to two vital factors (cost efficiency and classification accuracy) of real-time data streaming mining and classification. The data approximation technique in ADSC addresses cost savings by reducing unnecessary data while live data broadcasting. The data approximation mechanism significantly reduces the computational requirements (time and storage) to process the skewed streaming data. We proposed a novel Flajolet Martin (FM) algorithm that periodically approximates the received streaming data with minimum computational requirements for accurate data. After that, we applied the proposed adaptive clustering algorithm to the group and assigned the new classes to approximated data streams. The adaptive clustering mechanism addresses the concept drift and solves the class imbalance problem by distributing data stream samples equally across the different groups. The adaptive clusters are further utilized as the training data features to classify the new incoming data stream using the supervised classifier. The actionable knowledge is extracted before publishing the classified data streams for appropriate decision-making. Section 2 presents the review of different techniques for data stream mining and classification. Section 3 presents the design and methodology of the ADSC framework. Section 4 presents the simulation results and discussions. Section 5 offers conclusive remarks on the work done.

II. RELATED WORKS

Several studies have been published on static dataset classification; however, skewed changing data stream classification is a complex research challenge. Various strategies for data stream categorization were designed during the previous decade to resolve class imbalance, idea drift, or both concerns. We evaluated previously suggested methods for streaming data categorization, followed by analyzing research gaps and this paper's contributions.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

A. State-of-the-arts

The authors of [21] provide a unique ensemble method termed the Recursive Ensemble Approach (REA) for dealing with class imbalance difficulties in a nonstationary setting. REA used the K-nearest neighbor (KNN) method to compare the similarity of the previous block's minority class instances to the current block's minority class instances and then picked the previous minority examples to balance the classes in the current block. In [22], the Learn++ framework had proposed to cope with the class imbalance in a data stream context. The authors of [23] created an online version of Extreme Learning Machine to address the issue of class imbalance. To deal with the class imbalance problem, [24] developed a unique neural network framework based on a cost-sensitive technique. To address class imbalance difficulties, [25] presented an ensemble approach using a multiwindow technique. In particular, the algorithm creates three windows: the current data block, the most recent minority cases, and the pool of base classifiers. The authors of [26] developed an expanded and enhanced version of the conventional dynamic weighted majority (DWM) to deal with the imbalance issue properly, which they dubbed Dynamic Weighted Majority for Imbalance Learning (DWMIL). Furthermore, DWMIL adopted an under-bagging method during data preprocessing to deal with class imbalance. It does, however, have the disadvantage of overfitting. In [27], dynamic classifier ensemble selection had proposed for unbalanced drifting data streams. In [28], the Kappa Update Ensemble (KUE) technique was introduced, which used the Kappa statistic to update the weights of base classifiers dynamically. By examining changes in measure values, distributions, and gradients with diverging class proportions, the authors of [29] have provided measure dynamics. In [30], a novel solution to class imbalance learning called Probability Density Machine (PDM) had proposed. First, it analyzed why imbalanced data distribution makes the performance of the predictive model decline in theory. Then proposed a novel PDM solution to address the class imbalance problem.

In [31], an efficient incremental semi-supervised classification model called Classification Over Drifting and Evolving Streams (CODES). The CODES consists of an efficient incremental semi-supervised learning module and a dynamic novelty threshold update module. In [32], novel Streaming Data-based Markov Boundary (SDMB) by linking dynamic Action Dictionary (AD) trees with online streaming data. In [33], stream data classification SDC topology on storm had proposed. SDC suggested a self-adaptive stream data classification framework for effective stream data classification on storms for the classification algorithms based on the matrix. In [34], the majority voting-based mechanism called Ensemble learning Stream (ElStream) framework had proposed for concept drift detection. ElStream utilized the conventional and ensemble machine learning classifiers to classify streaming data. In [35], a realistic semi-supervised emerging class identification methodology had proposed. The authors performed live normalization along the data stream and suggested a classification strategy that leverages a minimal number of true labels to train and detect emergent classes.

Two-Stage Cost-Sensitive (TSCS) classification was developed in [36] to solve the difficulties of idea drift and class imbalance in data stream classification. TSCS used the cost information during the feature selection and categorization stages. In [37], an adaptive chunk-based dynamic weighted majority incremental learning approach was suggested (ACDWM). ACDWM had developed to cope with idea drift in uneven streaming data. A unique Cost-Sensitive based Data Stream (CSDS) categorization had presented in [38]. The CSDS included cost information throughout the data preparation and classification stages using the ReliefF algorithm. During the classification phase, a cost-sensitive weighting method had developed to improve the ensemble's overall performance. The approximate linear dependency (ALD) approach was one of the sparsification techniques used to create the Prototype-based Kernel Classifiers (PKC) in [39]. It offered a good balance between the accuracy and complexity of kernelized nearest neighbor classifiers.

B. Research Gap Analysis

In the above section, we have reviewed the various recently proposed streaming data classification methods under different categories. The streaming data classification to address the class imbalance problem had studied in [21-30]. The streaming data classification with addressing the concept drift had been proposed in studies in [31-35]. And the stream data classification with concept drift and the class imbalance had studied in [36-39]. However, given the research challenges listed below, skewed dynamic data stream categorization remains a difficult task.

- Although idea drift and class imbalance have received considerable attention [21-35], the combined solutions received less attention. Recently explored joint solutions in [36-39] are insufficient for the skewed dynamic data stream classification problem.
- Major solutions proposed for stream data learning and classification were based on a sliding window-based mechanism which leads to poor stream data mining performances with higher costs due to duplicate chunks caused by moving sliding windows.
- Data redundancy is a significant challenge for the existing data stream classification solutions [21-39]. The lack of data approximation techniques causes

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

higher computational overhead and poor stream data classification performances.

- Existing solutions failed to bridge the trade-off between class imbalance, concept drift, data redundancy, and performance parameters. Stream data processing solutions in [21-30] have focused on the class imbalance problem but limit the other challenges and vice-versa using [31-35]. Methods in [36-39] have addressed concept drift and class imbalance, but data approximation and performance improvement (computational and classification) remain unresolved.

C. Contributions

To end this, we propose a novel solution for ADSC to achieve the SEDS classification with trade-offs among all key requirements such as concept drift, class imbalance, data redundancy, and performance improvement. The novelty of ADSC is explored below contributions.

- The integrated mechanism of adaptively classifying the SEDS classification model ADSC consists of timestamp-based data approximation, preprocessing, and adaptive clustering with newly discovered classes.
- ADSC was further extended with supervised machine learning algorithms for the classification of incoming skewed data streams into either of the newly discovered classes for high accuracy.
- The effective FM method uses a suitable hash function to approximate the data stream currently being received in a single pass with the least amount of memory and processing time required. The predicted data streams are preprocessed to remove superfluous noise while maintaining sensitive and original data.
- The timestamp-based adaptive clustering algorithm is proposed to discover the new classes for the received data chunks. The proposed clustering solution solves the class imbalance problem by forming clusters with relative distributions of streamed data among different classes.

The adaptive clustering discovers the maximum possible new classes according to features found, which also helps overcome the concept drift problem while classifying the newly incoming stream data.

III. ADSC FRAMEWORK

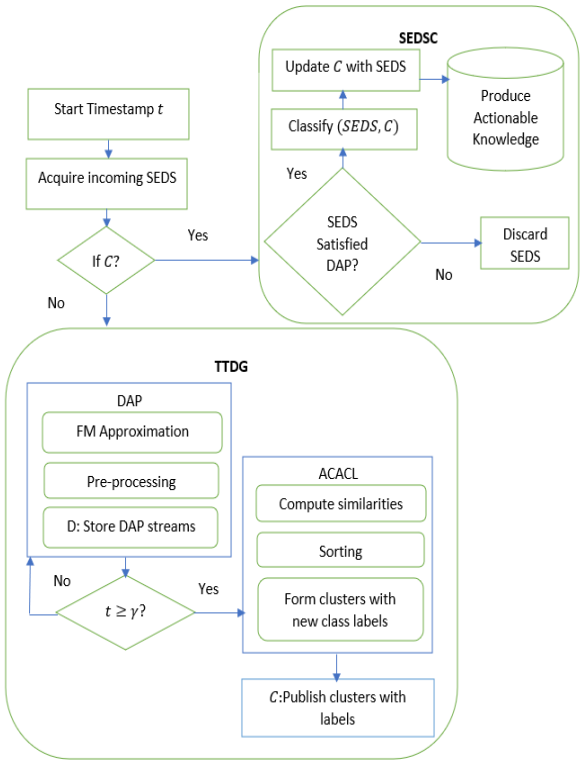

Figure 1. Architecture of the proposed ADSC framework

As per the contributions discussed for ADSC, this section presents the design and methodology for the same. Figure 1 shows the architecture of the ADSC framework for SEDS classification with higher accuracy and cost efficiency. The ADSC consists of phases like acquiring incoming SEDS, Timestamp-based Training Data Generation (TTDG), and SEDS Classification (SEDSC). The streaming data is generated daily in the context of the particular application. From that streaming data, end users require appropriate actionable knowledge discovery at the end of each day. Therefore, we started the timestamp parameter t at the beginning of each working day in the proposed model. The acquired SEDS is passed to the TTDG block until the publication of adaptive clusters with their class labels C. The TTDG block consists of phases like Data Approximation and Pre-processing (DAP) and Adaptive Clustering with Assigning Class Labels (ACACL). The goal of the TTDG phase is to perform the data approximation and preprocessing until the required timestamp threshold $\gamma$is satisfied. Once the threshold is satisfied, the ACACL mechanism is launched for the adaptive classification or grouping of initially acquired SEDSs using similarity metrics into k number of classes. The choice of k classes purely depends on the context of the end-user application. The ACACL produced the automatically classified SEDSs into different classes according to their properties. Hence, it suppresses the challenges of class

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

imbalance and concept drift for each incoming SDES classification.

The outcome of the TTDG phase is the published clusters with their corresponding newly discovered class labels in the form of vector C. Once vector C is published, the TTDG phase further will not be launched until the end of the particular day. We perform the SEDSC phase for each newly streamed data classification. In the SEDSC phase, we first verify the duplicate entry of incoming SEDS using DAP. If it's duplicate SEDS, then directly discarded. Otherwise, the classification is performed by passing inputs current SEDS and C to the supervised classifier. The supervised classifier first trains the data in C and then classifies the SEDS into either of the classes in C. According to the classification outcome, the current SEDS joins a particular class in vector C. Therefore, after the classification of each new SEDS, the vector C is updated. It means that clusters are changed regularly, thus the named adaptive classification or clustering approach. Once periodic data streaming and classification are completed, the actionable knowledge discovery is performed for the decision-making process. The functionality of ADSC is further explained step-by-step. Table 1 presents the list of symbols and their significance.

Table 1. List of symbols

Symbol	Significance
C	Set of clusters with associated labels
k	Number of clusters
t	Current SEDS time
γ	Timestamp threshold
s	Input data stream SEDS
S	Set of input data streams
D	Approximated and preprocessed set of data streams
j	Unique sensitive attribute in each data stream
R	Set of raw clusters
$\ddot{C}$	Set of centroids for clusters in R
C	Optimized clusters with associated labels
x	Number of SEDSs in D
y	Number of attributes in each SEDS
n	Maximum number of SEDS in each cluster
L	Set of class labels associated with each SEDS in C

A. TTDG

As shown in figure 1, the TTDG block consists of two phases, DAP and ACACL.

1) DAP

DAP capabilities include preprocessing each received tuple without losing crucial information and FM-based data stream approximation. Figure 2 depicts the DAP phase process in detail. The input is acquired into variable s and then added into vector S. S is the input data stream that contains one or more tuples. If the number of tuples in S exceeds 1, we start the FM and preprocessing algorithms. Estimating the total number of distinct data streams is the fundamental aim of the FM method. Nevertheless, we looked into it to separate the different data streams and get rid of the unnecessary data streams. This method is applied for each incoming tuple to prevent data redundancy. The improved FM method incorporates preprocessing mechanism, as shown in algorithm 1.As seen in algorithm 1,we were able to successfully approximate the periodic data stream using the FM algorithm. This study looks into the advantages of employing the FM technique to count the number of unique tuples in order to find any redundant or duplicate incoming tuples. The FM algorithm's primary functionality is in defining the hash function (sr. no 9 in Algorithm1), calculating which stream each attribute's hash function belongs to (sr. no 11 in Algorithm1), hash value conversion into Binary (sr. no 12 in Algorithm1), counting trailing zeros of a binary number (sr. no 13 in Algorithm1), and computing the total distinct streams in S (sr. no 15 & 16 in Algorithm1).

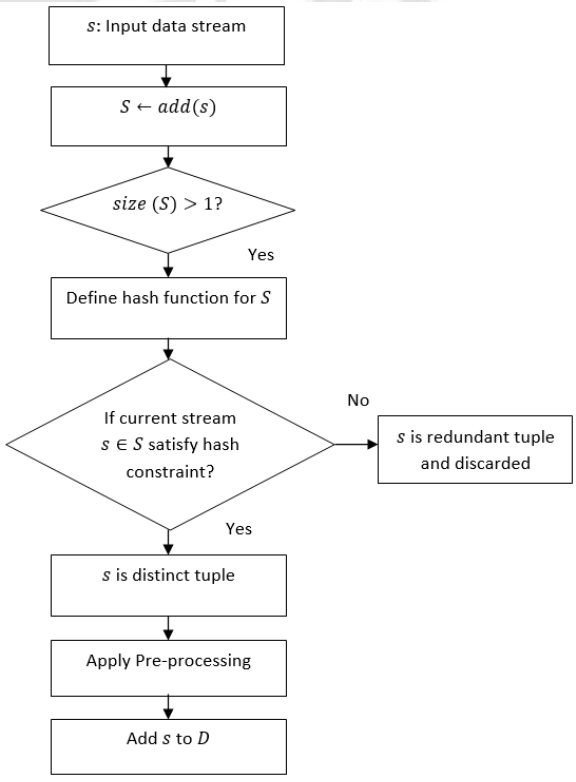

Figure 2. Process of data approximation and preprocessing

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

As seen in steps 17–24 of method 1, once the number of distinct items was known, we used that parameter to assess if the stream s was unique or redundant. We invoked algorithm 2 from algorithm 1 to preprocess the incoming data stream and saved the final preprocessed streams in D. The manual identification of the distinctive feature of each streaming. We defined this work's hash function shown in Eq. (1).

$$h(x) \leftarrow (a.x + b) \bmod c \quad (1)$$

Where we set, The x reflects the attribute value of the current stream. To compute the hash values, we used the values a=1, b=6, and c=32.

Algorithm 1: DAP

Input

s: $input\ data\ stream$

λ: $pre-defined\ time\ constraint$

j: $unique\ senstive\ attribute$

Output

D: $Approximted\ and\ preprocessed\ data\ stream$

1. Initial timer $t = 0$
2. $s \leftarrow acquire\ (SEDS)$
3. $S \leftarrow add(s)$
4. If $(size\ (S) > 1)$
5. For $i = 1: size\ (S)$
6. Estimate the unique sensitive attribute from all streams
7. $w(i) \leftarrow S(i,j)$, record the j^{th} position unique value
8. End For
9. Define a hash function for stream w using Eq. (1) and apply
10. For $i = 1: size\ (w)$
11. $h(i) \leftarrow (a.w(i) + b) mode\ c$
12. $h(i) \leftarrow binary\ (h(i))$
13. $r(i) \leftarrow trailingzeros(h(i))$
14. End For
15. Compute maximum value: $R \leftarrow \max(r)$
16. Compute distinct tuples: $N \leftarrow 2^R$
17. If $(N == size\ (w)$
18. The current stream s is unique and apply preprocessing
19. $p \leftarrow algorithm\ 2(s)$
20. $t++$
21. Else
22. Discard stream s from stream S as: $S \leftarrow substract(s)$
23. $t++$
24. End If
25. Else
26. $S \leftarrow algorithm\ 2(S)$
27. $t++$
28. End If
29. $D \leftarrow add(p)$
30. Check time constraint
31. If $(t \geq \lambda)$
32. Return (D), Launch ACPP phase
33. Reset timer $t = 0$, goto step 1
34. Else
35. goto step 2
36. End If

The preprocessing of each input stream s is depicted in Algorithm 2. First, we verified that the property is a string. If it was a string, we used NLP to transform the wrong strings into a meaningful form and eliminate the noise in the string. Aside from that, in our study for numeric characteristics, we tackled the difficulties of missing or partial data. Using the newVal(.) function, we have identified the numeric properties and replaced them with pertinent values. The newVal(.) uses statistical analysis of the same attributes of other streams to determine the value that is most pertinent.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

Algorithm 2: Data Pre-processing

Input

s: *input data stream*

Output

p: *preprocessed data stream*

1. Acquisition of test stream s
2. For each attribute, each attribute $i = 1$: $size\ (s)$
3. If $(s(i) == string)$
4. $p(i) \leftarrow Lemmatization\ (s(i))$
5. End If
6. If $(s(i) == NULL)$
7. $p(i) \leftarrow newVal()$
8. End If
9. End For
10. Return (p)

2) ACACL

If the timestamp threshold is satisfied, TTDG calls the proposed adaptive clustering mechanism to estimate the novel classes for the recorded SEDSs using a modified k-means clustering mechanism. ACACL aims to classify the recorded distinct SEDSs into unique classes to prevent class imbalance and concept drift problems via adaptive clustering. The received data D is initially divided into different raw clusters using the basic k-means clustering approach. The reasons for selecting k-means clustering over the others are (1) it is straightforward and computationally efficient, (2) concept drifts are effectively handled as outliers cannot be prevented, and (3) data loss is prevented. The centroids are estimated using k-means as:

$$[R, \ddot{C}] = kmeans\ (D, k) \quad (2)$$

where n indicates no. of clusters. R is the collection of k raw clusters into which the streams of D are subdivided. $\ddot{C}$ represents the centroid SEDS for each cluster. However, the basic k-means failed to overcome the class imbalance problem as each cluster may have a distinct number of SEDSs. It may also lead to concept drift problem for newly incoming SEDS classification. Therefore, we further proposed a novel approach to overcome the class imbalance and concept drift problems utilizing the initially computed centroids. We define the constraint value n for each cluster to have maximum n SEDSs. The value of n is computed by:

$$n = \left|\frac{size\ (D)}{k}\right| \quad (3)$$

Algorithm 3 shows the detailed procedure of the ACACL mechanism where the clusters are formed with their associated unique label. As demonstrated in algorithm 3, By improving the results of k-means clustering, it is accomplished. As shown in algorithm 3, it takes inputs such as R, $\ddot{C}$, $k, n, x,$ and y and returns the set clusters C that ensures the class balancing with k number of distinct class labels. Prior to improving the current clusters, we have first estimated the distance between i^{th} SEDS of the j[th] cluster and j[th]centroid. Manhattan distance technique is used to measure the distance in $getDist\ (.)$ function. It is determined as the total of two numeric vectors of two tuples' absolute differences. The vector P, which contains the full SEDS and its distance value, is used to measure all of the distances. The SEDSs in P were then arranged according to decreasing distance values. To solve the concerns of class imbalance and concept drift, the clusters are finally reorganised to guarantee a maximum of n SEDSs per cluster. The no. of SEDSs in each cluster should be less than or equal to n. The ACACL approach also ensures the privacy preservation notion of k-anonymization by classifying the streaming records equally into k number of different classes. Hence, it prevents the problem of data loss as well.

Algorithm 3: Adaptive clustering with labels

Inputs

R: *Set of raw clusters*
$\ddot{C}$: *set of centroid tuples for each cluster*
k: *number of clusters*
x: *number of SEDSs in D*
y: *number of attributes in each SEDS*
n: *maximum number of SEDSs in each class*

Output

C: *class imbalanced clusters*
L: *associated class labels*

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

1. Initialize, $P \leftarrow zeros(x, y+1)$, $q = 1$
2. For $i = 1{:}k$
3. For $j = 1{:}size\,(\mathrm{R(i)})$
4. d $\leftarrow getDist(\mathrm{R(i,j)},\ C(\imath))$
5. $P(q,\ 1{:}x) \leftarrow$ R(i, j)
6. $P(q, y+1) \leftarrow$ d
7. $q++$
8. End For
9. End For
10. $temp \leftarrow sort(\text{descending},\ P(:, m+1))$
11. $C \leftarrow zeros(x, y+1)$
12. for $j = 1{:}k$
13. for $i = 1{:}size\ (temp)$
14. if ($i \leq$ n)
15. C(i, :) $\leftarrow join\ (temp\ (i, :))$
16. $L(i) \leftarrow j$: assign new class label
17. else
18. break
19. end if
20. end for
21. end for
22. Return (C, L)

B. SEDSC

The TTDG phase acts as the formation of training data with its associated newly discovered labels during the initial timestamp constraint. Algorithm 4 shows the proposed approach for the adaptive classification of incoming SEDS. The produced training data C is further utilized by the SEDSC phase for the classification of each incoming SEDS into either cluster in C using the supervised classifier. After classifying each incoming SEDS, the original vector C is updated with the addition of SEDS to the cluster of classified classes. Therefore, SEDSC acts as the testing phase in the proposed model. The data in both C and incoming preprocessed SEDS p is normalized before applying the classifier using the min-max normalization technique. Normalization is a scaling method used in machine learning to modify the values of numeric columns in a dataset to use a common scale during data preparation. It is necessary when the ranges of machine learning model features differ to improve the classification performances. The online SEDS data contains significant variations among their feature ranges which may lead to misclassification problems. We applied the min-max classification on each SEDS in D (Eq. (4)), and incoming test preprocessed SEDS p (Eq. (5)) as below:

$$C^{norm}(i, :) = \frac{C(i,:) - \min(C(i,:))}{\max(C(i,:)) - \min(C(i,:))} \quad (4)$$

$$p^{norm} = \frac{p - \min(p)}{\max(p) - \min(p)} \quad (5)$$

Where C^{norm} and p^{norm} represent the normalized clusters features and test SEDS features.

Algorithm 4: SEDSC

Inputs

s: *Incoming SEDS as test data*
C: *current train database*
L: *current associated class labels*
S: *currently updated SEDSs*

Output

C: *Updated train database*

1. $S \leftarrow add(s)$
2. Verify redundancy of $s$using FM approach
3. if $(s\ satisfied, DAP)$
4. 's is unique'
5. $p \leftarrow preprocessed\ data$
6. else
7. 's is redundant'
8. Discard s
9. end if
10. Normalize data
11. $C^{norm} \leftarrow using\ Eq.(4)$
12. $p^{norm} \leftarrow using\ Eq.(5)$
13. $trainDB \leftarrow train(C^{norm}, ML)$
14. Classify SEDS
15. $class \leftarrow classify(p^{norm}, trainDB)$
16. Discover the index of class
17. $index \leftarrow (class == L(:))$
18. Insert discovered class for input SEDS
19. $L \leftarrow insert\ (L, class, index + 1)$
20. $C \leftarrow insert\ (C, p, index + 1)$
21. Return (C, L)

After that, C^{norm} is trained with different machine learning (ML) classifiers such as Artificial Neural Network (ANN), Support Vector Machine (SVM), Random Forest (RF), and Naïve Bayes (NB). After training, the p^{norm} is classified into either of the labels in vector L. According to the label detected for incoming SEDS p, it is added to the associated cluster in C with the addition of the detected class in L. Therefore, the training data C with L is periodically updated with the addition of newly classified incoming SEDS. This process goes on until the end of live streaming. Once the live streaming ends on a given day, the final vector of C with classified labels L is published to the end users. The end-users then applied the various mechanisms to discover the actionable information for decision making. This information can be the discovery of

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

associated samples for each class, i.e., the class distribution ratio. The class distribution ratio helps the decision makers for future planning about their business activities for higher gain and productivity. It is commonly called actionable knowledge extraction. The actionable knowledge can also be the average, maximum, and minimum information about the attributes responsible for productivity.

IV. Experimental Results

The results of experimental work for performance analysis are presented in this section. We utilized the Python tool to develop and analyze the suggested model using cutting-edge methodologies. The studies were conducted using a Windows 10 computer with an Intel I5 CPU and 8GB of RAM. Each scenario has been run 20 times, and the results have been averaged. In this study, we conduct experimental analysis on two real-world datasets and one synthetic dataset. To assess the classification performance of the data stream classifier, the real dataset is taken from The University of California Irvine (UCI) [40] machine learning repository. Table 2 shows the information about these datasets. A brief description of each dataset is given below.

Table 2. Dataset statistics

Dataset	Number of features	Classes	Type
LED	24	10	Synthetic
Covertype	54	7	Real-time
PockerHand	10	10	Real-time

A. Experimental Background

1) Streaming Datasets

The***LED*** [41] is a well-known synthetic dataset. The LED dataset aims to predict the next digit on a seven-segment LED display. Every digit has a 10 % probability of being displayed. The LED dataset comprises 24 binary features, 17 of which are worthless. Interchanging seven class-relevant characteristics causes concept drift. This effort produces a stream of 100,000 occurrences, with concept drift occurring every 25,000. In addition, we have introduced 10% redundant streaming data with every 25,000 instances. The original LED data size had increased to a stream of 1,10,000 occurrences.

The ***Covertype [42]*** dataset comprises 54 attributes defining different forest cover types. It includes 5,81,012 occurrences from the US Forest Service (USFS) Resource Information System that define 7 forest cover classes for cells of 3030 meters (RIS). It is real-time evolving streaming data that already incorporates drifts and redundant streams.

The ***PokerHand [42***] is another streaming dataset that evolves in real-time. This dataset represents the challenge of determining the winning hand in a poker game. It contains 1,025,010 instances defining all conceivable poker hands, with each instance depicting a hand made up of five cards drawn from a standard deck of 52 cards. Each card in the deck has two properties. Each hand has described by ten features.

2) State-of-the-art Methods

To compare the performance of the proposed ADSC model, we have selected the recently introduced similar techniques for SKDSs classification. The methods such as CODES [31], ElStream [34], SACCOS [35], TSCS [36], ACDDWM [37], and CSDS [38]. All these methods were proposed for the classification of SEDS to address challenges of class imbalance, concept drift, or both. We applied these methods to all three datasets. Each dataset is divided into training (70%) and testing (30%) for the evaluations. In the proposed model, we generated the initial vector C to contain the 70 % dataset with its corresponding labels vector L. The timestamp threshold was set to achieve 70 % training dataset generation in the TTDG phase and 30 % SEDSs classification using the proposed SEDSC phase. The value of k is set to the number of classes of each dataset in the proposed model. The value of k defines the number of clusters in the ACACL algorithm. The suggested technique had compared to the six current methods mentioned above in terms of overall accuracy and computing cost.

3) Performance Metrics

Performances are measured using the classification performance metrics such as accuracy, precision, recall, and F1-measure parameters. Additionally, we have measured the system execution time for the training and classification process to estimate cost-efficiency. The formulas for computation accuracy, precision, recall, and F1-measure are computed using True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). These four metrics were computed from the confusion matrix of each classifier and technique. The formulas are given below:

$$accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (6)$$

$$precision = \frac{TP}{TP+FP} \quad (7)$$

$$recall = \frac{TP}{TP+FN} \quad (8)$$

$$f1 = \frac{2.precision.recall}{precision+recall} \quad (9)$$

B. Performance Investigation

This phase aims to investigate the proposed model's performance using different classifiers for normalized dataset and original dataset features. We have measured the results for each dataset using ANN, SVM, RF, and NB classifiers.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

1) LED Dataset

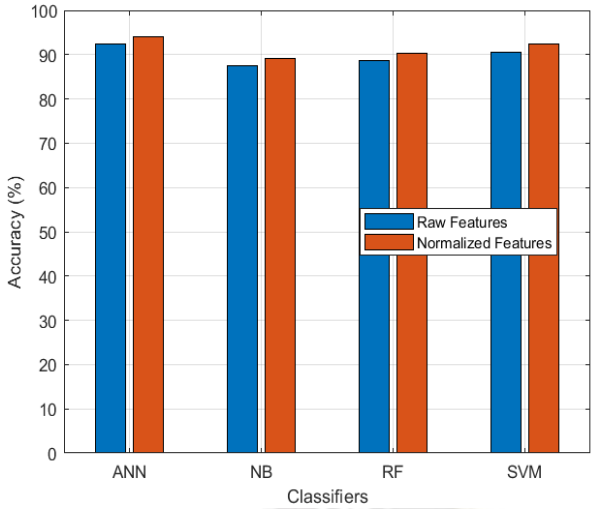

Figure 3. Accuracy performance investigation using LED dataset

Figures 3, 4, 5, and 6 show the results of calculating accuracy, precision, recall, and F1-measure using the synthetic LED dataset, respectively. In essence, in these results, we evaluated the suggested model ADSC by using four distinct datasets and considering both the normalization and the raw features for the training and testing phases. It is recommended that the ADSC model be studied utilizing a variety of datasets, methods, and algorithms before analyzing the model with its underlying approaches. It encourages us to look into the ADSC so that we can claim its effectiveness and scalability.

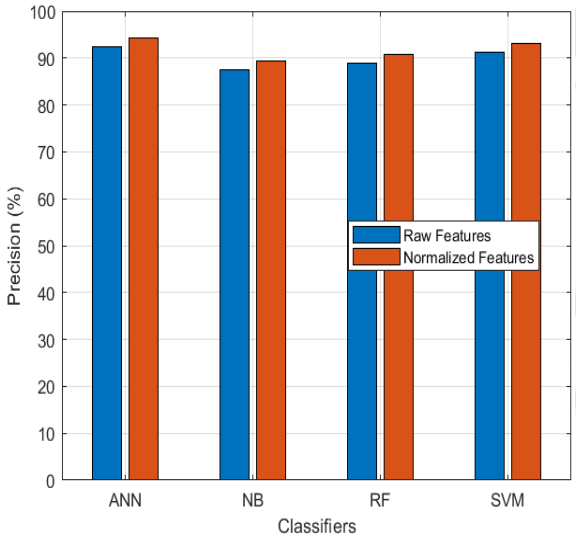

Figure 4. Precision performance investigation using LED dataset

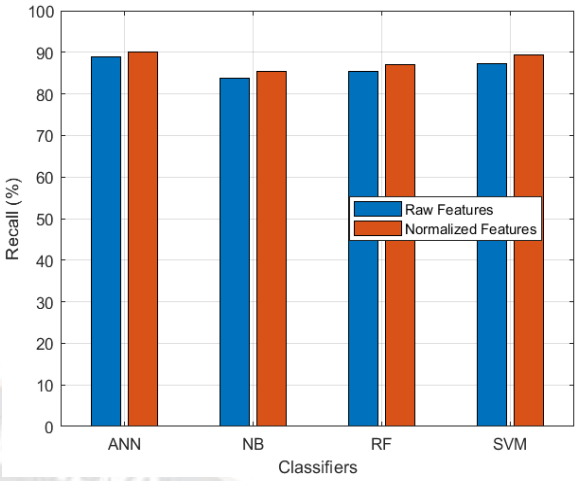

Figure 5. Recall performance investigation using LED dataset

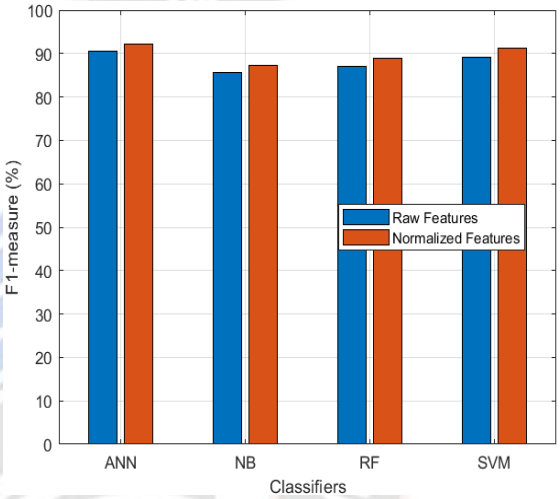

Figure 6. F1-measure performance investigation using LED dataset

We manually added the concept drifts and 10 % redundant streams in the LED dataset. Accordingly, the proposed model using different classifiers delivered the optimum performances. The performances are improved due to a timestamp-based mechanism that only overcomes the class imbalance and concept drift challenges but also reduces the redundant data in the DAP algorithm. The data approximation with preprocessing affects the overall classification performances using the LED dataset. As observed in Figures 3-6, the proposed model using a normalization mechanism achieved performance improvement for the SEDS classification compared to raw streaming features.

2) Covertype Dataset

This section presents the performance investigation of a real-time skewed evolving streaming dataset called Covertype. The dataset contains more than 5 lakh streaming data that consists of concept drifts and redundant streaming entries. Similar to the LED dataset, we have analyzed the performances of different classifiers using raw and normalized streaming features. Figures 7-10 demonstrate the accuracy, precision,

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

recall, and F1-measure results using ANN, SVM, RF, and NB classifiers. The raw features have significant variations among the feature ranges of different streams, directly affecting the classification performance using each classifier. Normalization prevents raw data and numerous dataset difficulties by establishing new values and keeping broad distribution and a ratio in data. It also increases the accuracy and performance of machine learning classifiers by employing a variety of methodologies and algorithms. Therefore, the proposed model using the min-max normalization mechanism delivered higher performances for accuracy (Figure 7), precision (Figure 8), recall (Figure 9), and F1-measure (Figure 10) for each classifier. The proposed model achieved 96.12 % accuracy for the Covertype dataset using the ANN classifier.

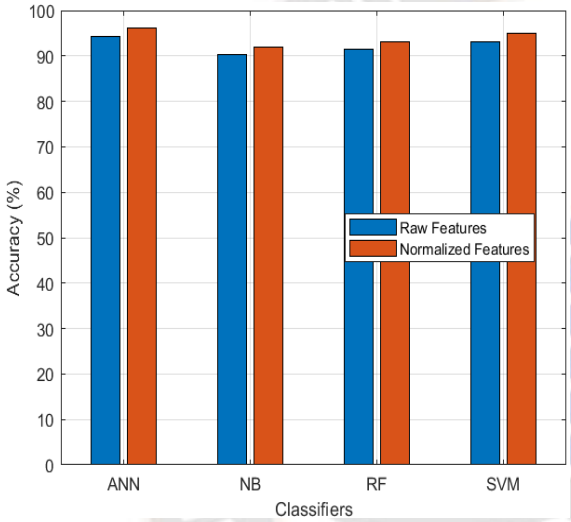

Figure 7. Accuracy performance investigation using Covertype dataset

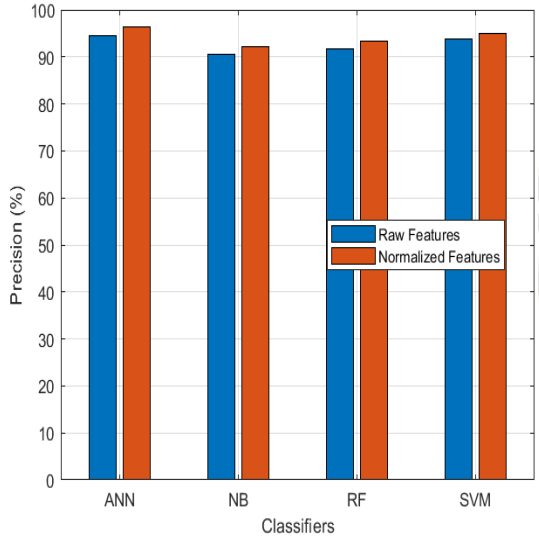

Figure 8. Precision performance investigation using Covertype dataset

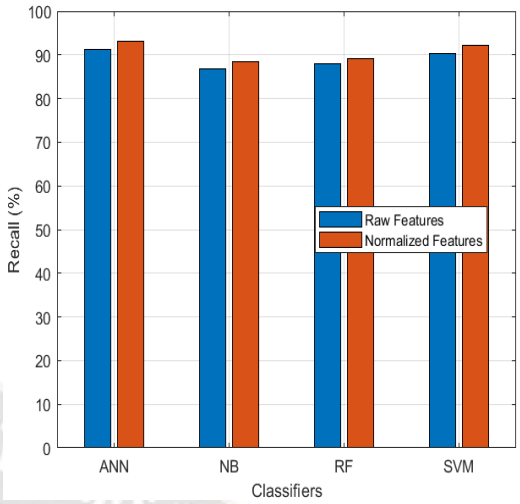

Figure 9. Recall performance investigation using Covertype dataset

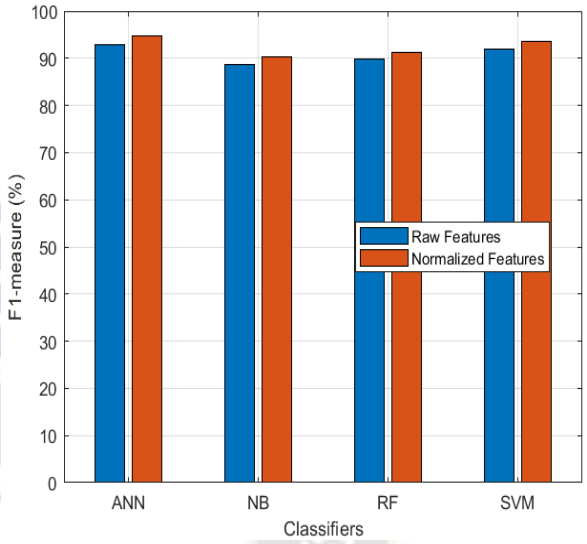

Figure 10. F1-measure performance investigation using Covertype dataset.

3) PokerHand Dataset

PokerHand is yet another real-time streaming dataset that is 20 times bigger than the Covertype streaming dataset. To verify the scalability of the proposed model, we evaluated the performances of the high-dimensional PokerHand dataset in this paper. Figures 11-14 show the accuracy, precision, recall, and F1-measure performances, respectively. These outcomes demonstrate the similar trend we have already observed for the other two datasets in Figures 3-10. We revealed reasons for performance improvement using the min-max normalization mechanism. Another observation about the classifiers is that ANN delivered higher classification accuracy than other classifiers. ANN organizes algorithms in layers so that they may understand and construct intelligent judgments independently. On the other hand, machine learning makes judgments solely based on what it has learned. There are two types of machine learning classifiers: supervised and unsupervised learning.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

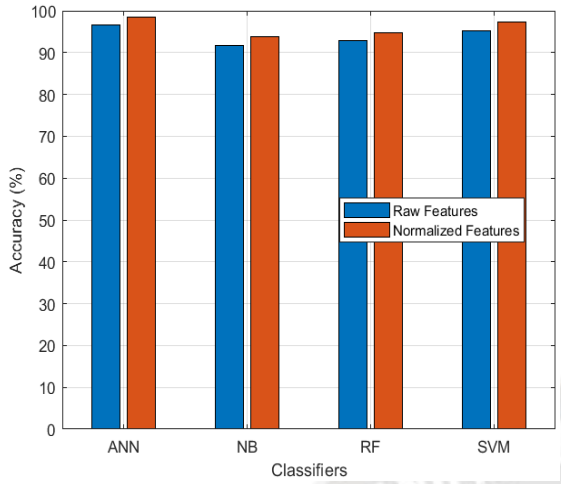

Figure 11. Accuracy performance investigation using PokerHand dataset

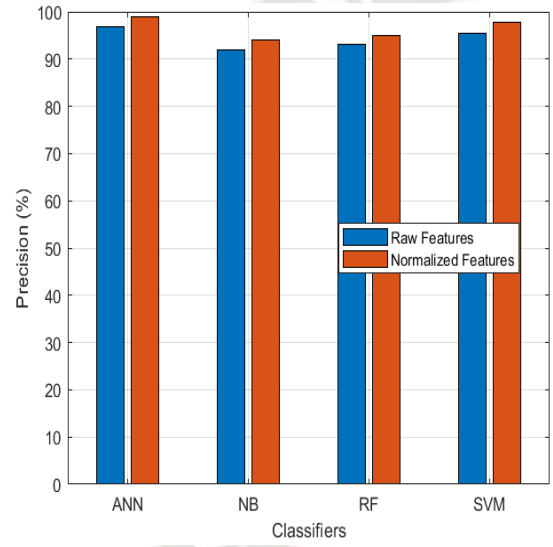

Figure 12. Precision performance investigation using PokerHand dataset

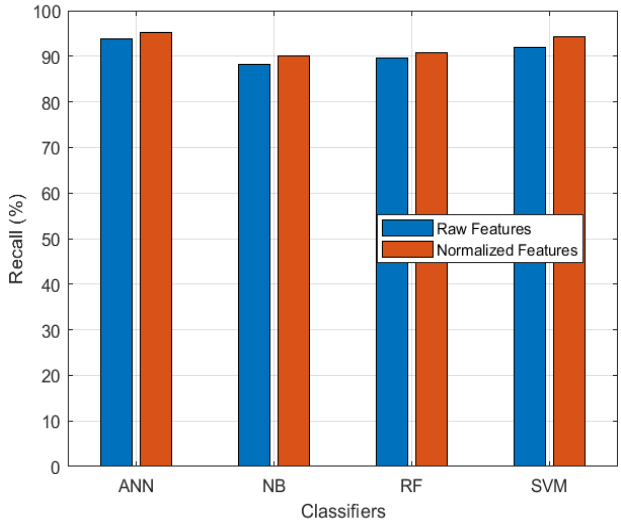

Figure 13. Recall performance investigation using PokerHand dataset

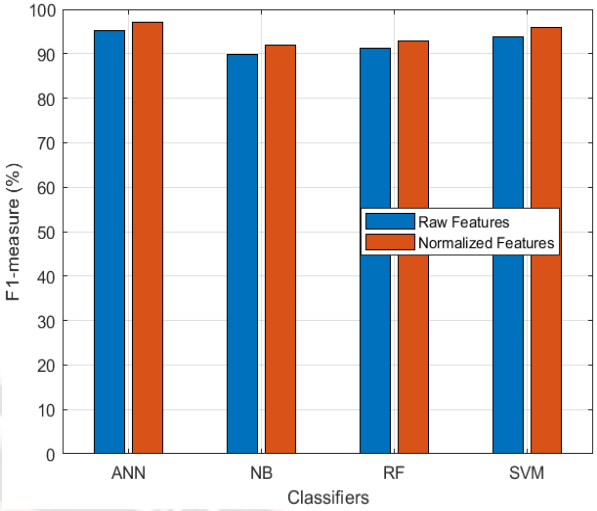

Figure 14. F1-measure performance investigation using PokerHand dataset

The higher classification accuracy received for the PokerHand dataset is 98.49 % using the ANN classifier. Among all three datasets, the proposed model delivered the best accuracy using the PokerHand dataset and lower accuracy using the LED dataset (94.12 %). Although PokerHand is the high dimensional dataset, it has a lower probability of concept drifts and redundant data than the LED dataset. Another reason for higher performances using the PokerHand dataset is that each class has a higher number of training streaming entries compared to the LED and CoverType datasets.

C. State-of-the-art Analysis

In this section, we present the comparative analysis of the proposed model with 6 recently proposed SEDS classification methods. All methods were implemented along with the proposed model under the same execution environment and dataset conditions (70 % training and 30 % testing). We measured the performances using each dataset and averaged their performances after 10 executions for the parameters such as accuracy, F1-score, and execution time (sum of training and testing). The execution time parameters demonstrate the cost-efficiency metrics for all methods. Tables 3, 4, and 5 demonstrate the comparative analysis of underlying classification techniques and the proposed ADSC method using LED, Covertype, and PokerHand datasets, respectively. The comparative results using each dataset reveal that the proposed ADSC model significantly improved the SEDS classification performance compared to underlying techniques. For the LED dataset, the ADSC model achieved 94.12 % accuracy and 92.12 % F1 measure with 138.22 seconds of cost-efficiency. For the Covertype dataset, the ADSC model produced 96.12 % accuracy and 94.64 % F1-measure with 317.22 seconds cost-efficiency. For the LED dataset, the ADSC model achieved 98.49 % accuracy and 97.04 % F1 measure with 1391.65 seconds of cost-efficiency. The

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

execution time increases with increased dataset dimensions as it needs additional processing for training and testing. Among all the techniques, ADSC takes minimum computational time (lower resource utilization) compared to other methods. The cause of cost efficiency using the ADSC model is the data approximation algorithm for reducing redundant data streams. The proposed DAP model significantly reduces further training and classification processing requirements. Whereas other techniques failed to overcome the challenge of redundant streaming entries, and hence, they required additional processing time to classify similar entries. Also, some existing methods are functioning according to the sliding window leading to significant data duplications and hence the higher resource utilization compared to the proposed model.

Table 3. State-of-the-art analysis using LED dataset

	Accuracy (%)	**F1-score (%)**	**Cost Efficiency (Seconds)**
CODES [31]	78.33	75.09	167.23
ElStream [34]	80.02	77.22	161.38
SACCOS [35]	79.32	75.81	172.88
TSCS [36]	84.91	81.92	193.33
ACDDWM [37]	81.23	78.67	174.33
CSDS [38]	87.31	84.93	154.38
ADSC	94.12	92.12	138.22

Table 4 State-of-the-art analysis using Covertype dataset

	Accuracy (%)	**F1-score (%)**	**Cost Efficiency (Seconds)**
CODES [31]	81.29	79.21	451.21
ElStream [34]	84.11	82.03	436.33
SACCOS [35]	82.45	80.19	463.12
TSCS [36]	87.22	85.02	488.99
ACDDWM [37]	84.02	81.78	470.32
CSDS [38]	91.09	88.23	405.33
ADSC	96.12	94.64	371.33

Table 5. State-of-the-art analysis using PokerHand dataset

	Accuracy (%)	**F1-score (%)**	**Cost Efficiency (Seconds)**
CODES [31]	82.51	80.29	1756.05
ElStream [34]	85.31	83.17	1681.65
SACCOS [35]	83.65	81.34	1815.6
TSCS [36]	88.47	86.33	1944.95
ACDDWM [37]	85.37	82.99	1851.6
CSDS [38]	92.59	89.63	1526.65
ADSC	98.49	97.04	1391.65

The accuracy and F1-score performances using the proposed model also shows improvement using each dataset compared to existing methods. The timestamp-based mechanism for data approximation, preprocessing, and adaptive clustering approach can overcome the problems of data redundancy, concept drift, and class imbalance using the proposed model effectively compared to underlying techniques. Therefore, it shows the significant performance improvement for SEDS classification using the ADSC model. The adaptive clustering produces the incremental training dataset while classifying each incoming SEDS entry which overcomes the concept drift problem and classifies each SEDS entry with higher accuracy. The clustering-based approach also suggests similar distributions of streamed data across all the classes to prevent class imbalance conditions. Finally, the proposed model achieved an efficient trade-off among all key requirements of SEDS classification.

V. CONCLUSION

We proposed the ADSC model in this paper for the efficient classification of SEDSs with concept drift, class imbalance, cost, and performance efficiency. The ADSC model had constructed using novel steps such as data approximation, adaptive clustering, and supervised classification. The data approximation with pre-processing using the FM algorithm produced the redundant data removal without affecting any data loss. It has improved the overall classification and cost efficiency of the proposed model. The ACACL approach effectively generated training data by mitigating the concept drift and class imbalance challenges. The SEDSC phase accurately classified the newly incoming streamed data after verifying it with the DAP technique for data redundancy. Overall mechanisms lead to significant performance improvement using the proposed model compared to underlying methods using LED, Covertype, and PokerHand

datasets. The proposed approach enhanced accuracy by 13+% and lowered cost needs by 18.97% for the LED dataset. For the Covertype dataset, the proposed model had improved accuracy by 11+% and reduced cost requirements by 17.92 %. The proposed methodology increased accuracy for the PokerHand dataset by 12+% while lowering costs by 21.05%. The future recommendations for this work include (1) applying the deep learning classifier like Long Term Short Memory (LSTM) to further improve the accuracy, (2) applying the optimization techniques to enhance the functioning of adaptive clustering, and (3) investigating the privacy preservation notions for the proposed model.

REFERENCES

[1]. Da Silva, T. P., Urban, G. A., Lopes, P. de A., & Camargo, H. de A. (2017). A Fuzzy Variant for On-Demand Data Stream Classification. 2017 Brazilian Conference on Intelligent Systems (BRACIS). doi:10.1109/bracis.2017.60.

[2]. Sasikala, S., & Devi, D. R. (2017). A review of traditional and swarm search-based feature selection algorithms for handling data stream classification. 2017 Third International Conference on Sensing, Signal Processing and Security (ICSSS). doi:10.1109/ssps.2017.8071650.

[3]. Roberto, J., Junior, B., & Nicoletti, M. do C. (2016). Functionally expanded streaming data as input to classification processes using ensembles of constructive neural networks. 2016 International Joint Conference on Neural Networks (IJCNN). doi:10.1109/ijcnn.2016.7727424.

[4]. Krawczyk, Bartosz &Stefanowski, Jerzy & Wozniak, Michal. (2014). Data stream classification and big data analytics. Neurocomputing. 150. 10.1016/j.neucom.2014.10.025.

[5]. Daniel, A., Subburathinam, K., Paul, A., Rajkumar, N., & Rho, S. (2017). Big autonomous vehicular data classifications: Towards procuring intelligence in ITS. Vehicular Communications, 9, 306–312. doi:10.1016/j.vehcom.2017.03.002.

[6]. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., &Bouchachia, A. (2014). A survey on concept drift adaptation. ACM computing surveys (CSUR), 46(4), 1-37.

[7]. Webb, G. I., Hyde, R., Cao, H., Nguyen, H. L., &Petitjean, F. (2016). Characterizing concept drift. Data Mining and Knowledge Discovery, 30(4), 964-994.

[8]. Ditzler, G., Roveri, M., Alippi, C., &Polikar, R. (2015). Learning in nonstationary environments: A survey. IEEE Computational Intelligence Magazine, 10(4), 12-25.

[9]. Khamassi, I., Sayed-Mouchaweh, M., Hammami, M., &Ghédira, K. (2018). Discussion and review on evolving data streams and concept drift adapting. Evolving systems, 9(1), 1-23.

[10]. Webb, G. I., Hyde, R., Cao, H., Nguyen, H. L., &Petitjean, F. (2016). Characterizing concept drift. Data Mining and Knowledge Discovery, 30(4), 964-994.

[11]. He, H., & Garcia, E. A. (2009). Learning from imbalanced data. IEEE Transactions on knowledge and data engineering, 21(9), 1263-1284.

[12]. Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). Learning from imbalanced data streams. In Learning from imbalanced data sets (pp. 279-303). Springer, Cham.

[13]. Liu, X. Y., Wu, J., & Zhou, Z. H. (2008). Exploratory undersampling for class-imbalance learning. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 39(2), 539-550.

[14]. Krishnamurthy, A., Agarwal, A., Huang, T. K., Daumé III, H., & Langford, J. (2017, July). Active learning for cost-sensitive classification. In International Conference on Machine Learning (pp. 1915-1924). PMLR.

[15]. Cao, P., Zhao, D., &Zaiane, O. (2013, April). An optimized cost-sensitive SVM for imbalanced data learning. In Pacific-Asia conference on knowledge discovery and data mining (pp. 280-292). Springer, Berlin, Heidelberg.

[16]. Zakerzadeh H, Aggarwal CC, Barker K (2016) Managing dimensionality in data privacy anonymization. Knowl Inf Syst 49(1):341–373.

[17]. Zhang Y., Szabo C., Sheng Q.Z. (2014) Cleaning Environmental Sensing Data Streams Based on Individual Sensor Reliability. In: Benatallah B., Bestavros A., Manolopoulos Y., Vakali A., Zhang Y. (eds) Web Information Systems Engineering – WISE 2014. WISE 2014. Lecture Notes in Computer Science, vol 8787. Springer, Cham. https://doi.org/10.1007/978-3-319-11746-1_29.

[18]. Shaoxu Song, Fei Gao, Aoqian Zhang, Jianmin Wang, and Philip S. Yu. 2021. Stream Data Cleaning under Speed and Acceleration Constraints. ACM Trans. Database Syst. 46, 3, Article 10 (September 2021), 44 pages. DOI:https://doi.org/10.1145/3465740.

[19]. Peter M. Fischer, KyumarsSheykhEsmaili, and Renée J. Miller. 2010. Stream schema: Providing and exploiting static metadata for data stream processing. In Proceedings of the 13th International Conference on Extending Database Technology. 207–218. DOI: https://doi.org/10.1145/1739041.1739068.

[20]. Ester Livshits, Benny Kimelfeld, and Sudeepa Roy. 2018. Computing optimal repairs for functional dependencies. In Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems. 225–237. DOI: https://doi.org/10.1145/3196959.3196980.

[21]. Chen, S., & He, H. (2011). Towards incremental learning of nonstationary imbalanced data stream: a multiple selectively recursive approach. Evolving Systems, 2(1), 35-50.

[22]. Ditzler, G., &Polikar, R. (2012). Incremental learning of concept drift from streaming imbalanced data. IEEE transactions on knowledge and data engineering, 25(10), 2283-2301.

[23]. Mirza, B., Lin, Z., & Liu, N. (2015). Ensemble of subset online sequential extreme learning machine for class

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5985
***Article Received**: 12 November 2022 **Revised**: 12 December 2022 **Accepted**: 04 January 2023*

imbalance and concept drift. Neurocomputing, 149, 316-329.

[24]. Ghazikhani, A., Monsefi, R., &Yazdi, H. S. (2013). Ensemble of online neural networks for nonstationary and imbalanced data streams. Neurocomputing, 122, 535-544.

[25]. Li, H., Wang, Y., Wang, H., & Zhou, B. (2017). Multiwindow based ensemble learning for classification of imbalanced streaming data. World Wide Web, 20(6), 1507-1525.

[26]. Lu, Y., Cheung, Y. M., & Tang, Y. Y. (2017, August). Dynamic Weighted Majority for Incremental Learning of Imbalanced Data Streams with Concept Drift. In IJCAI (pp. 2393-2399).

[27]. Zyblewski, P., Sabourin, R., &Woźniak, M. (2021). Preprocessed dynamic classifier ensemble selection for highly imbalanced drifted data streams. Information Fusion, 66, 138-154.

[28]. Cano, A., & Krawczyk, B. (2020). Kappa updated ensemble for drifting data stream mining. Machine Learning, 109(1), 175-218.

[29]. Brzezinski, D.W., Stefanowski, J., Susmaga, R., &Szczech, I. (2020). On the Dynamics of Classification Measures for Imbalanced and Streaming Data. IEEE Transactions on Neural Networks and Learning Systems, 31, 2868-2878.

[30]. Cheng, R., Zhang, L., Wu, S., Xu, S., Gao, S., & Yu, H. (2021). Probability Density Machine: A New Solution of Class Imbalance Learning. Sci. Program., 2021, 7555587:1-7555587:14.

[31]. Bi, X., Zhang, C., Zhao, X., Li, D., Sun, Y., & Ma, Y. (2020). CODES: Efficient Incremental Semi-Supervised Classification Over Drifting and Evolving Social Streams. IEEE Access, 8, 14024-14035.

[32]. Liu, C., Yang, S., & Yu, K. (2020). Markov Boundary Learning With Streaming Data for Supervised Classification. IEEE Access, 8, 102222-102234.

[33]. Deng, S., Wang, B., Huang, S., Yue, C., Zhou, J., & Wang, G. (2020). Self-Adaptive Framework for Efficient Stream Data Classification on Storm. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 50, 123-136.

[34]. Abbasi, A., Javed, A.R., Chakraborty, C., Nebhen, J., Zehra, W., & Jalil, Z. (2021). ElStream: An Ensemble Learning Approach for Concept Drift Detection in Dynamic Social Big Data Stream Learning. IEEE Access, 9, 66408-66419.

[35]. Gao, Y., Chandra, S., Li, Y., Khan, L., &Thuraisingham, B.M. (2022). SACCOS: A Semi-Supervised Framework for Emerging Class Detection and Concept Drift Adaption Over Data Streams. IEEE Transactions on Knowledge and Data Engineering, 34, 1416-1426.

[36]. Sun, Y., Sun, Y., & Dai, H. (2020). Two-Stage Cost-Sensitive Learning for Data Streams With Concept Drift and Class Imbalance. IEEE Access, 8, 191942-191955.

[37]. Lu, Y., Cheung, Y., & Yan Tang, Y. (2020). Adaptive Chunk-Based Dynamic Weighted Majority for Imbalanced Data Streams With Concept Drift. IEEE Transactions on Neural Networks and Learning Systems, 31, 2764-2778.

[38]. Sun, Y., Li, M., Li, L., Shao, H., & Sun, Y. (2021). Cost-Sensitive Classification for Evolving Data Streams with Concept Drift and Class Imbalance. Computational Intelligence and Neuroscience, 2021.

[39]. Coelho, D.N., Barreto, G.A. A Sparse Online Approach for Streaming Data Classification via Prototype-Based Kernel Models. Neural Process Lett 54, 1679–1706 (2022). https://doi.org/10.1007/s11063-021-10701-9.

[40]. C. Blake, UCI Repository of Machine Learning Databases, 1998, [online] Available: https://www.ics.uci.edu/~mlearn/MLRepository.html.

[41]. https://github.com/alipsgh/data-streams/tree/master/synthetic/led_500_n_0.1

[42]. https://moa.cms.waikato.ac.nz/datasets/2013/