THE DYNAMIC ENSEMBLE GENERATION AND CLASSIFICATION ALGORITHM FOR MINING HIGH SPEED DATA STREAMS

N. SIVAKUMAR[1], Dr. S. ANBU[2]

[1] *Research Scholar, Department of Computer Science and Engineering, St. Peter's Institute of Higher Education and Research [Deemed to be university], Avadi, Chennai-54, sivakumarn002@gmail.com*

[2] *Principal, Department of Computer Science and Engineering, Balaji Institute of Engineering and Technology, Tiruporur, Chennai*

Abstract

The class evolution, the occurrence of class appearance and disappearance, is an important research topic for mining high speed data streams. All previous studies implicitly observe class evolution as a temporary change, which is not true for many real-world problems. This paper concerns the situation where classes emerge or disappear gradually. A class-based ensemble method viz. The CBCE (Class-Based ensemble for Class-Evolution) is projected by keeping a base learner for every class and vigorously updating the base learners with new data, it can quickly adjust to class development. A novel under-sampling method for the base learners is also intended to handle the dynamic class-unbalance problem caused by the gradual evolution of classes. The data-based studies reveal the effectiveness of CBCE in a variety of class evolution state in comparison to existing class evolution adaptation methods.

Keywords: Ensemble Classification, Classification, Data Streams, High speed Data streams, Data mining

1. Introduction

The data stream is usually in huge volume, altering dynamically, possibly countless, and containing multi-dimensional features. The attention towards data stream mining is increasing as regards to its presence a wide range of real-world applications, such as e-commerce, banking, sensor data and telecommunication records. Similar to data mining, data stream mining includes classification, clustering, frequent pattern mining etc. Data arrives in a stream and it is composed of tuples. Suppose if it is not processed immediately, then it is lost forever. Streams need not have the same data rates or data types. The working storage stream is main memory but the problem is which cannot store all the data from the entire stream.

The growth of information technology has introduced different challenges in the problem of data maintenance. In earlier days the data records have been maintained in a static way and the users have followed the same strategy in presenting the data. But modern information technology has opened the gate for the users to represent their data in their own way. For example, any government would collect their data in their own format and force others to use the same one. While maintaining such format which is fluctuating would introduce more challenges to the users as well. The class ensemble approach is discussed, towards the problem of data maintenance. Given a set of data *Ds*, which contains *N* number of samples with *k* number of classes, classifying the *N* number of samples from the data set *Ds* towards *k* number of class is a challenging issue where the *k-l* number of samples does not belong to anyone of the class from *k*. This requires the attention of maintaining newly arrived samples in the class *K+1*. In case of classification, the ensembles have to follow a format and so that it can be classified in an accurate manner.

The dynamic ensembles are one which can be received in any format either it has been available in the set or a new one. Like a tweet from the twitter, the topic and the format of the data may be of any format. However, it must be classified into a class so that it can be maintained to propagate to the other users of the same group or interest. The ensembles may be seasonal or non-seasonal, but still, there will be the certain class of ensembles which has to be maintained in all the periods. The ensemble classes would change over a period of time but the common ensemble classes have to be maintained in all the stages to perform the classification in an accurate manner.

The class-based ensembles are generated from the online data streams based on their type of data or the topic of conversation. For example, the seasonal tweet in a Twitter dataset would vary according to the seasons like festivals. But the ensembles can be generated from the tweets and would be used to perform classification of the tweets. Where a terror attack oriented tweets would occur at any point in time or any time window of the year. So maintaining the tweets in a time window based one is more important and such tweets or the ensembles has to be maintained in all the time.

This research work is introduced a min-max influence based ensemble generation and classification approach. The minimum ensemble influence measure is the value which denotes a minimum quantitative value an ensemble to have in order to be carried to the next time window. It has been computed based on the number of occurrence of the ensemble in the particular time window or the number of time it has been discussed in the particular time window. Similarly, the maximum influence measure is the value which is computed based on the number of occurrence of the ensemble class in any time window. The minimum value has been used for a special class ensemble where the maximum influence measure has been considered for the generic class ensembles.

To improve the performance of classification, this research work presents a dynamic ensemble generation algorithm for the online data streams. An Impact measure based approach is discussed to improve the performance of classification. For each class identified, the method computes the impact measure towards each time window identified. The method classifies the ensemble topics into different classes and for each of them, different impact threshold are used. According to the impact measures and the thresholds, the ensemble will be carried to the future time window. The approach improves the performance of ensemble maintenance and classification performance. With the rapid development of incremental learning and online learning, mining tasks in the context of data stream have been widely studied. Generally, data stream mining refers to the mining tasks that are conducted on a (possibly infinite) sequence of rapidly arriving data records. As the environment where the data are collected may change dynamically, the data distribution may also change accordingly. This phenomenon, referred to as concept drift, is one of the most important challenges in data stream mining.

A data stream mining method is capable of constructing and dynamically updating a model in order to learn dynamic changes of data sharing. For classification problems, concept drift is formally defined as the change of joint distribution of data, i.e., P(x,y). Where x is the feature vector and y is the class label. Over the past, some decades, concept drift has been broadly studied. The majority of the previous works focus on the concept drift caused by the change in the class-conditional probability distribution, i.e., P(x|y). In comparison, class evolution is another factor that facilitates concept drift, has attracted relatively less awareness briefly speaking; class evolution is concerned with certain types of change in the prior probability distribution of classes, i.e., P(y), and usually corresponds to the emergence of a novel class and the disappearance of an outdated class. Class evolution occurs frequently in practice. For example, new topics frequently appear on Twitter and outdated topics are forgotten with time. Besides, old topics, e.g., topics on festivals, may also become popular again. Such phenomena can also be observed from other types of data streams, such as the click-through data of news or advertisements since the interests of clients may change over time. The class progress is also called class-incremental learning or concept evolution. More formally, let C_t denote the set of classes whose prior probability is positive at time stamp t.

The class evolution involves the following forms

- Class emergence represents an example of an unknown class is received at the current time. That is, class c emerges at time ***t*** if $c \notin C_1 \cup C_2 \ldots\ldots\ldots \cup C_{t-1}$ and $c \in C_t$. Such a class is called a novel class.

- Class disappearance describes the situation in which the example of an existing class would not be received in the next time stamp. That is, if class c disappeared at time ***t,*** then $c \in C_{t-1}$ and $c \notin Ct$.

- Class reoccurrence defines the point where a disappeared class recurs later in the data stream. Class c is a recurring class at time t, if $c \in C_1 \cup \ldots\ldots \cup Cd\text{-}1$, $c \notin Cd \cup \ldots\ldots\ldots \cup C_{t-1}$ and $c \in C_t$,

Since the number of classes may change when class evolution happens, the model needs to be adapted not only to capture the distribution of existing classes but also to identify that of the novel classes. At the same time, the effects of disappeared classes need to be removed from the model. Hence, in comparison to the change of category prospective probability, class evolution brings additional challenges to data stream mining. In literature, an amount approaches have been proposed to study class evolution problems. Although they have shown promising performance, they implicitly assume that classes come out or disappear in a transient manner.. In other words, the example generation rate (EGR, i.e., the number of examples generated per-unit-time) of a class switches between two states, i.e., a constant positive value and zero. However, in a real-world scenario, it is more likely that classes evolve in a gradual manner. For example, in an early stage, an event may be discussed by a few participants on Twitter; the topic grows in popularity over a period of time and then eventually fades away from attention. Motivated by this thought, this work investigates the category evolution problem with step by step evolved classes. The gradual evolution of classes refers to the case that classes seem or disappear in a very gradual instead of transient manner, i.e., the EGR changes more smoothly. A novel class-based (CB) ensemble approach, particularly Class-Based ensemble for class Evolution (CBCE), is proposed. In distinction to the above-mentioned existing approaches, which process a data stream in a very chunk-by-chunk manner and build a base learner for every chunk, CBCE maintains a base learner for each category that has ever appeared and updates the base learners whenever a replacement example arrives (i.e., in a very one-pass manner). What is more, a novel under-sampling technique is additionally designed to alter the dynamic category-imbalance problem induced by gradual class evolution.

2. Description of the Problem

The class evolution concerns a special case of concept drift and it will briefly review the typical strategies for dealing with concept drift. Then, it will be preceded by the previous works dedicated to class evolution. A sliding window method stores in memory a number of the most recent examples; the window size can be fixed or variable. The model is updated based on new data, which are stored in the window. Old data, which tend to be affected by concept drift, are forgotten. In the reality of class evolution, during this method is able to adopt a model to class evolution by dropping previous data, it further forgets potentially satisfying information of the non- inflated classes, clearly resulting in a negative impact on the mining performance.

The ensemble method mainly includes chunk-based ensembles, on-line ensembles, and hybrid. A chunk based ensemble constructs each base of operation learner by training it by all of a different chunk of data. A weighted aggregation of the base learners is applied to evaluate the concept drift. In the chunk-based ensemble strategy, class evolution would case the base learners to have diverse sets of classes. Taking class emergence as an example, this would cause the collective votes of the earlier base learners to outweigh the correct votes for the novel class. On-line ensembles, e.g., on-line bagging and boosting, update each base learner in an on-line manner. This scheme would take a long time for class evolution adaptation. Hybrid ensemble methods desire to accompany chunk-based ensembles and online ensembles, so aside have the advantages of both in a single framework. For example, the recently proposed AUE2 algorithm employs each chunk of data to initialize a new base learner and to update all existing ones. Then, base learners are weighted according to their accuracies to adapt to the concept drift.

Considering class emergence, as the base of operation learner is mainly trained by the non-evolved category, the new class is very imbalanced in the existing base learners. Moreover, the examples from the novel classes are not enough in the early stage of continuous class evolution. Hence, it is still difficult to recognize novel class efficiently when class evolution occurs. Apart from the previous strategies, drift detection methods explicitly determine the drift of concept and update the model accordingly. In order to adapt to the new concept, most of these approaches forget any information learned before the detected drift. Similarly to the sliding window strategy, for class evolution, this means that useful information will be forgotten. DDD is a special type of drift detection method that keeps old ensembles while they are useful. However, DDD can only keep old ensembles corresponding to one of the previous concepts.

Therefore, in the case of class evolution, DDD will also forget information when more than one class -evolution behaviour may happen over time. The class reoccurrence in class evolution is relevant to recurrent concept drift, which represents the case where a past concept reoccurs again in the data stream. However, the two cases are substantially different. Recurrent concept means a reoccurred joint distribution for all data, and thus the whole class set involved in the concept also reoccurs. On the other hand, when class reoccurrence happens, the current concept may not be identical to any previous concept since some other classes might have disappeared. Hence, class reoccurrence may not lead to a recurrent concept, and thus might not be handled effectively with existing algorithms for recurrent concept drift.

The previous in research on data stream mining assume class evolution to be the temporary changes of classes, which does not hold for many real-world assumptions. In this work, class evolutions are modeled as a gradual process, i.e., the sizes of classes' increases gradually. The new under-sampling method is designed for handling the dynamic class-imbalance problem caused by gradually evolved classes.

2.1 Data stream

The data stream is the great challenge in the current research where data arrives in a stream. The stream is composed of elements or tuples. If it is not processed immediately, then it is lost forever and this is need not have the same data rates or data types. Stream data working storage is main memory/disk. The main problem is which cannot store all the data from all the streams.

The example of a data stream is Sensor data such as temperature sensor in the ocean and Give the sensor a GPS unit, reporting surface height, image data such as Satellites and Surveillance cameras, Internet and web traffic such as Google use hundred million search queries per day.

2.2 Data stream classification

The conventional categorization techniques perform batch training on compact datasets. However, this batch information is not helpful when applied to completely fluctuated data streams. Most hallowed categories of data stream classification approaches cited in the literature feed.

- Windows-based Approaches
- Weight-based Approaches
- Approaches to Incremental Learning
- Ensemble classifier Approaches

Windows-based Approaches: Window Based Approaches only deal with the recent history of the stream and achieves better classification at resource observant environment where bounded memory and processor time are the major constraints. Fixed sliding window, adaptive window, landmark window, probabilistic relate window and self-adaptive sliding window epitome are preferably popular window-based approaches.

Weight-Based Approaches: In this approaches, weights are assigned to instances of data streams by its debility, which is calculated based on the time of its arrival. Damped window and family of FISH algorithms are the practically widely used weight-based approaches. Damped window model provides a window by the whole of a decay function that assigns a weight to each instance, and lowers the weight exponentially over the time. FISH algorithms or unified Instance Selection algorithm uses time and space information to uphold the training window and the training instances at each time step

Approaches of incremental Learning: Incremental learning is a classification responsibility in which the classifiers learn from dynamic training data which will evolve over time by incrementally revising the model, either by using single classifier approach or ensemble classifier approach. These incremental learning approaches demand less or no access to the previously used training data while preserving the knowledge about historical data and have the ability to learn novel classes.

Ensemble Classifier Approaches: It is abundantly known in the data mining society that there is no single algorithm achieving the excellent accuracy for the most part situations. That is, an algorithm that works well on such or part of datasets may function badly on others. To commit this present, an ensemble of classifiers can be used to produce better results in classifications in analogy to a single classifier. Ensemble classification is the process in which multiple classifiers are combined to solve a particular problem. Ensemble classification methods differ over several dimensions such as choice of the base classifier, handling of the input training data, the combination approach for the outputs of member classifiers, the manner to initialize and adapt the ensemble weights and to retrain the isolated models. Ensemble classification methods can broadly be categorized into homogeneous and heterogeneous ensemble methods. The homogeneous ensemble, each classifier in the ensemble is of uniform type, yet each by the whole of diverse denounce list, training applies and disjuncture centre. In heterogeneous as a whole, each classifier in the ensemble is different and also maintains high diversity.

3. Related Work

3.1 Weighted Combination Based Method

A weighted aggregation of the base learners is applied to manage the concept drift. In the chunk-based ensemble strategy, class adaptation would cause the base learners to have divergent sets of classes. Taking class emergence as an example, this would cause the collective votes of the earlier base learners to darken the correct votes for the new category

3.2 On-line bagging and boosting

Update each base learner in an on-line manner. This step by step diagram would take a long time for class evolution adaptation. The hybrid ensemble technique aims to merge chunk-based ensembles and on-line ensembles, so as to have the reward of both in a single framework.

3.3 AUE2

The algorithm employs each chunk of data to initialize a new base learner and to prepare all existing ones. Then, base learners are biased according to their accuracies to adjust to the concept drift. Considering class emergence, since the base learner is mainly subdued by the non-evolved share, the hot off the fire sector is fully imbalanced in the actual headquarters learners. Moreover, the examples from the new classes are not sufficient in the early stage of gradual class development. Hence, it is likewise difficult to distinguish novel class efficiently when class evolution occurs. Online Ensemble Learning of Data Streams by the whole of Gradually Evolved Classes:

A dressy announcement torrent mining big idea, CBCE, is approaching to direct the class evolution problem in this assumption. CBCE is built up based on the idea of a class-based ensemble. Specifically, CBCE maintains a base learner for each class and updates the headquarters learners no matter when a polished lesson arrives. Moreover, a hot off the fire under-sampling means is made for handling the dynamic class-imbalance problem induced by gradually evolved classes.

3.4 Chunk-based

The ensemble means mainly includes chunk-based ensembles, on-line ensembles, and hybrid ones. A chunk-based ensemble constructs each base first of may by learning by doing it mutually a disparate glaze of data. A weighted aggregation of the headquarters learners is applied to consider the concept drift. In the chunk-based ensemble strategy, class adaptation would cause the base learners to have different sets of classes. Taking class emergence as an example, this would cause the broad votes of the earlier base learners to out-weigh the correct votes for the novel class. On-line ensembles, e.g., on-line bagging and boosting, update each base learner in an on-line manner. This step by step diagram would take a conceive time for class evolution adaptation. Hybrid ensemble methods aim to combine chunk-based ensembles and on-line ensembles, so as to have the advantages of both in a single framework. For concrete illustration, the in a different way proposed AUE2 algorithm employs each chunk of data to initialize a new base learner and to update all existing ones. Then, base learners are weighted through their accuracies to adapt to the concept drift. Taking into consideration, the class emergence, since the base learner is mainly trained by the non-evolved class, the new class is fully imbalanced in the existing base learners. Moreover, the examples from the new classes are not enough in the early stage of continuous class evolution. Hence, it is still difficult to recognize novel class efficiently when class evolution occurs.

3.5 Concept Drift

The class re-count in class evolution is related to recurrent concept drift, which correspond to the case where a past concept reoccurs again in the data stream. However, the two cases are substantially different. Recurrent concept means a reoccurred joint distribution for all data, and thus the whole class set involved in the concept also reoccurs. On the other hand, when class reoccurrence happens, the current concept may not be identical to any previous concept since some other classes might have disappeared. Hence, class reoccurrence may not lead to a recurrent concept, and thus might not be handled effectively with existing algorithms for recurrent concept drift.

4. The Proposed Systems

In this research, the planned approach is promising the performance that implicitly assumes that the classes are rising or disappear in a transient manner. In other words, the instance generation rate (EGR, i.e., the number of examples generated per-unit-time) of a class switches between 2 states, i.e., a relentless positive value and zero. However, in a real-world situation, it's additional probably that classes evolve in a gradual manner. For example, an associate degree early stage, an occasion could also be discussed by a number of participants on Twitter; the subject grows in quality over an amount of time and so eventually fades far from attention. Motivated by this thought, this work investigates the category evolution problem with step by step evolved classes.

The gradual evolution of classes refers to the case that classes seem or disappear during a gradual instead of transient manner, i.e., the EGR changes additional smoothly. a novel class-based (CB) ensemble approach, particularly Class-Based ensemble for sophistication Evolution (CBCE), is proposed. In distinction to the preceding existing approaches, that method an information stream in a chunk-by-chunk manner and build a base learner for every chunk, CBCE maintains a base learner for each category that has ever appeared and updates the base learners whenever a replacement example arrives (i.e., in a one-pass manner).

The classification and adaptive novel class detection of feature-evolving data streams address four such major challenges, namely, infinite length, concept-drift, concept-evolution, and feature-evolution. Since a data stream is hypothetically unlimited in length, it is impractical to store and use all the past data for training. Concept-drift is a common phenomenon in data streams, which occurs as a result of changes in the underlying concepts. Concept- development occurs as a result of novel classes evolving in the stream. Feature-evolution is a frequently occurring process in many streams, such as text streams, in which new features (i.e., words or phrases) appear as the stream progresses. Most of the existing data stream classification methods address only the first two challenges and disregard the latter two.

In this, it proposes an ensemble classification framework, where each classifier is equipped with a novel class detector, to address concept-drift and concept-evolution. To address feature-evolution, it proposes a feature set homogenization technique. In this section, the problem of class evolution adaptation is analyzed first. Then, the new approach, as well as the details of each component will be described. Finally, the approach is analyzed and summarized.

4.1 System Architecture

Our proposed approach, promising performance, they implicitly assume that classes emerge or disappear in a transient manner. In other words, the example generation rate (EGR, i.e., the number of examples generated per-unit-time) of a class switches between two states, i.e., a constant positive value and zero.

However, in a real-world scenario, it is more likely that classes evolve in a gradual manner. For example, in an early stage, an event may be discussed by a few participants on Twitter; the topics grow in popularity over a period of time and then eventually fade away from attention. Motivated by this circumstance, this inquires the class evolution problem with gradually evolved classes. The gradual evolution of classes denoted to the case that classes appear or disappear in a gradual rather than temporary manner, i.e., the EGR changes more smoothly.

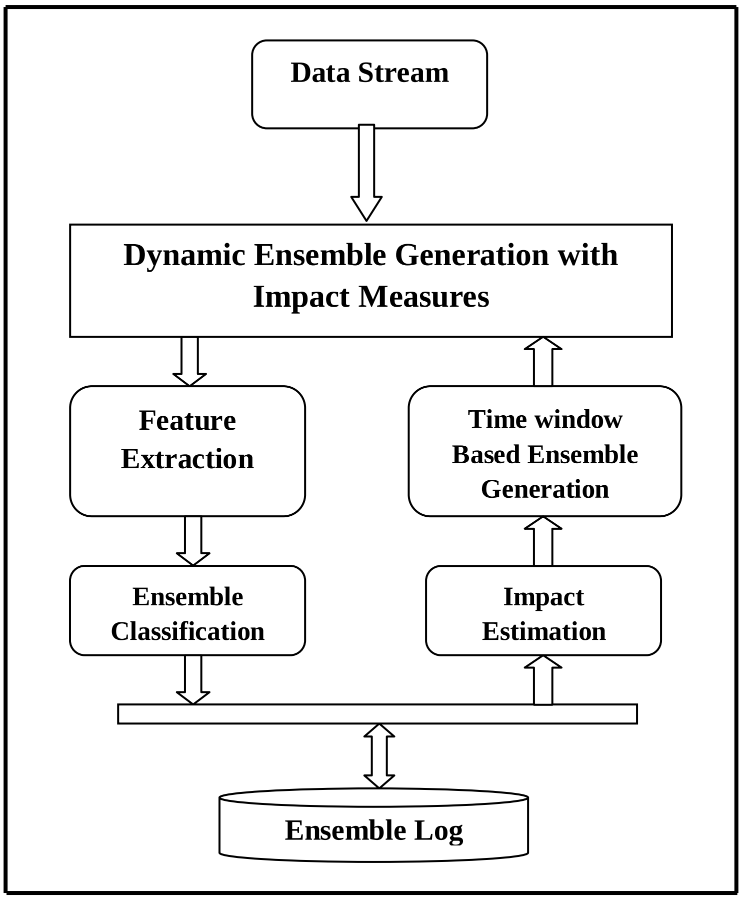
Data Stream
Dynamic Ensemble Generation with Impact Measures
Feature Extraction
Time window Based Ensemble Generation
Ensemble Classification
Impact Estimation
Ensemble Log

Figure 1 Block diagram of Impact measure based classification

A novel class-based (CB) ensemble approach, namely Class-Based ensemble for Class Evolution (CBCE), is proposed. In contrast to the above-mentioned existing approaches, which process a data stream in a chunk-by-chunk manner and build a base learner for each chunk, CBCE maintains a base learner for each class that has ever appeared and updates the bottom learners whenever a new example arrives (i.e., in an exceedingly one-pass manner). Moreover, a unique under-sampling methodology is additionally designed to deal with the dynamic class-imbalance problem evoked by gradual class evolution.

A class-based model is one that is specifically constructed for a certain class to get the likelihood (or related score) of a test example. A variety of models are possible candidates for a CB model, e.g., one-class classifier and clustering model. In this work, the CB model is implemented as a binary classifier that is able to output its classification posterior probability. In each CB model, with the one-versus-all strategy, the represented class is a positive class (+1) and the others are the negative one (-1) as a whole.

The positive and negative classes are likely to be imbalanced in a CB model. Although the class-imbalanced problem has been intensively investigated, most previous studies focus on static class-imbalanced problems. In our case, the prior distribution may change over time, leading to a dynamic class-imbalanced problem. To address this issue, an under-sampling strategy is embedded in each CB model. The sampling probabilities for the positive and negative classes are different. As each CB model acts as an “expert” for its corresponding class, all of the examples received from this positive class are selected. The data size of the negative classes is usually larger than the positive one.

The learning procedure is summarized in algorithm one. Once a replacement example is received, each CB model can update the estimation of the previous probability of its category (lines two and 5). For the class that the presently received example belongs to, its CB model uses it for change directly (line 3). For the alternative CB models, the instance is 1st sampled with the dynamic sampling probability and then used to update the models as a negative training example.

4.2 Problem Statement

In the previous investigations on data stream mining assume class evolution to be the transient changes of classes, which does not hold for many real-world scenarios. In this work the class evolution is modeled as a gradual process, i.e., the size of classes increase or shrinks gradually. The Novel under-sampling method is designed for handling the dynamic class-imbalance problem caused by gradually evolved classes. A failing class might be of less significance than non-evolved or emerging classes in some real-world purposes. In such cases, since CBCE put more emphasis on evolved classes, its performance may decay on non-evolved classes. The algorithm extracts various features from the stream data and such extracted features are converted into stream feature. Generated stream feature has been used to perform ensemble classification

The Eq. (1) for the arrival of complete stream data sd as follows, where ds is data stream and md is metadata and Eq (2) for adding stream Data to stream feature which is described in section 4.4. The Eq. (3) used for calculating the multi-attribute ensemble similarity where sf is the size of feature and sf is stream feature, EC_i represents ensemble class as an output. The Eq. (4) is used to calculate the cumulative MAE and choose the ensemble class with less MAE value using Eq. (5).

$$Sd_i = \sum_{i=1}^{size(md)} Md(i) \in Ds \quad \text{--- (1)}$$

$$Sf = \sum(Sd \in Sf) \cup Sd_I \text{ – (2)}$$

$$MAES = \sum_{i=1}^{size(ECi)} \sum_{j=1}^{size(Sf)} Dist(SF(fj, ECi(j).Fj) \text{ -- (3)}$$

$$MAES/size(Eci) \text{ -- (4)}$$

$$Ec = Min(MAES.Class) \text{ -- (5)}$$

In this, it suggests an ensemble classification structure where each classifier is capable with a new class detector, to address concept-drift and concept-evolution. To address feature-evolution, it proposes a feature set homogenization technique.

4.3 Methodology

The impact measure based ensemble classification algorithm has various stages. They are discussed in detail in this section. From the detailed study of the related works, analyzed that the ensemble generation has the number of factors namely concept drift, concept emergence, ensemble length. This research work addresses all the problems of ensemble generation and for the problem of concept drift, the time window based approach is discussed.

The approach maintains different concept window, where the number of class ensembles is stored. For the problem of emergence, the window adapts the space with modern ensemble class; the length of the window has been a dynamic one to challenge the novel ensembles. Figure 1 shows the block diagram of impact measure based on a classification of ensembles.

4.4 Feature Extraction

The feature extraction is the method used to read the features available from the data stream. The stream would contain different types of and has been identified from its format. For example from the text stream present in the stream, the method extracts the text stream, and the subsequent streams have been identified from the stream according to the format. Extracted features have been used to perform ensemble classification in the next stage.

5. Algorithm

5.1 Pseudo code for Feature Extraction

```
Input: Data Stream Ds

Output: Stream Feature Sf.
Start
        Read data stream Ds.
        Read Meta Data Md = Ds.Meta-Data.
        For each frame Fi from Md
                Extract data from Ds.
                Stream Data(sd) using Eq. (1)
                Add stream Data to stream feature sf. Using Eq. (2)
        End for
Stop
```

The feature extraction algorithm extracts various features from the stream data and such extracted features are converted into stream feature. Generated stream feature has been used to perform ensemble classification.

5.2 Pseudo Code for Ensemble Classification

```
Input: Stream Feature Sf, Time window Ensemble Twe.

Output: Ensemble Class Ec.
        Read stream feature Sf.
        Read Time window Ensemble Twe.
        Compute the size of feature Fs = $\sum Feature \in Sf$
        For each ensemble class $Ec_i$
For each feature $F_i$ of Sf
        Compute data length Dl = size(Sf(Fi)).
                End
                For each feature Ef of class $Ec_i$
                Compute Multi Attribute Ensemble Similarity (MAES) as given in Eq.(3)
                Compute cumulative MAES as given in Eq. (4)
End For
                Choose the ensemble class with less MAES value as given in Eq.(5)
```

The ensemble classification algorithm computes the multi-attribute ensemble similarity with different ensemble classes available and based on the measure estimated a single class has been selected.

5.3 Pseudo Code for Min Max Influence Measure

```
Input: Stream Feature Sf or Ensemble Class Ec,
      Time Window Ensemble Twe.

Output: Min and Max Influence Measures as Mif and Maif.

Start
        Read Ensemble Class Ec.
        Ensemble Class Enc= Perform Ensemble Classification (Ec)
        If Enc.Type==Special

Minimum Influence Measure Mif as

        $Mif = \frac{\sum_{i=1}^{size(Twe)} Twe(i) \in Enc}{size(Twe)}$ ---- (6)
Maximum Influence Measure Maif as
        $Maif = \frac{\sum_{i=1}^{size(Twe)} \frac{\sum Twe(i).Ec==Enc}{size(Twe(i)}}{size(Twe)}$ --- (7)
        Else
                Mif = 0.
        End
        If Enc.Type==Generic
                Minimum Influence Measure Mif as

        $Mif = \frac{\sum_{i=1}^{size(Twe)} \sum Twe(i) \in Enc \; \forall \; Twe}{size(Twe)}$ --- (8)

                Maximum Influence Measure Maif as

        $Maif = \frac{\sum_{i=1}^{size(Twe)} \frac{\sum Twe(i).Ec==Enc}{size(Twe(i)}}{size(Twe)}$ --- (9)

        End
Stop
```

The above-discussed algorithm computes the maximum and minimum influence measure for the ensemble feature given or ensemble class given. The computed measures have been used to generate time window ensemble class to support classification.

6. Experimental Studies

The proposed dynamic time window based ensemble generation algorithm has been implemented using Apache, and its performance in ensemble generation has been validated using the twitter data sets of the year 2015. The method has produced efficient results in all the parameters considered than other methods compared.

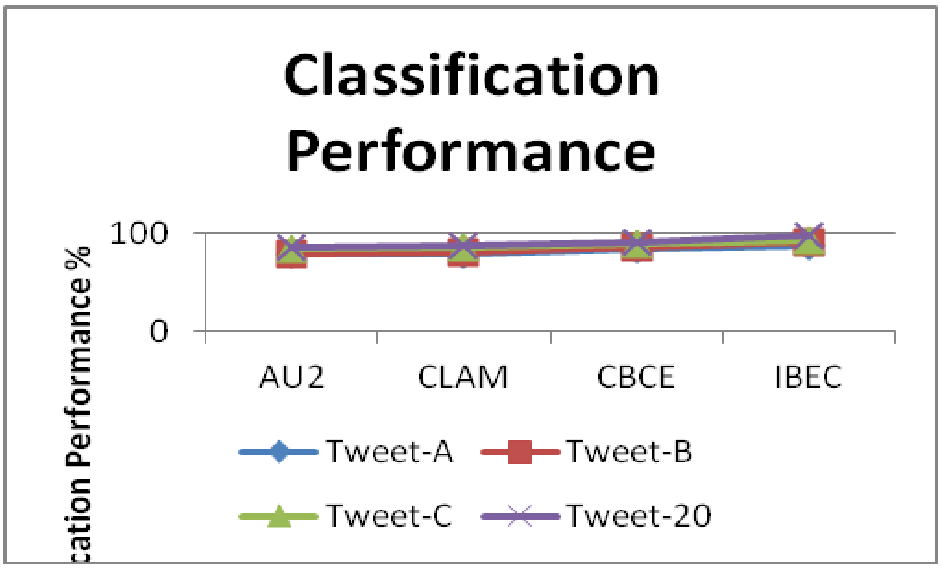

Figure 2 Comparison on classification performance with varying number of classes

Figure 2 presents the result of comparison being performed with varying number of classes. The proposed IBEC algorithm has improved the classification performance up to 97% than other methods in all the number of classes considered.

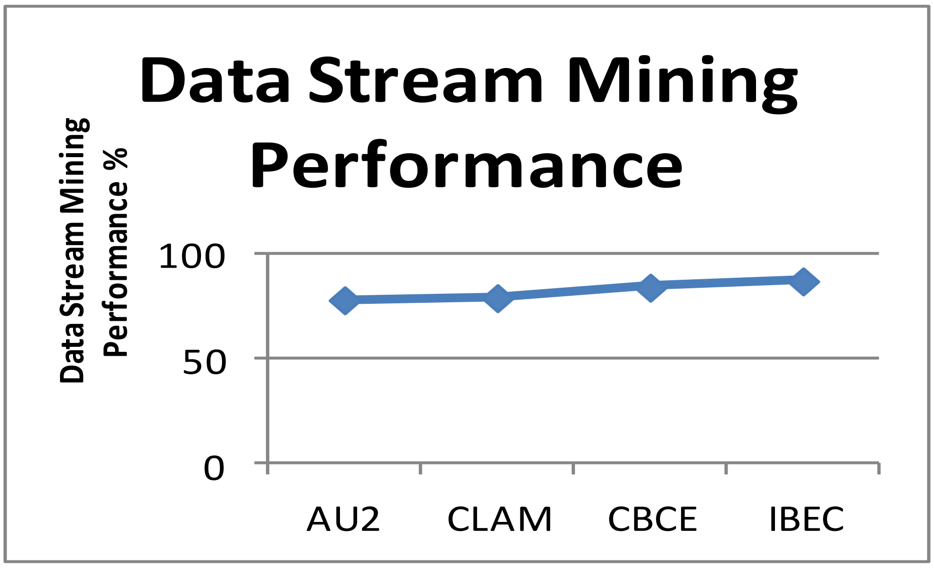

Figure 3 Comparison on data stream mining performance

Figure 3 shows the comparative result on data stream mining performance produced by different approaches. The proposed Min-Max algorithm has produced higher mining performance than others.

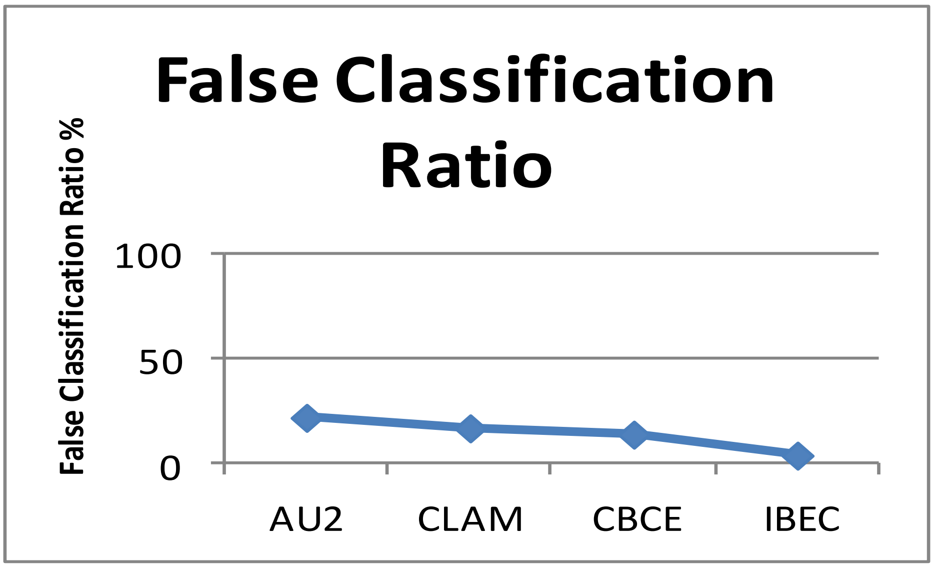

Figure 4 Comparison on false classification ratio

Figure 4 shows the comparative result of false classification ratio produced by different methods. The results show that the proposed approach has produced less false ratio than others.

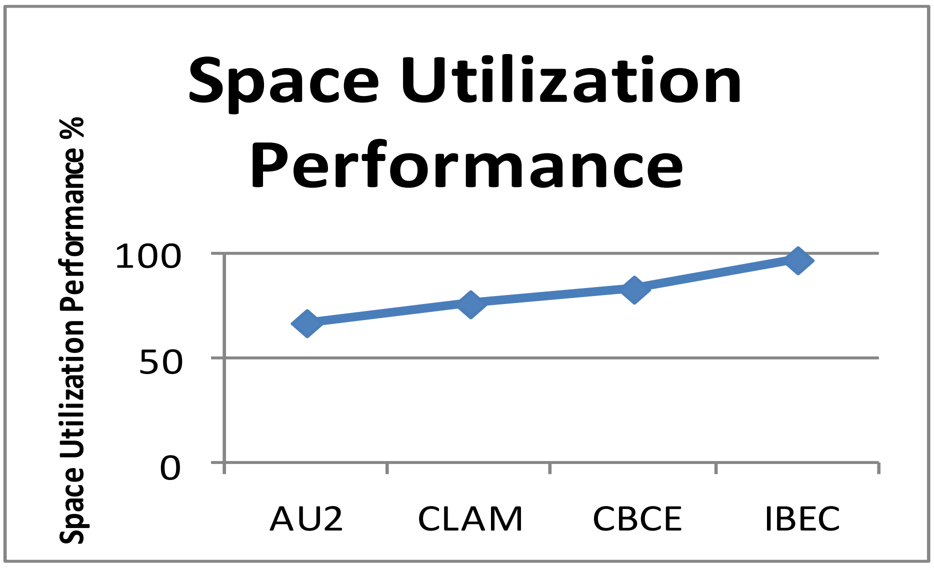

Figure 5 Comparison on space utilization performance

The efficiency of the ensemble generation and classification algorithm is depending on how the space available has been utilized. The method has been evaluated for its efficiency in space utilization. The proposed approach has produced higher utilization performance than other methods compared.

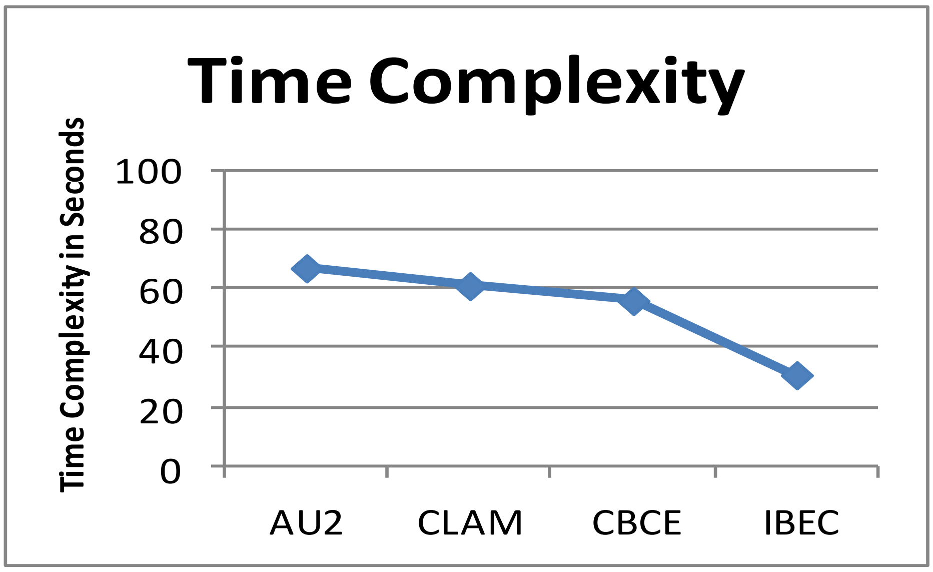

Figure 6 Comparison on time complexity

Figure 6 shows the time complexity being introduced by different methods on ensemble generation and classification. The proposed IBEC approach has reduced the time complexity than other methods.

7. Summary

The above research work was analyzed the high-speed streaming of data can be categorized in the form of classification and clustering can be mainly achieved the performance as well as improve the data security using CBCE. The CBCE has a number of advantages in comparison to the existing methods. First, since a CBCE model is particularly maintained for a definite class, it is flexible to be created or removed to adapt to class evolution. This also decouples the whole model and makes each CBCE model simple and concentrate on a single class. Second, by using the CB model, only a few of base learners need to be maintained, equal to the number of classes. Third, for massive-volume data streams, the master-slave structure (CB model – ensemble strategy) of the learning system is also very convenient for parallelization and distributed implementation.

The algorithm has achieved the high speed streaming of data pre-processing in order to improve the data mining accuracy. It is also verified under different kinds of data sets experimented. This research work analyses a novel Min-Max influence measure based time window ensemble generation algorithm. The method maintains various time window ensemble classes and handled the ensemble generation problem in an efficient manner. For each ensemble, class identified the min-max measure has been estimated to carry or drop the ensemble class to the future time window. This improves the performance of data stream mining performance and classification. The method reduces the space complexity and time complexity compared to other methods.

References

[1] D. Brzezinski and J. Stefanowski, "Reacting to different types of concept drift: The accuracy updated ensemble algorithm," IEEE Trans. Neural Netw. Learning Syst.,vol. 25, no. 1, pp. 81–94, 2014.

[2] J. Gama and P. Kosina, "Recurrent concepts in data streams classification," Know. Inform. Syst., vol. 40, no. 3, pp. 489–507, 2014.

[3] J. Gama, I. _Zliobait_ e, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," ACM Comput. Surv.,vol. 46, no. 4, pp. 44:1–44:37, Mar. 2014.

[4] J. Gao, W. Fan, and J. Han, "On appropriate assumptions to mine data streams: Analysis and practice," in Proc. IEEE Int. Conf. Data Mining (IEEE ICDM), pp. 143–152. 2007.

[5] L. Minku, A. White, and X. Yao, "The impact of diversity on online ensemble learning in the presence of concept drift," IEEE Trans. Know. Data Eng.,vol. 22, no. 5, pp. 730–742, 2010.

[6] M. Karnick, M. Ahiskali, M. Muhlbaier, and R. Polikar, "Learning concept drift in nonstationary environments using an ensemble of classifiers based approach," in Proc. IEEE Int. Joint Conf. Neural Netw., pp. 3455–3462, 2008.

[7] N. Oza, "Online bagging and boosting," in Proc. IEEE Int. Conf. Syst., Man Cybern., vol. 3, pp. 2340–2345, 2005.

[8] P. Domingos and G. Hulten, "Mining high-speed data streams", in Proc. 6th ACM SIGKDD Int. Conf. Know. Discovery Data Mining", pp.71-80, 2000.

[9] S. Ramamurthy and R. Bhatnagar, "Tracking recurrent concept drift in streaming data using ensemble classifiers," in Proc. 6th Int. Conf. Mach. Learning Appl., pp. 404–409, 2007.

[10] W. N. Street and Y. Kim, "A streaming ensemble algorithm (SEA) for large-scale classification," in Proc. 7th ACM SIGKDD Int. Conf. Know. Discovery Data Mining, pp. 377–382, 2001.

[11] X. Zhu, P. Zhang, X. Lin, and Y. Shi, "Active learning from stream data using optimal weight classifier ensemble," IEEE Trans. Syst., Man,Cybern. B, Cybern., vol. 40, no. 6, pp. 1607–1621, Dec. 2010.

[12] Yu Sun, Student Member, IEEE, Ke Tang, Senior Member, IEEE, Leandro L. Minku, Member, IEEE, Shuo Wang, Member, IEEE, and Xin Yao, Fellow, IEEE, "Online Ensemble Learning of Data Streams with Gradually Evolved Classes", IEEE Transactions on knowledge and data Engineering", vol. 28, no. 6, pp.1532-1545, 2016.