RESEARCH ARTICLE

DOI: 10.54392/irjmt2566

Asian Research Association

INTERNATIONAL RESEARCH JOURNAL OF
MULTIDISCIPLINARY TECHNOVATION

A Systematic-Architectural-Perspective Based Performance Analysis of A-MERIT-C- Dynamic Learning Multitiered Ensemble-Based Real Time Flight Data Analysis

Shailaja B. Jadhav [a, *], D.V. Kodavade [b], Nagaraj V. Dharwadkar [c]

[a] Department of Computer Science Engineering, Marathwada Mitra Mandal's College of Engineering, Pune, India
[b] Department of Computer Science Engineering, DKTE Society's Textile & Engineering Institute, Ichalkaranji, Kolhapur, India
[c] Department of Computer Science, Central University of Karnataka, Kalburgi, Karnataka, India
* Corresponding Author Email: msgshalom@gmail.com

DOI: https://doi.org/10.54392/irjmt2566
Received: 09-01-2025; Revised: 28-09-2025; Accepted: 12-10-2025; Published: 30-10-2025

Abstract: Large-scale data analysis has been the subject of numerous studies recently. In many applications of today's data-intensive world, data is typically brought in continually as data streams. Analytics engines that handle streaming data must be able to react to data that is in motion. Data streams provide special challenges because traditional methods for data mining and machine learning are meant for static information. They are less suited to consider the representative characteristics of data streams and are very less suitable to effectively analyse data that is growing quickly. The authors through this research viz. A-MERIT-C - a dynamic learning multitiered ensemble-based flight real time data analysis system. Through this research authors have presented an active learning dynamic real time data stream analysis model built with self-tuning ensemble learning framework, able to quickly adapt to concepts in near real time streaming data analysis. The conceptual architectural framework illustrated through this research is adaptive to deal with the dynamics related with real time data through the evolving classifier pool (i.e. best performing classifiers get added to classifier pool at every epoch). One more distinguishing characteristic of -A-MERIT-C is instead of using traditional hold out evaluation, it uses prequentially evaluated classifiers. A-MERIT-C's unique features provide significant gains in accuracy, precision, and AUC for streaming data analytics; however, it can also overcome the drawbacks of current algorithms, including concept evolution and feature drift, by using incremental learning and feedback.

Keywords: Data stream learning, Dynamic learning, Adaptive Ensemble, Multitiered architecture, Prequential evaluation

1. Introduction

One of the most important success criteria for modern businesses is the ability to base business decisions on knowledge concealed in stored data. Numerous other spheres of human endeavour exhibit comparable curiosity in investigating novel forms of information [1, 2]. Data typically arrives continually in the form of data streams in many of these applications [3]. Examples that exemplify this , include click log mining, analysis of emotions and sentiments, network analysis, financial data prediction, traffic control, sensor measurement processing, ubiquitous computing, GPS and mobile device monitoring and numerous other applications [4].

The value of data has led to the widespread adoption of data technologies today. While the challenge of handling hundreds of gigabytes of data is in place, it is to be taken into account that data generated several months ago has far less value as compared to that generated within an hour. It implies that more recent the data, of higher value it is [5, 6]. Because they are typically generated in a few seconds, or even less than a millisecond, the real-time data are therefore most important. Real-time data possesses high potential value for business but it is also perishable as it comes with quick expiration time [7]. Potential data analysis systems should realize the enormous value of this data over a typical time window [8], if not it’s of no value and the purpose for which it has been made for (a decision or resulting action) never occurs. Such type of data always comes continuously and with an unmanageable speed, quite quickly; therefore, it is called streaming data [9 and 10]. Thus, Data streaming requires a special attention from researchers’ community. Some of these streaming data examples can be blips in log file, rapidly changing sensor readings, sudden price changes holding immense value, provided that it is alerted in time [1].

Here, the value and importance associated with old data diminishes with time, even if saved for latter processing. It is because of the change, which affects data mining strategies in various ways: target variable change, available feature information change. Changes in the available feature information arise when new features become available, e.g. due to a new capturing IOT device, changed method or sensor instrument. On the other hand, some existing features might need to be ignored or a feature is changed in its scale. These challenges posed by streaming analysis give rise to the extensive research as given in Gama [11] Domingos & Hulten [12] Krawczyk *et.al* [13]. Active learning and Online processing are also of utmost importance since all data cannot be kept [2, 14]. The data analysis with streaming data expects mechanisms to address volatility, the inherent characteristic which corresponds to patterns which are ever-changing with dynamic environment.

The major goal of data stream learning is to aptly use limited resources, and create and maintain an up-to-date model which is ingesting speedy data arriving from a stream data source. As stated in Shan *et al.* [19], Zhai *et al.* [14], a variety of learning frameworks and methods—mostly ensemble-based—are appropriate choices to deal with data streams and get even greater accuracy. There is an extensive need to design and develop an Ensemble Learning Framework which will learn concepts in real time streaming data environments and will address the limitations of existing algorithms such as concept evolution, feature drift, along with improvements in the performance of streaming data analytics. The aim of the proposed research is to design an adaptive classifier ensemble. The Proposed approach is to use a Stacked Ensemble approach with high level goal to analyse real time stream data.

The motive behind this research is to tune the machine learning algorithms for real time data streams. In order to overcome the limitations of existing machine learning techniques, if applied for real time data streams ; a more suitable and relevant method prequential evaluation [15] is used. The research algorithm is designed to handle non-stationary and imbalanced data. This research work concentrates on use of feature engineering, prequential evaluation, Multitiered ensemble learning and classifier pool. Adaptivity of ensemble through self-tunning mechanism is major contribution of the research which distinguishes it from the earlier works in this area. The detailed design of proposed real time data stream classification algorithm named as Adaptive Multitier Ensemble for Real Time (Integrated with Feature Engineering) stream data Classification [A-MER (I) T-C] is built with the abovesaid research objectives.

The remainder of the paper is arranged as follows. A quick know-how of the related work in the similar research area is taken is section 2. Section 3 discusses the main idea behind architecture – design principles, overall system design and the major components. Section 4 is dedicated to the detailed algorithms and pseudocode associated with the major design components. Section 5 illustrates the experimental aspects associated with AMERIT-C, data description and metadata. Following section 6 deals with experimental results and performance analysis as real time stream data classification -flight stream data analysis system. Section 7 summarizes the overall design and experimental observations through concluding remarks. The idea is discussed here through architectural concerns as prime objective and associated part of experimentation; but the system has major scope for further extension so, section 8 deals with future scope and limitations in the existing research design.

2. Background -Literature Study

Since existing approaches were created for static datasets and cannot effectively analyse rapidly increasing amounts of data, data streams present new issues for machine learning and data mining. Supervised classification has drawn the greatest amount of study attention among the various tasks examined in data streams. It is frequently used to address a wide range of real-world issues. The need for novel algorithmic solutions arises from the intrinsic features of stream data, namely its magnitude, speed, and dynamic nature. Because the distribution of the data in motion can change, classifiers specifically designed for data streams must have adaptive capabilities.

Streaming data differs from static data in the following ways:

- ***Boundless amount:*** Since data may be infinite, it is impractical to store all of it for multi-pass processing. Classifiers should operate in situations with restricted resources in order to process this.
- ***High rapidity:*** Data comes in so fast that a real-time reaction is necessary. In these kinds of settings, algorithms that depend on retraining fresh classifiers on the most recent data could not work well.
- ***Concept drifting:*** Data has changing distributions rather than being dispersed independently and identically. While the distribution of the observed variables may fluctuate or stay constant over time, the relationship between the observed variables and the goal variable is anticipated to change in an unpredictable manner. Classifiers must be able to forget the past data and remain in tune with the most recent data in order to be impactful on decision-making.

Mining changing data streams has garnered a lot of research attention currently in an effort to solve the aforementioned characteristics. Until now, most of the proposed autonomous and supervised data stream mining methods have been developed for classification and clustering tasks. Most of them are based on the standard methods for learning in static data [1, 16], with significant modifications to handle data streams. Data stream classification algorithms are based on popular machine learning approaches such as instance-based classifiers, neural networks, Bayesian classifiers, and decision trees [5, 17, and 4].

Because they perform better than depending on strong and stable single learners [18], ensemble-based methods are one of the most popular and often used techniques for data stream classification. Several baseline or primary learners undergo training on the same data in an ensemble, also known as a combination of classifier system, and then their predictions are aggregated. In this manner, the single learners' strengths are merged while their weaknesses are lessened. Furthermore, compared to a single classifier, ensembles are typically more noise-resistant [19, 2]. Enhancing both performance and robustness is the aim of ensemble approaches. The misclassified examples must be diverse for an ensemble to function. Another attribute is its capacity to be readily implemented in practical applications [5].

Ensemble algorithms can be used with drift detection tasks in stream learning and to consider dynamic updates, which include adding classifiers with higher performance and removing classifiers with lower performance [19, 14]. For data stream analysis, the ensemble-based approach—a pooled conclusion based on the opinions of a few chosen specialists—is increasingly common. Numerous combination techniques and heuristic approaches [20] are used on a range of fundamental algorithms to address specific issues. There isn't a general guideline for creating the ensemble that yields the optimal solution for each problem [3]. These approaches do not pre-impose the learners to be utilized as component classifiers, examining the most current papers from [18, 21] in general.

Another effective method for integrating several classifiers is stacked generalization, or stacking. It was first presented by Wolpert [18] and has proven to be highly effective in the classification domain, particularly in cases when the classifiers are only generally accurate in certain regions of the dataset [18]. There are two stages to stacking. Training begins with the base or single learners (level 0). Together with the actual value that those models are meant to forecast, the models' output is gathered into a new dataset [21]. The stacking model learner (level 1), sometimes referred to as the meta-learner, uses this fresh dataset as input before combining it to produce the final output. It is possible to set up a stacking architecture so that the meta-classifier receives the probabilities as input. This approach has the benefit of strengthening inter-level communication because the meta-classifier is aware of the base classifier's confidence [22].

Another notable experimentation with stream data as given by AlQabbany *et. al* (2021) [23] is to investigate concept drift impact using adaptive random forests. This research has given meaningful insights to build an efficient prediction model , accuracy enhanced with the importance of resampling . Three case studies viz. Amazon customer reviews data set, the Hotel Arabic-Reviews data set, and the COVID-19 related tweets from the US president and state governors in the US were used in this research. The research confirms the enhancement of the random forest algorithm in stream learning scenario.

Paim, Enembreck (2025) [24] presented Adaptive Random Tree Ensemble (ARTE), a unique ensemble-based classification technique for data streams. Using a random-sized feature subspace for each ensemble member, online bagging and a classifier selection method for final ensemble voting, the research has yielded into high prediction accuracy. After a thorough experimental analysis, ARTE demonstrated strong prediction performance.

By adaptively learning the dynamic correlation between several streams, En Yu *et al.* (2024) [11] presented Online Boosting Adaptive Learning (OBAL) technique. Extensive experimentation on a number of artificial and actual data streams, shows that OBAL makes noteworthy progress in solving multistream classification issues. In addition to advancing multistream classification, the knowledge gathered from this study offers a promising avenue for future adaptive learning research in a variety of dynamic data contexts.

Samant R. (2022) [7] proposed EBOLE-PS - a boosting ensemble-based strategy which improved classification process accuracy and speeded up the mining process. This work proposed retaining “results of previous stage ” for enhanced accuracy in online boosting ensemble, specially addressing concept drifts. Also, this work contributes to keep track and repetitive testing of incorrectly classified instances, thus results in more rectification. The research shows improved prediction accuracy with weighting strategy of boosting method.

Abbas Sani (2024) [25] the study explores how deep learning models can be used to enhance pattern recognition and tackle big data analytics problems including processing streaming data and managing high-dimensional data. It emphasizes how crucial domain adaptation, active and semi-supervised learning, and the best data sampling techniques are for deep learning models. Also, the work has contributed to how various data mining tasks like feature engineering, pattern identification, and big data analytics are impacted by deep learning. It addresses recurring problems with data selection, semi-supervised learning, and domain adaptability.

3. Research Design

The kernel-backbone for real time data stream analysis is design of A-MERIT-C (Adaptive Multitier Ensemble for Real Time stream data Classification - Integrated with feature engineering). Flight data grabbed with live API from Opensky.org, aviation stack etc. is used for the experiments carried out here. (Section 5.1) It undergoes data preprocessing and flight data is made concise enough. Next step is application of feature engineering in order to focus on potential feature set i. e. only the features or characteristics of flight data which contribute to the target data analysis objective. (for details of feature engineering the readers are requested to refer [6] by the authors) The A-MERIT-C is built with the objective of predicting flights if they are delayed or not. Once feature engineering is accomplished, the chunk of data stream is now ready to analyse with A-MERIT-C. The detailed system architecture for the multi-tiered ensemble learning integrated with feature engineering is as shown in figure.1, where every incoming real time data stream undergoes feature engineering and then it becomes a stable input to A-MERIT-C.

The major research contribution of this research is dynamic selection of component classifiers to be added in classifier pool. On the fly learning of classifiers takes place here, and our model dynamically selects the best performing classifiers every time a new chunk of data is received. Both, correctness of prediction and time taken to learn are taken into account as the decision parameters for adding a classifier to the classifier pool. (Table 3)

3.1 Design Principles

The main motive behind this research is to tune machine learning algorithms for real time data streams. The design principles followed when designing this system framework can be roughly illustrated as under.

1. In order to overcome the limitations of existing machine learning techniques that are less suitable, if applied for real time data streams ; a *more suitable and relevant method prequential evaluation* is used. In dynamic learning scenarios, the prequential approach is a majorly applied choice of estimation procedure. The distinguishing merit of this prequential approach comes from training phase incorporated here. There is no need for distinct holdout set, instead of that, all the instances are used in training and testing , thus making *maximum use of large and varying nature of incoming data instances*, which makes it the most suitable choice for streaming data analysis.
2. Existing machine learning techniques are *enhanced with multiple tiers of heterogenous classifiers* and then combined with a metaclassifier which in turn is responsible for giving final classification result.
3. Before applying the classifier chain in the form of a multi -tiered ensemble, the real time stream data undergoes a *feature engineering approach* wherein, data preprocessing takes place and then a set of optimal features is selected in order to reduce data dimensionality that will result in reducing computational complexity of the algorithms applied thereon.

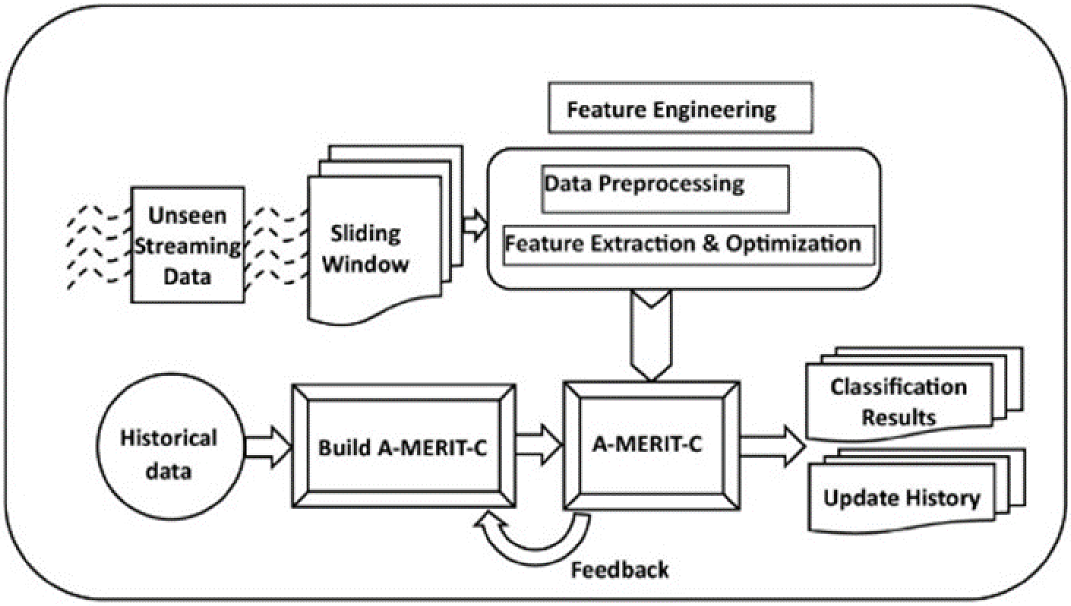

Figure 1. Overall Block diagram

4. The major unique design characteristic of the idea behind the research is getting rid of frequent training of classifiers with every new concept drift of dynamism. The proposed system model of A-MERIT-C tries to overcome this time-consuming task ; by *building a classifier pool* . The classifier pool is kept ready to predict once it receives an incoming chunk of ready to analyse data. Concept drift adaptation is taken care with inclusion of *concept adaptive classifiers such as CVFDT, HAT* etc. as one of the classifiers in a classifier pool.
5. One more adaptive way of handling dynamism in real time data is A-MERITC 's ensemble is not static with fixed set of classifiers. Every time depending on the prequential evaluation of candidate classifiers, only the best performing classifiers are selected to be the part of ensemble.
6. As we are dealing with real time data, the performance decision parameters are, how best the classifier can predict with lesser train time and test time along with the required threshold value of accuracy and precision.

3.2 System Architecture A-MERIT-C

The research idea for data stream classification majorly revolves around the core design of adaptive multitiered ensemble which has been designed keeping in mind the prime requirements of dynamism and concept drift handling through best performing classifiers selected on the fly. These component classifiers are kept as a classifier pool for tier 1 and a separate pool for tier 2. The reason behind keeping a separate set of component classifier has a basis from studied literature [25, 26] and an objective of speedy analysis at the bottom layer followed by a steady and heterogenous classifier set , at the middle layer. The metaclassifier is chosen with the objective of better handling the earlier predictions and simultaneously doing a speedy analysis. The beauty of all this design again relies on a prequential analysis- i. e. predictive sequential analysis strategy which is most suitable choice when dealing with real time stream data analysis [10, 27]. Figure. 2 represents the system architecture and centralised in-depth idea of a A-MERIT-C (Adaptive Multitier Ensemble for Real Time stream data Classification).

3.2.1 Components of Design

The overall design can be divided into sub-modules as given below. The detailed system architecture is given in figure 2.

a. Incoming stream sliding window- ready to analyse (feature engineered and pre-processed)

b. Train / build initial A-MERIT-C

- with the historical data i.e. where the target labels are present

- Right hand side -figure. 2

c. Predict with trained -A-MERIT-C

- For new incoming data stream instance

- Left Hand side -figure. 2

The proposed strategy aims to train a model on the first stream of data received from the source. After that, for every new stream of data received from the source the model will predict the labels for those examples. Based on the prediction, the performance of the model is investigated and the model is updated. The entire process is repeated for every arrival of a new stream of data from the source. At any instant model dynamically selects the best performing classifiers, which are trained previously through prequential evaluation. A sliding window approach is used since it's a real time stream data. A concept drift adaptation is handled through continuous learning and feedback mechanism. Classifiers are updated based on the streams of the data received till that instant. Hence the model adapts to the change in the target concept and thus taking care of concept drift or concept evolution in the data stream.

As shown in figure. 2, the in-depth architecture of A-MERIT-C can be illustrated as modelling the naïve stage with historical (or real time data at T_0) depicted as right-hand side of the figure. and predicting the real time labels using learnt classifiers in building stage, depicted as left-hand side of the figure. Building the model mainly addresses the concept evolution challenge as prequential evaluation approach is focused here, instead of traditional cross validation approach. This approach tries to use hundred percent data for interleaved test-then train (predictive sequential). The model keeps adapting and self-tuning since only the best performing classifiers for a given sliding window of a real time data are always selected and the classifier pool is kept ready. The underlined research uses a multitiered (more than one levels) stacking classifier working on the principle of ensemble learning strategy [28]. There are two tiers with ready set of classifiers waiting for a fresh stream of real time data, built through the building stage of A-MERIT-C. One more distinguishing characteristics is for every next layer, not only the previous predictions are sent but they are also enhanced with the original feature set. As the input is made to stabilise by first passing it through data preprocessing and feature engineering [29] stage, it is also taken care for any feature drift in the data.

4. Design Algorithms

4.1 Notations Used In Algorithms and Figures as Applicable

DS: Data Stream HD: Historical Data,N, n: Tier 1 classifiers, N: Probable set of candidates.

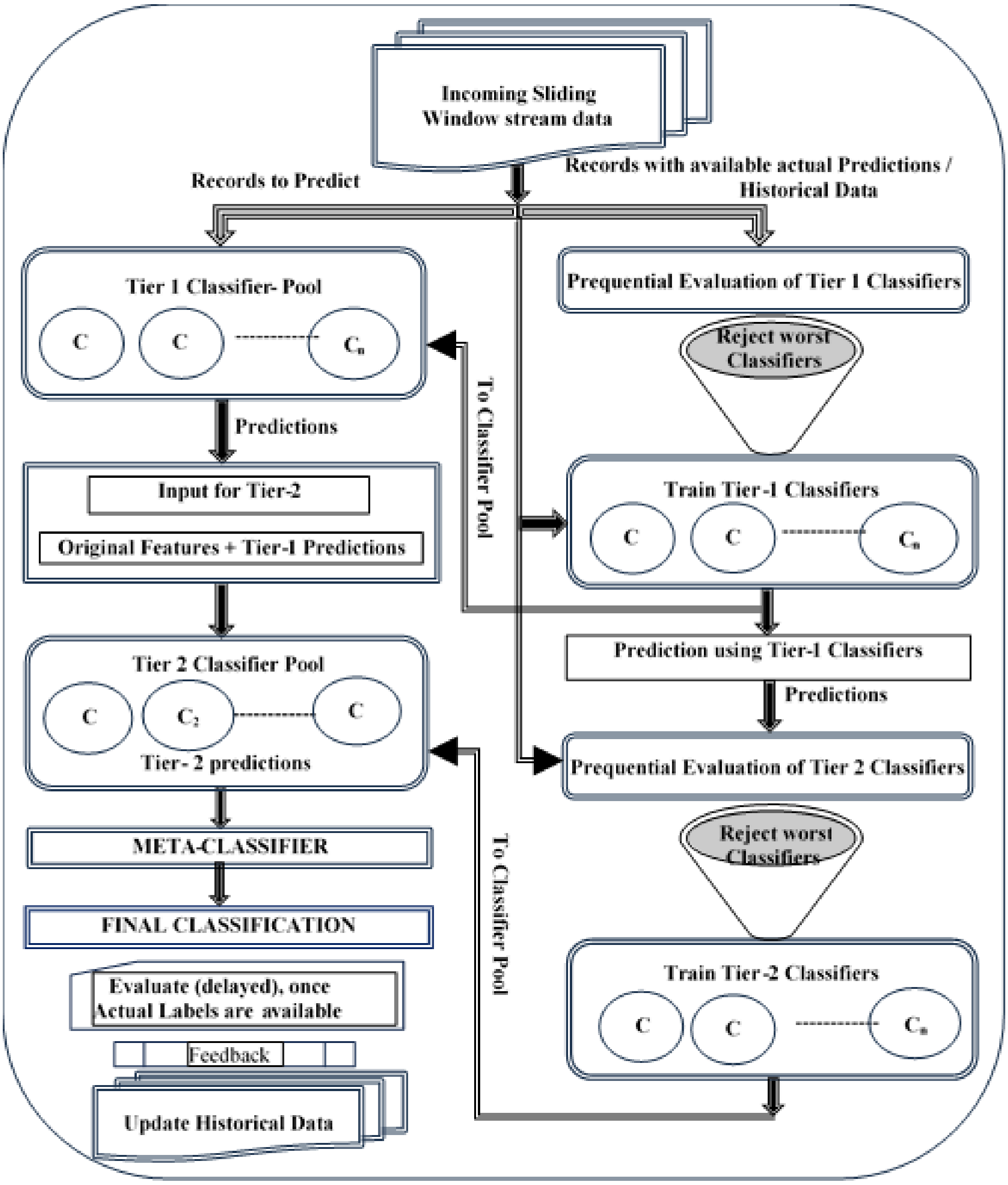

Figure 2. Architecture: Adaptive Multitiered Ensemble for Real Time Classification [A-MER (I) T-C]

N: selected set of candidates,M, m: Tier 2 classifiers, M: Probable set of candidates, m: selected set of candidates,TR_1C: TR_1C_1 to TR_1C_n -Tier 1 Classifiers,TR_1P: TR_1P_1 to TR_1P_n -Tier 1 Predictions

TR_2C: TR_2C_2 to TR_2C_m - Tier 2 Classifiers, TR_2P: TR_2P_1 to TR_2P_m - Tier 2 Predictions

Window: W_1 to Wn (size: 1000 rec.), MetaC, CC-Component Classifiers

4.2 Algorithmic pseudocode

Algorithmic pseudocode

AMERIT-C

In: RTwin-Window of Real time data stream

Out: 1. AMERIT-C Prediction Model 2. Final predictions

1. Data pre-processing and Feature engineering

3. Categorize acc. To classifier label availability

4. Do in Parallel (step 5 & 6)

5. labels not present (delayed labelling)

_AMERIT-C_predict

return predictions

6. with actual class labels available –

ReBuild/Update_AMERIT-C

AMERIT-C_Predict

In: 1 Stream of data DS = {x_i, y_i} I = 1 to ∞

```
2 $\{TR_1C_i\}$ i= 1 n , $\{TR_2C_j\}$ j=1to m} AMERIT-C -trained
Out: FP : predictions-Classification Results
For each Stream of data DS = $\{x_i, y_i\}$ I = 1 to
    For each clsl1 in Tier1
        Predict using clsl1 classifier
    Inputfortier2 -Combine –
        {original features + Tier 1 predictions }
    For each record in { Inputfortier2}
        For each clsl2 in Tier2
            Predict using clsl2 classifier
    final class labels - MetaC
Return FP - Final predictions
```

AMERIT-C -Update/Rebuild

```
In: Stream of data DS = $\{x_i, y_i\}$ I = 1 to $\infty$
Out: $\{\{TR_1, C_i\}$ i =1 to n, $\{TR_2C_j\}$ j= 1 to m}
For i= 1 to n
    Using (x,y) belonging to DS where $y \neq$ null
    $(C_i,P_i)$ = AMERIT_C (PreqEval)Ci
For j = 1 to m
    Using $(X, y_1,y_2....y_n)$
    where (x,y) € DS & $y_1=y_2=......y_n= Y$
    $(C_j,P_j)$ = Prequential evaluation $C_j$
$\{CT_1,CT_2\}$ = AMERIT_C (SelCls_AMERITC) ($\{C_iP_i\}$i=1 to n , $\{C_jP_j\}$j =1 to m
$\{TR_1Ci\}$ i= 1 n , $\{TR_2Cj\}$ j=1to m} Train AMERIT_C(DS,{CT1,CT2})
```

AMERIT-C(Prequential Evaluation)

```
In: 1. Stream of Data DS= {xi, yi} i=1 to infinity
    2. Classifiers selected after preq- evaL{CT1, CT2…}
Out:1.Tier 1 & 2 ready pool of classifiers
For i= 1 to n
    $C_{tmp}$= instance of $TR_1C_i$ trained in past -all (x,y) from DS
        where actual labels are present
    $TR_1C_i$ = Partial fit $C_{tmp}$
For each record (x) from DS if y doesn't exist
    $(x,y_i)$ = predict y value using $TR_1C_i$
    $TR_1out = \{x_k,y_1,y_2,.....y_n\}$ for k=1 to num
For j=1 to m ,$C_{tmp}$ = $TR_2C_j$ trained previously
    $TR_2C_j$ = Partial fit $c_{tmp}$ using $TR_1Out$
Return $\{TR_1C_i\}$ i=1 to n, $\{TR_2C_j\}$ j=1 to m
```

AMERIT-C (SelCls_AMERITC)

```
In:Prequential evaluation results-$\{TR_1C_i\}$ i=1 to n, $\{TR_2C_j\}$ j=1 to m
PTR1= $\{Cname_j, P_i\}$ i from 1 to n, PTR2= $\{Cname_j, P_j\}$ j from 1 to m
Out: $CT_1=\{Cname_i\}$ I 1 to n , $CT_2 = \{Cname_j\}$ j 1to m
$CT_1$ = select best top n by $P_i$ from PTR
If acc(C1)>=thresh && run-time <=min. time
    C1 -> Pool
Else Reject C1
$CT_2$ = select best top m by Pj from PTR2
If acc(C1) is >=thresh && run-time <=min. time
else Reject C1
Return {$CT_1$, CT2……}
```

5. Experimental Works

All the experiments mentioned herein are carried out on the platform scikit multiflow [30], and Anaconda Jupiter platforms as mentioned in Table 1. Following are the minimum requirements to carry out the experiments.

5.1 Metadata and Data Preprocessing- Making It Analysis Reday

This research used a real-time flight data collected from opensky.org [31] It contains data for 583987 flights, with each flight represented by a row in the dataset. The dataset originally had 17 features, which include information such as flight number, origin and destination airport, and flight status. In order to extract more context-sensitive information, we have generated some hierarchical features from the baseline features. Original features are used to create new features that provide additional insights or context. These new features are called extended features. As a result, the dataset used here has 27 features in total, 17 original features and 10 extended features. The extended features are hierarchical features, built on top of the original features. These additional features provide more context-sensitive information and are useful for feature engineering tasks such as feature selection, feature reduction and feature extraction. Throughout the data preparation procedure, unbalanced dataset management tools and categorical data encoding were necessary. All categorical features were transformed into numerical values for accurate modelling. For this, one-hot encoding—which is commonly used for categorical data conversion—was chosen. With this method, each categorical feature's distinct observations were used to construct additional features. An additional data preparation step was necessary for the unbalanced dataset.

Table 1. Experimental Setup

Sr. No.	Parameters	Values
1	Framework	Scikit Multiflow
2	Real Time Stream Data	1 Real time live API from Opensky.org [31] 2 Real time Live API from Aviation Stack [32]
3	CPU	Intel Core i7
4	GPU	A100 from Google Colab , T4 Tesla 16 GDDR6 memory and 2,560 CUDA cores
5	NumPy	1.15.4
6	SciPy	1.1.0
7	Pandas	0.23.4
8	Scikit-Learn	0.20.2
9	Anaconda Navigator	Version 5.3.1; x86_64 bit
10	Jupyter Notebook	Version 5.7.2; 64 bit

Table 2. Data Description taken for experiments

	on-time airline data	Size(#in millions)	#attributes
1	Airline on-time performance data	120	29
2	Opensky-data	2.58	13
3	BOT-US-DATA	5.61	19
4	Aviation-stack	10.68	21

To solve this issue, SMOTE [12], one of the suggested methods, was chosen. By creating synthetic data that is comparable to the minority class, this strategy reduces the disadvantage faced by the minority class.

The pre-processed data is then optimised with feature engineering algorithms- details available in Shailaja , DVK [6] , only the features which are contributing to delay prediction are taken further, rest of the features are not considered. The Live Real time data is grabbed from Open sky and Aviation Stack URL, also the historical data is taken from the link provided via the same API. The historical data is used to train the classifiers and evaluate them prequentially for their possible selection in constructing the heterogeneous ensemble through classifier pool. Every epoch of live data contains about 5k instances of USA geographical area , which is used for testing the system through stacked ensemble classifiers , for prediction of flights delayed or not. In addition to this data, authors of this research have also included following data (Table 2) for experiments in order to check the feasibility of the model and observing models' performance.

5.2 Experiments

We have conducted several experiments with real time data streams for binary classification, to predict flight delays. The scope of this research is limited to only binary classification. Further it can be extended for multiclass classification as well. The major goal of this research is to build the stacked ensemble such that it will find a golden midpoint between the expected accuracy of binary classifiers over a real time continuous data stream and the computing power requirements as well as time & memory requirements simultaneously. This is a very first requirement of any real time stream data analysis system, since the stream data that too if it is live streaming has to undergo data preprocessing, data pipelining and various feature engineering aspects almost on the fly, at least in a single run. Having said this, the system of this type of data analysis should have the prime capacities of giving highest performance consuming as less memory as possible and with very less computing power as far as possible, so that it will not be resource and memory savvy (Table 3).

We have evaluated each of the candidate classifiers for their selection into classifier pool of stacked ensemble layer (viz.1,2,3 etc.) and keeping a threshold value for mean precision and train and test time required for each classifier , our model finally selects the classifiers dynamically. Figure. 3 (a,b) shows the example execution results of a random epoch execution over a part of the sliding window of real time flight data stream.

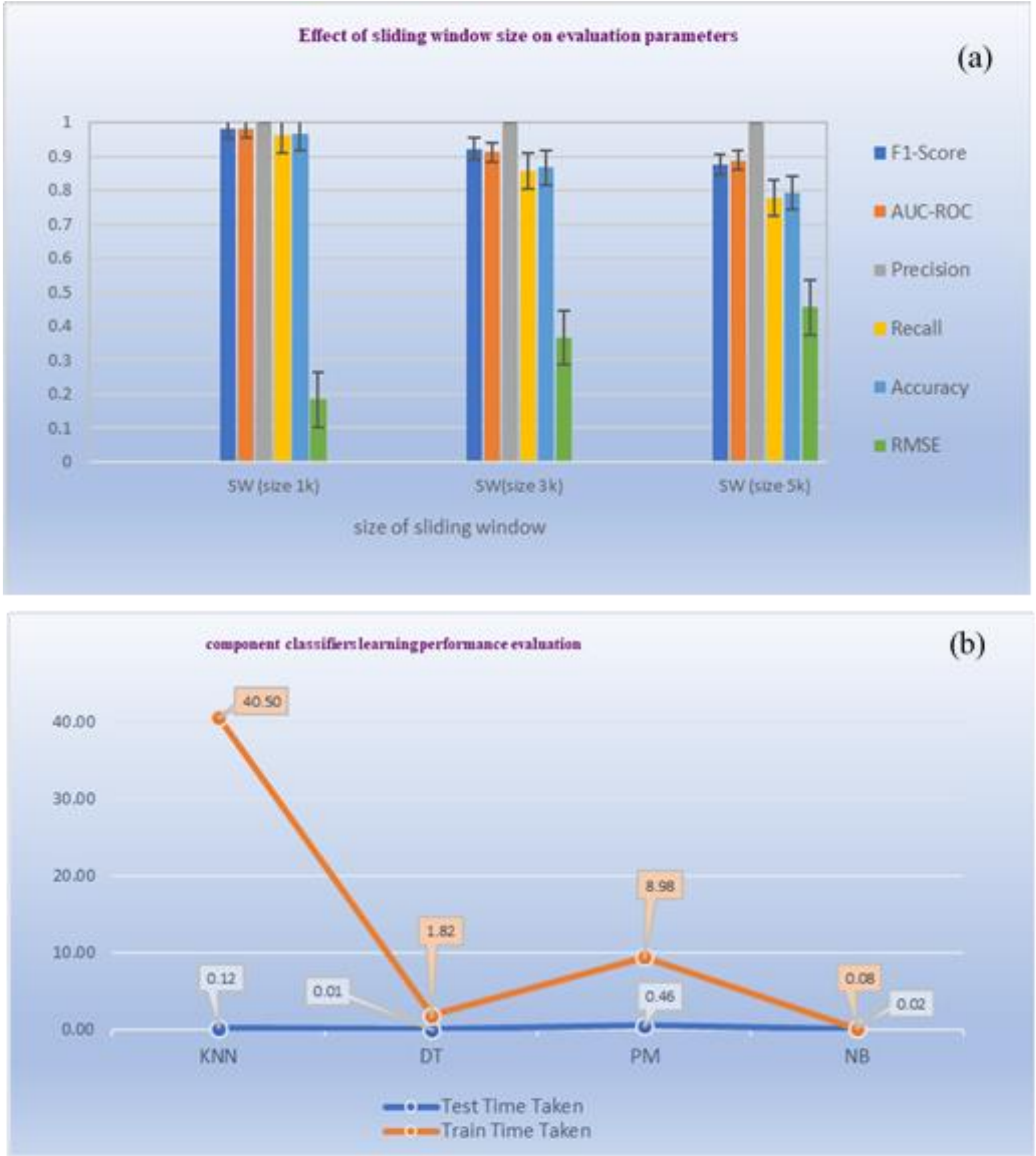

Figure 3 (a) optimum size selection for sliding window, **(b)** Performance evaluation of component classifiers to be in dynamic ensemble

Table 3. Design of Stacked Ensemble – Selection Criteria

Sr.		Component Classifiers	Parameters/Criteria
1	Tier 1	Hoeffding Tree (HT)	Precision, Train_time
		Extremely Fast Hoeffding Tree (EFDT)	
		Naïve Bayes (NB)	
		Hoeffding Adaptive Tree (HAT)	
2	Tier 2	Very Fast Decision Tree (VFDT)	Precision, Train_time
		Very Fast Decision Rules Classifier VFDRC	
		Naïve Bayes (NB)	
3	Final Tier (Metaclassifier)	GB,LR,ARF	Accuracy,AUC-ROC and Quick Response

** hyperparameter setting for the representative classifiers are as mentioned in annexure

The similar process is adopted in order to select a metaclassifier, there are directions received from subsequent literature surveys to go for GB-Gradient Boosting, LR- Logistic regression and RF- Random Forest and some variations thereof. We have selected ARF- Adaptive Random Forest since its performance is undoubtedly good when acting as metaclassifier over a big data stream of dynamic nature [33].

Once the classifiers are trained over a historical data stream, the selected classifiers are added to the classifier pool 1, 2 as per their place and they are used for layer 1, 2 predictions. Layer 2 receives layer1 prediction results for reference plus it does its own prediction, these are given as a combined input to layer 3. Where a metaclassifier is giving a result for prediction. Here, it is Adaptive Random Forest selected as a metaclassifier since its higher precision. Once the component classifiers are added to the classifier pool at tier 1 and 2 respectively, we have tested an incoming sliding window of actual real time flight data, (already underwent data preprocessing and feature engineering as shown in the system architecture - refer fig. 1, and the corresponding data passes through two tiers of ensemble, and then through a metaclassifier before it gives a final classification result for each flight record, whether it belongs to a class of delayed flights (class 1) or to the class of not-delayed flights(class 0). We have conducted the experiments on scikit multiflow and Anaconda Jupiter platforms, (details in table 1) and we have achieved remarkable progress as compared with the similar systems that we found in the literature. Fig. 4 and 5 show the representative evaluation we have achieved for A-MERIT-C's performance over a real time data stream of flights.

6. Results and Performance Analysis

We have experimented with real time stream datasets in order to investigate the performance of component classifiers for their selection into ensemble. Table 4 shows performance analysis of component classifiers. As we are dealing with stream data, we have to found out a golden midpoint between classifiers ability to correctly classify an incoming data instance, predicting accurately in very less time as much as possible .So, the evaluation parameters for classifiers performance are kept as accuracy, kappa and time (Table 4) . Naïve Bayes classifier although with average accuracy gain, it fits for the requirements of base layer [34], hoeffding Tree classifier as mentioned in various data stream related research [5, 33] is a good choice to be one of the classifiers of an ensemble . Also, KNN variant SAMKNN [35] is suitable as it has built in capacity of self-adjusting memory. As we have to design a heterogenous classifiers, the base and middle tier is composed of classifiers which are heterogenous enough, one of the main requirements of ensemble design in order to expect the better performance gain.

The design of multitiered ensemble relies on a metaclassifier at the final tier , so we need to choose a metaclassifier with optimized performance gain , studies from literature through the works of [19, 2, 18, 25] have suggested the probable metaclassifiers as Logistic regression [36], SVM [37]and ARF [38]. We have continued our experimentation for a good choice of a metaclassifier. All these results are obtained at intermediate level of ensemble design, but they have played a vital role and laid a strong foundation in the final multitier ensemble design.

In order to evaluate A-MERIT-S's performance, we as researchers have reviewed a long list of relevant research articles but very few of them have really taken into consideration live real time data. Also, it was irrelevant if at all we could have compared the flight data analysis with that of a generic data stream analysis system.

Table 4. Performance Analysis of component classifiers

Component Classifiers of Ensemble	**Prequential Evaluation**		
	Accuracy	**Kappa**	**Time(Sec)**
Naïve Bayes Classifier	**0.693**	0.1513	**76.76**
SAMKNN - Self-Adjusting Memory KNN	**0.8013**	0.5902	**34.18**
Hoeffding Tree	**0.7767**	0.5394	**11.27**
Extremely Fast Decision Tree	0.7886	0.5668	467.93

Table 5. A-MERIT-C performance evaluation

Sr.	**Metric**	**FLRDAR-24**	**A-MERIT-C**
1	F1	72.85	87.48
2	Precision	71.28	100
3	Accuracy	79.57	79.27
4	Recall	75.25	77.75
5	AUC	87.43	88.889

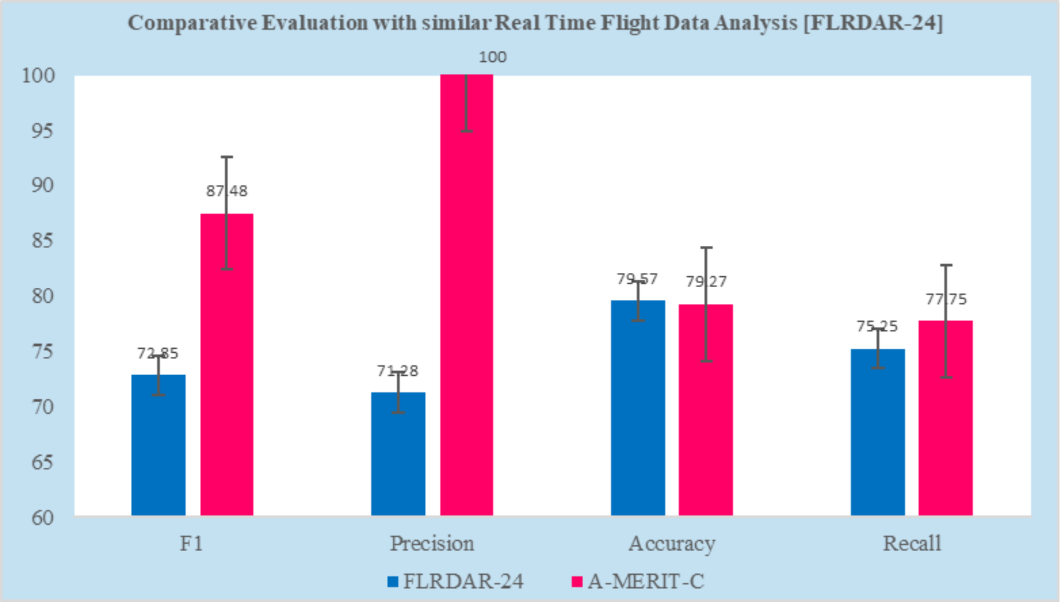

Figure 4. A-MERIT-C performance evaluation with another flight data Analysis system

Table 6. Performance evaluation of – AMERIT-C

Data	Methodology	Mean Acc.(%)	Mean Time (S)
airline-on-time performance data	**Online AUE**	77.06	7.26
	DWM	77.29	7.27
	ACE	79.61	13.00
	A-MERIT-C	80.15	12.50
	LB	68.80	15.25
	GB	76.98	8.5
BOT-US-DATA	**Online AUE**	99.23	30.86
	DWM	95.92	23.46
	ACE	97.42	45.71
	A-MERIT-C	98.11	52.20
	LB	95.64	56.23
	GB	96.56	42.23

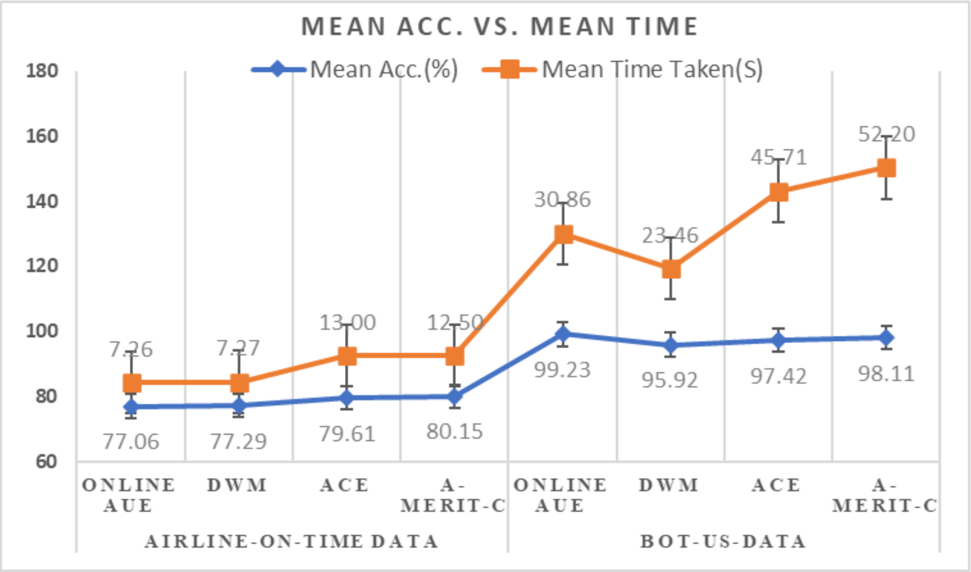

Figure 5. Performance evaluation of A-MERIT-C (Refer data description from table 2)

In keeping the same objective in mind, we found a similar research work in, where Souza et.al. [36] Have worked with live real time flight data through various logistic regression models. After doing comparative performance analysis with that of [36], it is clear that our model A-MERIT-C has achieved considerable gain in evaluation parameters like f1, precision, recall. Also, accuracy of our model is at par with that of [36]. Table 5 shows the example results of this experimentation.

Figure 4 shows the comparative performance analysis where it is clear that model A-MERIT-C has achieved considerable gain in evaluation parameters like f1, precision, recall. Also, Accuracy of the model is at par with that of [36].Aviation stream data analysis is also taken further with other aviation datasets – on time performance data viz. as given in [39] –as described below. A-MERIT-C with prequential evaluation is tested for the data. It has shown the remarkable performance (figure 5) when compared with the benchmarked online active learning ensembles – Online AUE [40], DWM[41, 42], ACE [43] , LB [44] and AdaBoost [45]as shown in the table 6.

too if they are online active learning ensembles , time taken for learning matters most, if accuracy is more. For airline -on time performance data its huge voluminous data approx. 120 million records with approx. 25-30 features, the inherent veracity and volume has limited the accuracy evaluation. On the other hand, Bot-US-Data is balanced enough, so it has taken accuracy to highest level but only at the cost of mean time increased on an average.

This observation is very representative as working with real time data, since one has to find out a golden midpoint between time and accuracy. So, the models should be able to predict in as less time as possible and will be able to produce commendable progress in accuracy. It is also possible to evaluate the system for other performance parameters such as validation loss, MSE, prequential AUC, but the scope for the paper is kept limited only to the accuracy, precision and mean time taken approx. The research work is still continued for the extension.

6.1 Ablation Studies

6.1.1 Impact of Feature Engineering

The authors through [6] have done extensive experimentation on impact of feature engineering for real time stream data. Here, the performance of various classifiers is evaluated using engineered features and then the results are contrasted with those obtained using raw features. The results showed that feature engineering significantly improves the performance of the classifiers and allows for more accurate prediction for stream data. This case study emphasises the significance of feature engineering for real-time data streams in machine learning and its potential to enhance the effectiveness of prediction models.

6.1.2 Impact of multi-tiered classifier

To assess the contribution of each component in our proposed adaptive multilayer ensemble ML model, an ablation study is conducted by systematically removing key components and evaluating the model's performance. The results demonstrate that the full model achieved the highest accuracy of about 78.15 to 80.15% for repetitive epochs. Additionally, omitting classifiers/layers of the ensemble, such as only layer 1 resulted into accuracy of approx. 65 %, layer 1 combined with layer 2 resulted into accuracy of approx. 72.35 %), and adding metaclassifiers with two earlier layers (A-MERIT-C) achieved substantial accuracy gain . This ablation study shows that there is notable accuracy drop if any of the component layer or classifier is dropped from the model design. Hence these findings validate the necessity of each module in maintaining high prediction performance.

7. Conclusion and Discussions

This research is a primitive attempt to address near real time stream data classification task w.r.t. aviation streams. There are some observations which will be definitely helpful for the novice researchers, in the similar area.

1. In case of live real time data, feature engineering and feature optimization play a very vital role in order to prepare data for analysis and to reduce dimensionality, which in turn further leads to minimum computational overhead for data analysis algorithms.
2. Using Stacking with three layers serves better as long as time required for analysis and computational efficiency is considered.
3. Concept Drift Adaptation is widely taken care of and contributes in great extent when dealing with dynamic real time data stream. This is made possible via incrementally learning the algorithms in classifier pool through recent past historical data and always selecting the best performing classifiers to be the part of stacked ensemble learning.
4. Also, metaclassifiers like ARF used here are capable of dealing with drift adaptation. It is an added advantage of A-MERIT-C.
5. Keeping a learnt classifier always ready in a classifier pool leads to reducing the time required to train the classifiers drastically, as decision tress and random forests although yield very good performance, they have high demand for train time. In order to tackle with this issue, A-MERIT-C always keeps the pool of

classifiers, ready to test an incoming data stream on the fly.

6. The research approach is modular so, this can be easily scaled to be applied for other real time domains such as finance and IoT data. Although the experimentation carried out here, limits to only airline data, similar streaming data available in plenty of the domains as real time tweets, geo-political blogging, finance – trading [38] and geographical -geospatial stream for the various purposes can also be explored. The benchmarked data sets and experimental data is very limited so it creates a major constraint that we could not have illustrated with the model. But, nevertheless the experimentation is continued further and will take care of this aspect.
7. In order to evaluate the model for other real time streams, the Pre-processing module and feature engineering module will be of immense help, since these streams are inherently noisy and with redundant data. The application of adaptive classifier ensemble is also explored through the works such as [38, 22, and 10], A-MERIT-C if applied over these domain specific data, will achieve substantial improvements in data analysis.

8. Constraints, limitations and Future scope

It's a very true observation as working with the real time data stream analysis that, there is still scarcity of benchmarked platforms for computation. Although, real time stream analysis is need of the hour, not a plenty of standardised algorithms are being explored by research community; (with very few exceptions). So, the desire of building a real time system concludes with actually building near-real time when facing challenges in data stream analysis [4, 46].

As all real time systems are to be greatly improved with human intervention, same is the case with this research, if added with human intervention-intelligence, this can be of great help. This research lacks this aspect and can be taken further. One more aspect to be noted here is, this research does not take into consideration any weather delays, or catastrophic incidents in aviation stream analysis just to keep the system complexity to the minimum level. This can be taken for further extension of the research.

Further work can be inclusion of GPUs for feature engineering and feature pre-processing. These steps normally are time and resource intensive as far as stream analysis is considered. Also, for creation of pool of classifiers as per the incoming stream data size, scalable computing environment can be an optimized solution. The current work is still in experimental stage where we are experimenting with Google's T4 tesla and V100 GPUs, more sophisticated versions of GPUs for further expansion of the research.

One more prevalent area is as per the stream data volume and velocity, the number of good performing classifiers can be cloned into multiple copies, (miniature deep learning), to deal with scalability issues.

References

[1] D. Leite, I. Škrjanc, F. Gomide. An overview on evolving systems and learning from stream data. Evolving Systems, 11, (2020) 181–198. https://doi.org/10.1007/s12530-020-09334-5

[2] H.M. Gomes, J.P. Barddal, F. Enembreck, A. Bifet, A survey on ensemble learning for data stream classification. ACM Computing Surveys (CSUR), 50(2), (2017) 1-36. https://doi.org/10.1145/3054925

[3] D. Brzezinski, J. Stefanowski, R. Susmaga, I. Szczech, On the Dynamics of Classification Measures for Imbalanced and Streaming Data. In IEEE Transactions on Neural Networks and Learning Systems, 31(8), (2020) 2868-2878. https://doi.org/10.1109/TNNLS.2019.2899061

[4] V.M. Souza, D.M. dos Reis, A.G. Maletzke, G.E. Batista, Challenges in benchmarking stream learning algorithms with real-world data. Data Mining and Knowledge Discovery, 34, (2020) 1805–1858.

[5] Z. Yu, D. Wang, Z. Zhao, C.P. Chen, J. You, H.S. Wong, J. Zhang, Hybrid Incremental Ensemble Learning for Noisy Real-World Data Classification. In IEEE Transactions on Cybernetics, 49(2), (2019) 403-416. https://doi.org/10.1109/TCYB.2017.2774266

[6] M.S.B. Jadhav, D.V. Kodavade, Enhancing Flight Delay Prediction through Feature Engineering in Machine Learning Classifiers: A Real Time Data Streams Case Study. International Journal on Recent and Innovation Trends in Computing and Communication, 11, (2023) 212-218. https://doi.org/10.17762/ijritcc.v11i2s.6064

[7] R.C. Samant, S. Patil, An Enhanced Online Boosting Ensemble Classification Technique to Deal with Data Drift. International Journal of Computing, 21(4), (2022) 435-442. https://doi.org/10.47839/ijc.21.4.2778

[8] A.A. Hassan, T.M. Hassan, Real-Time Big Data Analytics for Data Stream Challenges: An Overview. European Journal of Information Technologies and Computer Science. 2(4), (2022), 1–6. https://doi.org/10.24018/compute.2022.2.4.62

[9] O.R. Amosu, P. Kumar, A. Fadina, Y.M. Ogunsuji, S. Oni, K. Adetula, Harnessing real-time data analytics for strategic customer insights in e-commerce and retail. World Journal of Advanced Research and Reviews, 23(02), 2(024) 880–889. https://doi.org/10.30574/wjarr.2024.23.2.2407

[10] D. Brzezinski, J. Stefanowski, Prequential AUC: properties of the area under the ROC curve for data streams with concept drift. Knowledge Information Systems, 52, (2017) 531–562. https://doi.org/10.1007/s10115-017-1022-8

[11] E. Yu, J. Lu, B. Zhang, G. Zhang, Online boosting adaptive learning under concept drift for multistream classification. Proceedings of the AAAI Conference on Artificial Intelligence, 38(15), (2024) 16522-16530. https://doi.org/10.1609/aaai.v38i15.29590

[12] D. Elreedy, A.F. Atiya, F. Kamalov, A theoretical distribution analysis of synthetic minority oversampling technique (SMOTE) for imbalanced learning. Machine Learning, 113(7), (2024)4903–4923. https://doi.org/10.1007/s10994-022-06296-4

[13] B. Krawczyk, L.L. Minku, J. Gama, J. Stefanowski, M. Woźniak, Ensemble learning for data stream analysis: A survey. Information Fusion, 37, (2017) 132-156. https://doi.org/10.1016/j.inffus.2017.02.004

[14] T. Zhai, Y. Gao, H. Wang, L. Cao, Classification of high-dimensional evolving data streams via a resource-efficient online ensemble. Data Mining and Knowledge Discovery September, 31(5), (2017) 1242–1265. https://doi.org/10.1007/s10618-017-0500-7

[15] X. An, C. Hu, G. Liu, H. Lin, Distributed online gradient boosting on data stream over multi-agent networks. Signal Processing, 189, (2021) 108253. https://doi.org/10.1016/j.sigpro.2021.108253

[16] D. Leite, I. Škrjanc, F. Gomide, An overview on evolving systems and learning from stream data. Evolving Systems, 11, (2020) 181–198. https://doi.org/10.1007/s12530-020-09334-5

[17] L. Rutkowski, M. Jaworski, P. Duda, (2020) Stream Data Mining: Algorithms and Their Probabilistic Properties. Cham: Springer International Publishing, 83-89.

[18] D. Brzezinski, J. Stefanowski, Combining block-based and online methods in learning ensembles from concept drifting data streams. Information Sciences, 265, (2014) 50-67. https://doi.org/10.1016/j.ins.2013.12.011

[19] J. Shan, H. Zhang, W. Liu, Q. Liu, Online active learning ensemble framework for drifted data streams. IEEE transactions on neural networks and learning systems, 30(2), (2018) 486-498.

[20] J.R.B. Junior, M. do Carmo Nicoletti, An iterative boosting-based ensemble for streaming data classification. Information Fusion, 45, (2019) 66-78. https://doi.org/10.1016/j.inffus.2018.01.003

[21] J. Kunnen, M. Duchateau, Z. Van Veldhoven, J. Vanthienen, (2020) Benchmarking Stacking Against Other Heterogeneous Ensembles in Telecom Churn Prediction. 2020 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, Canberra, ACT, Australia. https://doi.org/10.1109/SSCI47803.2020.9308188

[22] N. Liu, H. Gao, Z. Zhao, Y. Hu, L. Duan, A stacked generalization ensemble model for optimization and prediction of the gas well rate of penetration: a case study in Xinjiang. Journal of Petroleum Exploration and Production Technology, 12(6), (2022) 1595-1608. https://doi.org/10.1007/s13202-021-01402-z

[23] A.O. AlQabbany, A.M. Azmi, Measuring the Effectiveness of Adaptive Random Forest for Handling Concept Drift in Big Data Streams. Entropy, 23(7), (2021) 859. https://doi.org/10.3390/e23070859

[24] A.M. Paim, F. Enembreck, Adaptive random tree ensemble for evolving data stream classification. Knowledge-Based Systems, 309, (2025) 112830. https://doi.org/10.1016/j.knosys.2024.112830

[25] A. Sani, B.L. Pal, A.S. Dhabariya, F. Rasheed, A. Shah, U. Haruna, B.S. Mu'az, J. Habu, S. Abbas, B.L. Pal, S. Ajay, Deep Learning Techniques in Data Mining: A Comprehensive Overview. International Journal of Innovative Science and Research Technology, 9(9), (2024) 1254-1270. https://doi.org/10.38124/ijisrt/IJISRT24SEP367

[26] P. Stefanovič, R. Štrimaitis, O. Kurasova, Prediction of flight time deviation for lithuanian airports using supervised machine learning model. Computational intelligence and neuroscience, 2020(1), (2020) 8878681. https://doi.org/10.1155/2020/8878681

[27] J.I.G. Hidalgo, B.I. Maciel, R.S. Barros, Experimenting with prequential variations for data stream learning evaluation. Computational Intelligence, 35(4), (2019) 670-692. https://doi.org/10.1111/coin.12208

[28] B. Shailaja, D. Jadhav, V. Kodavade. Performance analysis of ensemble learning for artificial and real time data streams - Research directions. In AIP Conference Proceedings AIP Publishing LLC, 2917(1), (2023) 060003. https://doi.org/10.1063/5.0175615

[29] E.A.K. Zaman, A. Mohamed, A. Ahmad, Feature selection for online streaming high-dimensional data: A state-of-the-art review. Applied Soft Computing, 127, (2022) 109355.https://doi.org/10.1016/j.asoc.2022.109355

[30] J. Montiel, J. Read, A. Bifet, T. Abdessalem, Scikit-multiflow: A multi-output streaming framework. Journal of Machine Learning Research, 19(72), (2018) 1-5.

[31] M. Schäfer, M. Strohmeier, V. Lenders, I. Martinovic, M. Wilhelm. (2014) Bringing Up OpenSky: A Large-scale ADS-B Sensor Network for Research. In IPSN-14 proceedings of the 13th

IEEE/ACM International Symposium on Information Processing in Sensor Networks (IPSN), IEEE, Berlin, Germany, 83-94. https://doi.org/10.1109/IPSN.2014.6846743

[32] AviationStack: https://aviationstack.com

[33] H.M. Gomes, A. Bifet, J. Read, J.P. Barddal, F. Enembreck, B. Pfharinger, G. Holmes, T. Abdessalem, Adaptive random forests for evolving data stream classification. Machine Learning, 106(9), (2017)1469–1495. https://doi.org/10.1007/s10994-017-5642-8

[34] Y. Yang, G.I. Webb, Discretization for Naive–Bayes learning: managing discretization bias and variance. Machine Learning, 74(1), (2009) 39–74. https://doi.org/10.1007/s10994-008-5083-5

[35] V. Losing, B. Hammer, H. Wersing. (2016) KNN Classifier with Self Adjusting Memory for Heterogeneous Concept Drift. In 2016 IEEE 16th international conference on data mining (ICDM), IEEE, Barcelona, Spain, 291-300. https://doi.org/10.1109/ICDM.2016.0040

[36] A. Aljubairy, W.E. Zhang, A. Shemshadi, A. Mahmood, Q.Z. Sheng, A system for effectively predicting flight delays based on IoT data. Computing, 102(9), (2020) 2025–2048. https://doi.org/10.1007/s00607-020-00794-w

[37] Q. Wang, Z. Luo, J. Huang, Y. Feng, Z. Liu, A Novel Ensemble Method for Imbalanced Data Learning: Bagging of Extrapolation-SMOTE SVM. Computational intelligence and neuroscience, 2017(1), (2017) 1827016. https://doi.org/10.1155/2017/1827016

[38] W. Li, X. Xu, Ensemble learning algorithm - research analysis on the management of financial fraud and violation in listed companies. Decision Making: Applications in Management and Engineering, 6(2), (2023) 722–733. https://doi.org/10.31181/dmame622023785

[39] Airline On-time Performance Data: https://www.kaggle.com/datasets/bulter22/airline-data

[40] Y. Sun, Z. Wang, H. Liu, C. Du, J. Yuan, Online Ensemble Using Adaptive Windowing for Data Streams with Concept Drift. International Journal of Distributed Sensor Networks. 12(5), (2016) 4218973. https://doi.org/10.1155/2016/4218973

[41] J.Z. Kolter, M.A. Maloof, Dynamic Weighted Majority: An Ensemble Method for Drifting Concepts. The Journal of Machine Learning Research, 8, (2007) 2755–2790.

[42] P. Dhaliwal, A. Kumar, P. Chaudhary, An Approach for Concept Drifting Streams: Early Dynamic Weighted Majority. Procedia Computer Science, 167, (2020) 2653-2661. https://doi.org/10.1016/j.procs.2020.03.344

[43] K. Nishida, K. Yamauchi, T. Omori, (2005) ACE: Adaptive Classifiers-Ensemble System for Concept-Drifting Environments. In International workshop on multiple classifier systems, Berlin, Heidelberg. https://doi.org/10.1007/11494683_18

[44] A. Bifet, G. Holmes, B. Pfahringer. (2010). Leveraging bagging for evolving data streams. In Joint European, conference on machine learning and knowledge discovery in databases Berlin, Heidelberg, 6321. https://doi.org/10.1007/978-3-642-15880-3_15

[45] R. Yadu, R. Shukla, A hybrid model integrating Adaboost approach for sentimental analysis of airline tweets. Revue d'Intelligence Artificielle, 36(4), (2022) 519-528. https://doi.org/10.18280/ria.360402

[46] R. Elwell, R. Polikar, Incremental Learning of Concept Drift in Nonstationary Environments. IEEE Trans. on Neural Networks, 22(10), (2011) 1517-1531. https://doi.org/10.1109/TNN.2011.2160459

Authors Contribution Statement

Shailaja B. Jadhav: Conceptualization, Methodology, Writing - Original Draft. D.V. Kodavade: Data Curation, Investigation, Writing - Review & Editing. Nagaraj V. Dharwadkar: Formal Analysis, Visualization, Supervision. All authors have read and agreed to the published version of the manuscript.

Funding

The authors declare that no funds, grants or any other support were received during the preparation of this manuscript.

Competing Interests

The authors declare that there are no conflicts of interest regarding the publication of this manuscript.

Data Availability

The data supporting the findings of this study can be obtained from the corresponding author upon reasonable request.

Has this article screened for similarity?

Yes

About the License