DETECTION AND HANDLING OF DIFFERENT TYPES OF CONCEPT DRIFT IN NEWS RECOMMENDATION SYSTEMS

Nayer Wanas[1], Ahmed Farouk[1], Dina Said[1], Nabila Khodeir[1] and Magda Fayek[2]

[1]Informatics Department, Electronics Research Institute, Giza, Egypt
[2]Computer Engineering Department, Cairo University, Giza, Egypt

ABSTRACT

To address the increase in volume of data streams online users interact with, there are a growing number of tools and models to summarize and extract information. These tools use prediction models to personalize and extract useful information. However, data streams are highly prone to the phenomena of concept drift, in which the data distribution changes over time. To maintain the performance level of these models, models should adapt to handle the existence of adrift. In this work, we present the Incremental Knowledge Concept Drift (IKCD) algorithm, an adaptive unsupervised learning algorithm for recommendation systems in news data stream. Data modelling in IKCD uses k-means clustering to determine the occurrence of a drift while avoiding the dependency on the availability of data labels. Once a drift is detected, new retraining data is composed from the old and new concept. IKCD is tested using synthetic and real benchmark datasets from various domains, which demonstrate the different drift types and with different rate of change. Experimental results illustrate an enhanced performance with respect to (a) reducing model sensitivity to noise, (b) reducing model rebuilding frequency up to 50% in case of re-occurring drift and (c) increasing accuracy of the model by about 10% with respect the accuracy of confidence distribution batch detection algorithm.

KEYWORDS

Concept drift, Change Detection, Recommendation systems.

1. INTRODUCTION

Recommendation Systems (RSs) are widely used to help users find items from repositories according to his/her personal interests [1]. This is driven by the patterns of the users in online shopping, news and video content consumption, search, as well as consumption of content and in many different domains [2]. Fundamentally RSs develop a model of the user's interests by observing his/her patterns over a period of time. Modelling approaches could be widely categorized into two categories, namely (i) collaborative based and (ii) content based recommendation [2], [3]. Collaborative based RSs assume that users with the same "taste" would value items similarly. In turn, the collaborative based approaches cluster users with similar tastes into group and generate recommendations from within this group. On the other hand, content-based RSs assume that a user will react to similar content items in the same manner. In turn the content based RSs build user models utilizing features of items that the user favourites in the past.

One of the major issues facing RSs is that the user's interest, and/or the item features themselves may change over time especially for data streams that online users interact with [4]. This problem is referred to in the literature as "concept drift" [2], [4], [5]. Concept drift causes the output recommendations to skew from the current user interest over time and become irrelevant [6]. In turn, the existence of concept drift requires RSs to adapt the user modelling process to the

DOI: 10.5121/ijcsit.2019.11107

changing user interest of item features. News, a major application of RSs, is one of the domains that are highly prone to concept drift [7],[8]. This is due to the fact that news is continuously flowing, and usually short-lived. On the other hand, when the user's interest in certain topics of news change over time, models generated from user history becomes obsolete and no longer generate meaningful recommendations [9], [10].

A data stream is an unbounded and ordered sequence of instances that arrive over time [11] [12]. That leads to a number of difficulties in stream mining since there is no access to all instances in addition to arriving of instances continuously and rapidly. The main target of stream mining is the prediction of the class or value of new instances in the data stream using previous instances. Machine learning techniques especially incremental learning can be used to automatically learn this prediction task from labelled examples. Incremental learning continuously uses input data to further train the model and consequently extend the existing model's knowledge. Such a model has been used for user modelling as it can face the dynamic nature of the student model and consequently for content based recommended systems. However, data streams have a non stationary nature and may be subject to concept drift, in which the data distribution changes over time [4]. The existence of drift reduces the relevance of used classifier (predictor) where data used to train a classifier is not representative for the data a classifier will later encounter. Therefore, it is important that learning algorithms be capable of adjusting to these changes quickly [13][14].

1.1. TYPES OF CONCEPT DRIFT

The literature classifies types of concept drift based on either the time or the predictive views [6][15]. From the point of view of time there are four types of concept drift; namely (i) sudden, (ii) gradual, (iii) incremental, and (iv) re-occurring drift.

1. Sudden Drift: takes place when a concept C1 is abruptly replaced by another concept C2[6],[16],[17]. For example, a researcher working on big data storage and retrieval algorithms is assigned a task to prepare a survey on security issues in big data. This abrupt change in the researcher's interest from storage and retrieval algorithms to security issues represents a sudden drift

2. Gradual Drift: takes place when concept C2 starts growing while C1 is still of interest [6], [16]. However, by the time, C2 continues growing until it becomes the dominant concept while C1 starts decaying until it totally disappears. For example, a researcher who is interested in storage and retrieval algorithms in big data, wants to extend his knowledge to security issues in big data. Over the time, he/she becomes totally interested in security issues in big data and stops reading about data storage and retrieval in big data..

3. Incremental Drift: can be identified only over an extended period of time, because small changes accumulate over time [6], [16]. For example, if members of big data community start discussing security issues of big data rather than storage and retrieval algorithms. By the time, the security issues attract their major attention and the storage and retrieval algorithms are out of their interest. It is worth pointing out that incremental drift demonstrates only one active concept at any time, while on the other hand gradual drift occurs when two concepts are concurrently active.

4. Re-occurring Drift: refers to the case when a previously concept reappeared after some time [6], [14]. For example, a researcher who is interested in big data storage and retrieval algorithms, changes his/her attention towards security issues in big data. However, the researcher discovered that security issues are out of his/her interest, and in turn he/she returns back working on storage and retrieval algorithms.

On the other hand, from the predictive point of view, there are two types of concept drift; namely (i) real drift and (ii) virtual drift.

1. Real Drift: refers to the change in the prior probability of the class [18],[19]. For example, if a student regularly reads sports news, this implies that sports news is within his/her interest. Afterwards, the student, for some reason (maybe in the course of preparing a report), changes his/her interest to the political news. Such a change is called a real drift, as the interest of the student changes although the distribution of both topics is constant.

2. Virtual Drift: refers to the change in the incoming data distribution [6], [18]. For instance, if a user has an interest in sports news. When major sporting event starts, all the sports news focus changes from football, basketball, handball, etc. to the current event. In turn, the distribution of the sports news changed although the user did not change his/her interest.

2. CONCEPT DRIFT HANDLING ALGORITHMS

Handling concept drift is a major challenge for data stream mining. It enables the update of the classification model using recent data representative of the changed concept to maintain it's accuracy. Handling concept drift techniques are categorized broadly into two main groups with respect of how they deal with the problem of recognizing that concept drift has occurred, namely (i) Continuous rebuild approaches, and (ii) triggered rebuild approaches[6],[9], [16]. Continuous rebuild approaches do not monitor the data stream; they build a new model that continuously reflects the current concept. Thus, they have an accurate model through a continuous rebuilding process. However, the main pitfall of such approaches is the processing power required to build the new models. On the other hand, triggered rebuild approaches are based monitoring the data stream through of the value of an indicator to a change in concept. The indicators for changing the concepts are based on using properties of the classifier or using properties of the data or using properties of the classification output [20]. Using an indicator for determining the existence of concept drift maintains the processing power. However, it overlooks using preserved historical model/knowledge while building the new model. In addition, the two groups require the true class of instances (historical labelled data) presented to the classifier for the training process [21], [22]. Labelling instances have a high cost in domains such as news where receiving a continuous stream of news articles need an expert to determine which predefined category each article belongs to.

The most common continuous rebuild approach to handling concept drift is the sliding window. This approch is based on a window containing a labelled set of recent instances as the new training data, which is used to train/update the classifier. Sliding window methods forget previous training instances, which could represent a previous concept. As mall window can generate a sensitive system that reacts quickly to changes. This is associated with a high cost especially in the case of stable data distribution. On the other hand, a large window could create a well-trained classifier that slowly adapts when the concept changes [20], [23], [24].

On the other hand, triggered rebuild approaches monitor the value of an indicator that one usually associated with concept drift. When a defined threshold value is violated a change in concept is flagged. Klinkenberg and Renz [20] classified concept change indicators into three groups according to the used source, (i) using properties of the classifier, (ii) using properties of the data, and (iii) using properties of the classification output.

Using properties of the classifier is based on tracking the internal characteristics of the classifier that is believed to change with the drift. For example, Decision tree leaf changing statistics are

correlated with changes in concept. Hulten et al. [25] propose a tree-based concept drift detection algorithm. This algorithm is called Concept Drift Rule mining Tree (CDR-Tree). CDR-Tree, not only predicts the target class of new incoming instances but also can determine the rule which controls the concept drift. CDR-Tree extracts rules that describe concept drift by integrating old and new data instances together. Thereon, the CDR-Tree is built like traditional decision trees. However, it uses integrated instances in contrast to traditional trees, which uses pure instances. Rules are extracted and concept drift is determined using the difference between the pure data construction and the merged data construction.

The second group of concept drift indicators is based on the data feature values. For example, concept drift in textual data streams can be identified by monitoring word frequencies. Hsiao and Chang presented a cluster-based classification method, called ICBC [26]. It proceeds in two phases. The first phase starts by clustering emails in defined two groups (legitimate and spam emails) into several clusters. In addition, an equal number of features (keywords) are extracted from each group to manifest the features in the minority class. The second phase, incremental learning mechanism is added, which can adapt itself to accommodate the changes of the environment in a fast and low-cost manner. ICBC can effectively deal with the skewed and changing class distributions, and its incremental learning can also minimize the cost of re-training.

The final group of indicators is those derived from the output of a classifier. Bifet and Gavalda [27] propose A Daptive WIN dowing (ADWIN) to handle concept drift in data streams. ADWIN change detection uses the error rate of the model as the concept drift trigger. It uses two windows: (i) a reference window, and (ii) a test window. The reference window is used to calculate the error rate and other parameters when there is no change and the output value is taken as a reference. The test window represents the new data batch and the algorithm monitors its error rate to detect the concept drift. A threshold is used to declare the existence of the drift, and it is calculated based on the average error rate of the reference sub-windows and other parameters. The advantages of using classifier output as a concept change indicator are that it is data independent, and does not pre-suppose knowledge about which feature(s) might be indicative of a change in concept if monitored.

However, these algorithms depend on the availability of labelled data to detect drift and to rebuild the model and cannot handle unlabelled data. To overcome this problem of concept drift detection and model rebuilding in case of high expensive labelled data Lind strom proposes confidence distribution batch detection algorithm (CDBD) [28]. The CDBD algorithm splits the data stream into batches of fixed size and uses a two window paradigm. To detect a concept drift, the algorithm uses the confidence of the classifier outputs. The difference between the current batch confidence and the reference confidence is calculated and compared with a threshold. If the difference is larger than the threshold, the drift is alarmed, otherwise, there is no drift. Although CDBD does not require labelled data to detect the drift, it still depends on the availability of labels to rebuild the model. Kim et al. [29] proposed a concept drift detection method that overcomes the leakage of labels needed for a model rebuilding. He uses clustering to produce virtual classification labels. Generally, the methodology uses two windows, the first window used as a reference window and it represents the concept at the time of building the model, and the second window(detection window) represents the incoming concept of the stream. The concept drift detection mechanism uses a confidence measure that depends on both mean and standard deviation of both reference and detection window. The confidence difference between the reference and the detection is compared to a threshold to decide about the existence of the drift. Support Virtual Machine (SVM) classification algorithm is used as a classifier and K-means algorithm is used as a clustering algorithm. Liu et al proposed nearest neighbour-based density variation identification (NN-DVI) which can detect local concept drift [30]. It consists of three component k -nearest neighbour-based space-partitioning schema (NNPS), distance function and

tailored statistical significance test. NNPS converts data instances from un measurable discrete form into a set of shared subspaces. Distance function uses the output of the previous stage and estimates the density to calculate the differences. The statistical significance test determines the confidence interval to detect the drift. NN-DVI uses two windows of data and compares them to detect drift. The main advantage of NN-DVI is its ability to detect the partition of data that suffers from drift and hence eliminate the need to replace the entire data model. Mello et al. proposed the Plover algorithm, which is an unsupervised concept drift detection algorithm [30]. Plover uses statistical measures functions such as Statistical Moments and the Power Spectrum to detect the drift. The statistical function is applied on two windows; reference windows and current concept window and the divergence between the output of the current window and the output of the reference window is compared relative to a predefined threshold to warn about the drift. Mello has concluded that each data stream has its application domain characteristics and this requires the use of different measures functions to detect the drift.

However, all previous algorithms do not make use of the previous knowledge because they eliminate the previous model and build a new model. Building a new model that depends only on new concept increases the number of model rebuilding, additionally, the model becomes more sensitive to noise.

To overcome these pitfalls, this paper introduces the incremental knowledge concept drift (IKCD) algorithm. IKCD is a trigger based concept drift algorithm based on the data feature values in detecting concept drift. IKCD can maintain model performance through using of preserved historical data in addition to the new data in building the new learning model. Moreover, it totally removes the need for labelled data for ret raining. IKCD also introduces a strategy to handle the news stream problems, namely (i) high dimensionality, (ii) sparsity, (iii) concept drift, (iv) labels shortage, and (v) the raise of temporary events[22], [31].

3. INCREMENTAL KNOWLEDGE CONCEPT DRIFT (IKCD)

This research proposes the Incremental Knowledge Concept Drift (IKCD) algorithm, which is an unsupervised concept drift detection algorithm that uses a trigger as a concept drift indicator. IKCD addresses the problems of (i) depending on labelled data to rebuild the model, and (ii) removing the old knowledge completely, and (iii) long delays prior to rebuilding the new model. The IKCD algorithm uses unlabelled data to build the data model and use prior knowledge in building the new model. IKCD is specifically designed to deal with the concept drift in the news stream. It aims at enhancing the ability of the model to deal with news temporary events, minimizing the time of model rebuilding, maximizing the accuracy of the model, and maximizing the model resistance to noise.

The main contributions of the proposed IKCD algorithm are using unlabelled data to build the learning model in addition to utilizing preserved historical data in adapting that model. IKCD basically includes two phases, (i) the training phase and the (ii) adaptive learning phase, as shown in the Figure 1.The training phase is activated when the drift indicator triggers the existence of a drift. The adaptive learning phase is activated after the model rebuilding is complete and lasts until the arrival of a subsequent trigger. The following sections elaborate more on the main components of IKCD.

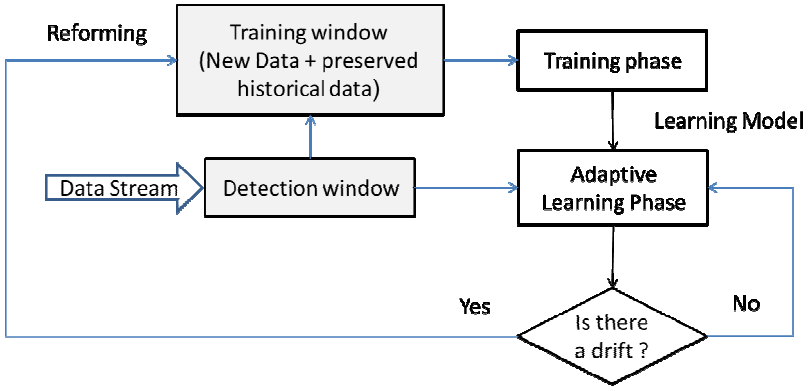

Figure 1: The structure of IKCD algorithm.

3.1 IKCD TRAINING PHASE

Data is segmented into a window with a fixed number of instances taken from time interval..This window is called the training window, since it is utilized in the retraining process of the classifier and formulated in the learning process. The training phase starts by dimensionality reduction of the data stream and selection of the most represented features from text instances that are implied in the retraining window. Then clustering technique is applied and the resulting clusters are used in the training process of the learner (classifier). Figure 2 indicates the structure of the training phase

3.1.1 FEATURE EXTRACTION:

The textual nature of news data streams is exploited in extracting the most important features using Natural languages Processing (NLP) techniques. IKCD starts by applying news stream to Stanford Tagger, which defines the types of each word [32]. IKCD filters the tagged stream and is limited to use nouns. Nouns carry the most valuable information regarding news items. Then stemmer is utilized to extract the stem (base word) of each noun. Next, IKCD uses Term Frequency (TF) to represent nouns and form vectors. In the feature selection step, IKCD selects the most important features, based its term frequency. Then features are arranged based on TF since we assume that it is representative of its importance. The top α (=15%)of the features are used to represent the data distribution. The value of α is selected using a comparative experiment to select the ratio of nouns that should be used as features. This includes considering both the clustering accuracy and performance

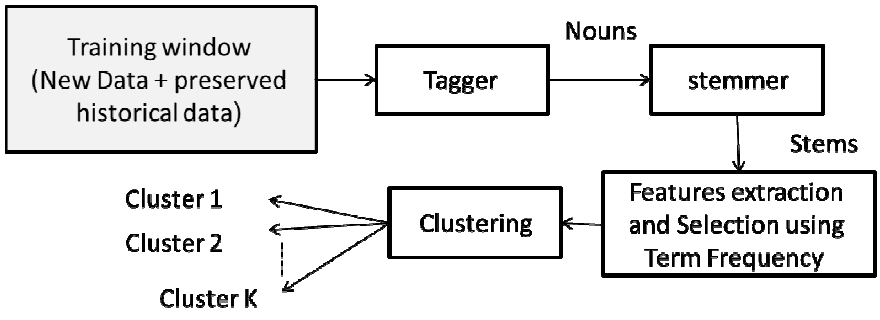

Figure 2: Training Phase

3.1.2 CLUSTERING

One of the simplest and effective point assignment clustering algorithms is the *K*-means algorithm. *K*-means isa clustering process, which aims to partition n instances into *K* clusters. Each instance is assigned to the nearest cluster based on the distance between this instance and the centroid of that cluster.*K*-means assumes a fixed number of clusters; this implies that it is suitable for well-defined data. The main challenges of the *K*-means approach are (i) determining the number of clusters, (ii) determining the initial centroids, which are used by the algorithm to initiate every cluster. The process of assigning values to these centroids is very critical for the operation of the *K*-means algorithm since different values for initial centroids will lead to different results. In turn, it is important to make sure that initial centroids are diverse enough from each other to achieve good clustering. *K*-means follows a sequence of steps. In the first step, each instance is assigned to a cluster based on its distance from the initial centroids. Next, *K* centroids are calculated for the previously clustered instance. Based on these *K* new centroids, all data instances are re-clustered, and again, new centroids are calculated. For each iteration, the centroids are changed. When no more changes are required, or the maximum number of iterations is reached, clustering is terminated. The IKCD algorithm uses *K*-means in its training phase for its effectiveness and performance. However, it is worth mentioning that any clustering algorithm can be used in this regard. The clustering number *K* is selected as a constant number based on the natureof the data applied, while initial centroids are selected randomly from the dataset. Each cluster is represented by its centroid and this centroid is known as reference centroid. Using clustering enables the IKCD to operate using unlabelled data in contrast to other approaches which depend on labelled data for training.

3.1.3 TRAINING CLASSIFIER.

Each identified cluster represents a certain topic of news i.e. sports, political, etc. Such clusters are used as an alternative to the labelled data in the training process. The training produces a learning model that can classify the received data into different classes where each one is corresponding to a specific reference cluster.

3.2 I KCD ADAPTIVE LEARNING PHASE

The goal of the adaptive learning phase is to add the incremental learning ability to the IKCD technique. It detects the drift, and formulates retraining window to be used in the training phase.

3.2.1 DRIFT DETECTION PROCESS.

A new detection window is used, containing the most recently received data. The classifier uses the predefined learning model to classify the detection window data. Then, the centroid of each class is compared with respect to the centroid of the equivalent reference cluster. The distance between the two centroids is measured using cosine distance. If the distance is larger than a threshold, the drift is triggered as shown in Figure 3.

Distance measure:

IKCD uses the cosine distance (defined in Equation 1) to measure the distance between instances and the centroids of clusters in the clustering process. While there are different distance metrics that measure similarity, there is enough evidence in the literature to indicate that the cosine distance is (i) suitable for sparse data, and (ii) effective for text data [33], [34].

$$\cos(\theta) = \frac{A.B}{||A||||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (1)$$

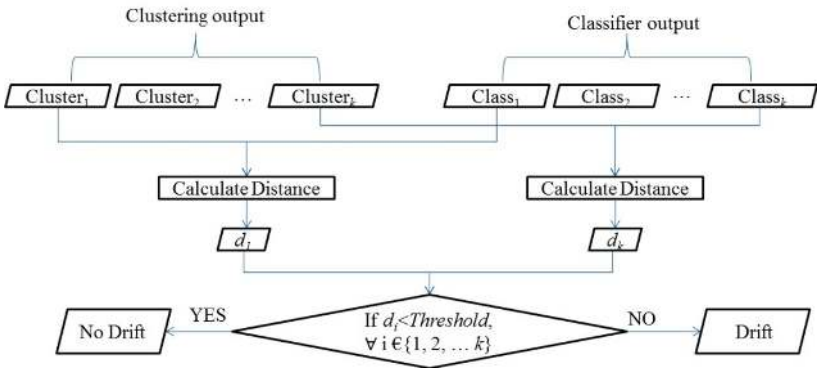

Figure 3: The structure of drift detection process

Threshold selection:

The threshold is a value that represents a decision boundary used to determine the existence of the concept drift. A large value for the threshold leads to model stability since it will demonstrate a large resistance to noise. However, this may lead to a high rate of undetected drifts in the data stream [false negatives].On the other hand, small values for the threshold would provide a higher probability to detect all drifts. However, it would increase the noise sensitivity and false alarms [false positives].In this work we conduct a comparative study to assess the performance relative to the different values of threshold.

3.2.2 RETRAIN WINDOW REFORMATIONS.

The retrain data is used to build the new model following the detection of the drift. IKCD is intended to be designed in a manner that the rebuilt model is able to handle the new concept, in addition to retaining the ability to handle the old. To do that, IKCD composes retraining data that consist of two partitions, (i) historical data, and (ii) the current data. The historical data partition represents the summary of the old concept which is the data in the previous retrain window. On the other hand, the current data partition represents the new concept which is the most recently received data. However, the ratio of the size of the new data to the old data is a parameter for this approach. If the old knowledge represents the majority of training data, the model performance poorly with the new concept and vice versa. IKCD uses a comparative study to consider the effect of the different values of mixing ratio between old and new concepts on the overall performance. It is worth noting that *p%* of the training is selected from the historical data while *1-p%* is selected from the new data.

3.2.3 SENSITIVITY TO THE RETRAIN WINDOW SIZE.

The window size selection is a fundamental and challenging process because it affects the (i) model stability, (ii) memory requirements, (iii) noise sensitivity, and (iv) adaptation speed of the drift detection process. The smaller the window size, the faster it is able to reflect the current distribution status, and the faster its potential to adapt to concept changes. However, in more stable phases, a small window can affect the learner performance [35] making it more sensitive to noise and reducing its accuracy. While a large window would produce good and stable learning,

it, however, cannot react quickly to concept changes [35]. The window size can be either (i) adaptive, or (ii) fixed. An adaptive window size stores a variable number of data in stances to train the model and detect the concept drift. It decreases the window size whenever there is drift and increases it otherwise. ADWIN [27] is one of the most common algorithms that utilizes adaptive window. The fixed window size techniques, on the other hand, use a fixed number of data instances to detect the drift and train the model. DDM [36] and EDDM [27] are examples of the most common algorithms that use a fixed window, where the size of the window is large enough to build a stable model with the best accuracy. The IKCD conducts a comparative study to assess the performance of the algorithm against different window sizes. It is worth noting that a fixed window size approach allows for the discovery of the effect of mixing between old and new knowledge and its effect on the accuracy of the model. The initial window size is selected to represent 10% from the total size of data

3.3 EVALUATION METRICS

In this work, we are interested in measuring the model accuracy as well as the efficiency of the concept drift detection technique. With respect to the model evaluation, the model accuracy overtime is used. To evaluate the concept drift detection technique this research uses F-measure, model retrain number and weighted model retrain number [37]. Furthermore, some measures are related to others. For example, the computational cost is proportional to the model retrain number where more retrain number requires more processing power. In addition, the memory usage is proportional to the training window size and the detecting window size. The maximum required memory size is reached when the system is collecting the new training data. In this case, the total required memory is twice the size of the training window.

3.3.1 EVALUATION OF THE PROPOSED CONCEPT DRIFT DETECTION AND HANDLING TECHNIQUE.

The concept drift detection evaluation is tightly coupled with the overall performance of the complete systems. Examples for system evaluation metrics are accuracy, precision, recall, and error rate of the system [10]. This research also uses the F-measure, which is a statistical test that considers the recall and the precision.

- **F-measure**

  Recall is defined as the percentage of the retrieved relevant instances with respect to the total number of relevant instances as shown in Equation 2

  Additionally, precision is defined as the ratio between the relevant items retrieved and total number of retrieved items as shown in Equation 3.

$$precision = \frac{True\ Positive}{True\ Posotive + False\ Positive} \tag{3}$$

  The F-measure is defined by Equation 4

$$F_\beta = (1 + \beta^2).\frac{precision\ .recall}{(\beta^2.precision) + recall} \tag{4}$$

Where β defines the relative weight of recall and precision. If $\beta < 1$, F-measure favors recall over precision, while if $\beta > 1$, it favors precision over recall. If $\beta = 1$, it means that both recall and precision have the same weight. In this work, we use the F_1-measure.
It is worth noting that these metrics are interested with the detection of the drift, and not on the performance of the classifier/clustering algorithms themselves. They measure the ability of IKCD to detect the drifts, which is a fundamental requirement for the overall performance of the system.

- **The weighted model retrain number (WR)**

  WR represents the normalized ratio between the total number of model rebuilding andthe number of real drifts in the data as shown in Equation 4.

3.3.2 MODEL ACCURACY.

Model accuracy (A) represents the ratio between the correctly predicted instances labels and the total number of instances. Equation 5 shows the accuracy calculation

$$Accuracy\ (A) = \frac{True\ Positive + True\ Negative}{Total\ Population}. \quad (5)$$

4. EXPERIMENTS AND RESULTS

We conduct two main experiments; First usingthe CDBD dataset which is limited to the re-occurring drift. The target of the first experiment is to compare the proposed concept drift detection and handling technique to CDBD and sliding window algorithms. The second experiment considers the different concept drift types using proposed new dataset.

4.1 CDBD DATASET

In this section, we conduct four experiments to investigate the effect of reforming of the training window (the ratio of the new data to the old one), the threshold value (sensitivity to concept drift) onthe learning model accuracy. In addition, accuracy comparison is conducted among the proposed IKCD algorithm; CDBD algorithm and sliding window Continuous rebuild technique.

4.1.1 DATA SET

In this work we use the same dataset proposed by Lindstrom et al. [28] in the evaluation process. The CDBD dataset is a drift induced dataset, which is composed of the 20-newsgroups. Table 1 indicates the structure of the data, which defines a binary classification problem, and forms four blocks of data. The first block is used to train the model. The second and the fourth block are test data that does not contain drifts, while the third contains a drift.

Table 1: Synthetic drift generation from 20 Newsgroups data [28]

	Interval	Target Topic	Relevant	Non-relevant
Training	Comp.*	300	150	150
C1	Comp.*	2000	1000	1000
C2	Rec.*	3800	1900	1900
C3	Comp.*	2900	1450	1450

4.1.2 EXPERIMENT I

We begin with the effect of the drift on the accuracy of the proposed learning model compared to the learning model CDBD, with the adaptation (incremental learning) not being considered in both models. Figure 4shows the accuracy over time of CDBD and IKCD algorithms with respect to window number. Both models maintain an average accuracy of 80% when the data is similar to the data learned, while this accuracy decreases to approximately 60% when drift occurs. The accuracy of both models returns to the original value when old data appears.

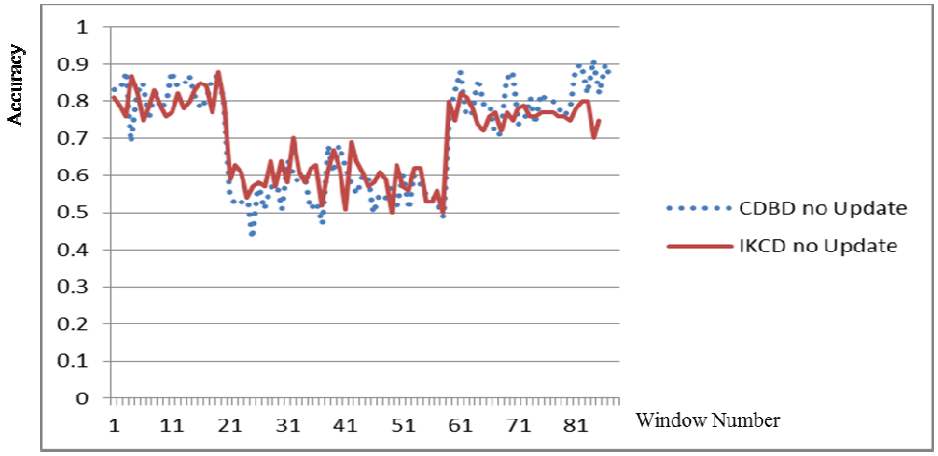

Figure 4: Accuracy overtime of CDBD and IKCD learning models without adaptation

4.1.3 EXPERIMENT II

In the second experiment, we establish the threshold used to detect the drift at 0.1 which is a small value that increases the sensitivity of the model to any change and then adapts itself. At this value, we study the effect of the training window mixing ratio of new and old data on the accuracy of the model and weighted retrain number. As shown in Figure 5, 70% of the new data is given high accuracy with lower weighted retrain number. When the IKCD uses 70% from the new data and 0.1 as a threshold, its accuracy outperforms both the CDBD and the sliding window algorithms as Figure 6shows. However, the sliding window algorithm uses the new data only to rebuild the model. It is worth noting that at this small value of threshold the cost of rebuilding the model is very high and cannot be neglected.

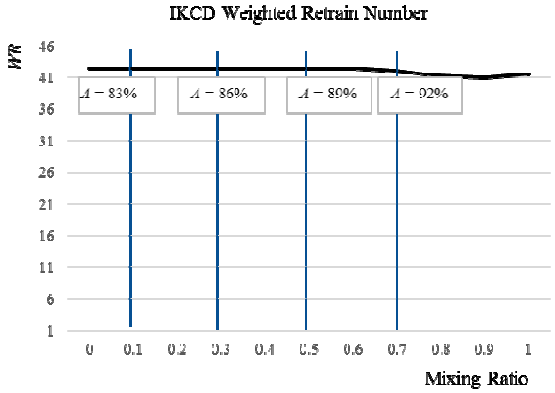

Figure 5: Model weighted retrain-number (WR) and Accuracy(A) of the IKCD and using threshold value (th) = 0.1

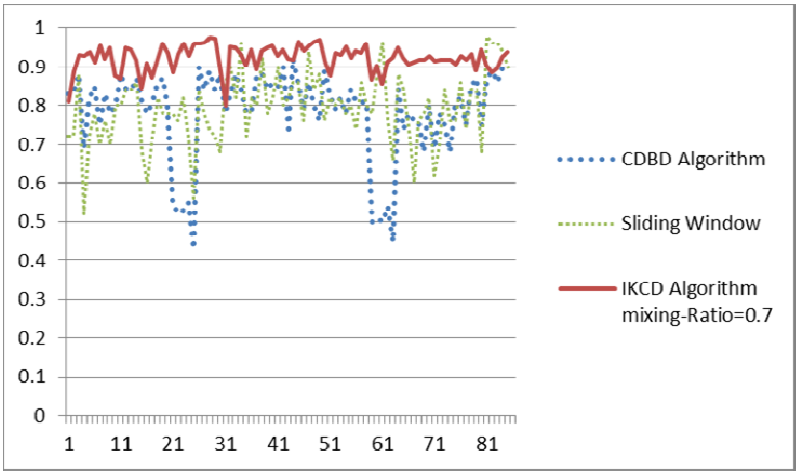

Figure 6: The accuracy of the IKCD, sliding window, and the CDBD algorithms using threshold value (th) = 0.

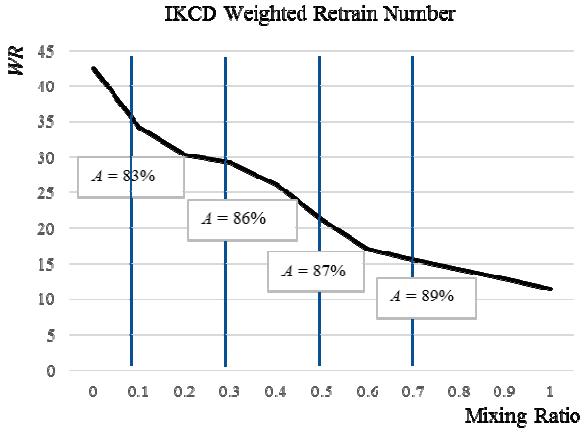

Figure 7: Model weighted retrain of IKCD algorithm using threshold (th) = 0.3

4.1.4 EXPERIMENT III

If the threshold increases to moderate value (th=0.3and th=0.5), the weighted retrain decreases rapidly, as shown in Figure 7and Figure 8. However, the IKCD performance is still better than or equal to the sliding window approach and the CDBD algorithm, as illustrated in Figure 9 (IKCD=89%, sliding window=79%, CDBD=78%) and Figure 10 (IKCD=79%, sliding window=79%,CDBD=78%), respectively.

4.1.5 EXPERIMENT IV

When the threshold increases, the experiments show that when using a high threshold (th=0.6), the retrain number decreases to its minimum value (Figure 11). This is a major target of the IKCD algorithm. Also, IKCD offers a comparative performance with respect to CDBD and sliding window algorithms as Figure 12 shows. In this experiment, we increase the threshold to a high value equal to 0.6. Figure 10shows the effect of increasing the threshold value, which has led to a decrease in the weighted retrain number to the lowest level. This is what the proposed model

targets to achieve, since this means reducing the number of retraining the learning model and thus reducing the cost. In addition, IKCD offers high performance compared to CDBD and sliding window algorithms as shown in Figure 11.

In this experiment, we increase the threshold to a high value equal to 0.6. Figure 10shows the effect of increasing the threshold value, which has led to a decrease in the weighted retrain number to the lowest level. This is what the proposed model targets to achieve, since this means reducing the number of retraining the learning model and thus reducing the cost. In addition, IKCD offers high performance compared to CDBD and sliding window algorithms as shown in Figure 12.

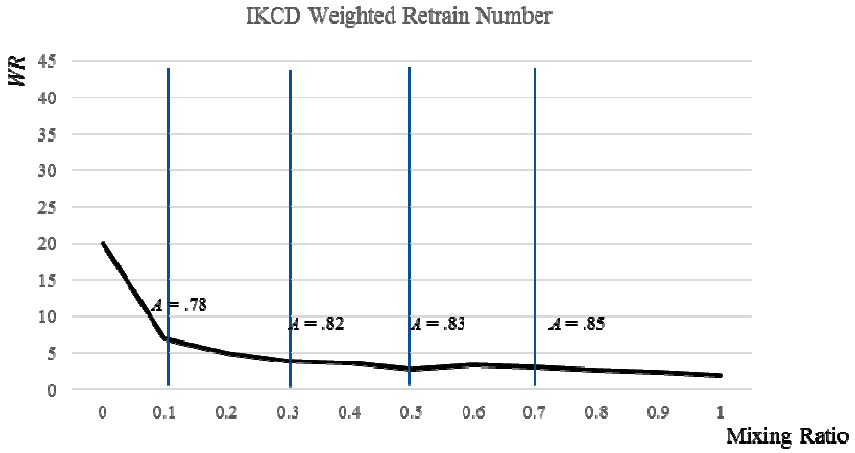

Figure 8: Model weighted retrain of IKCD algorithm using threshold (th)=0.5

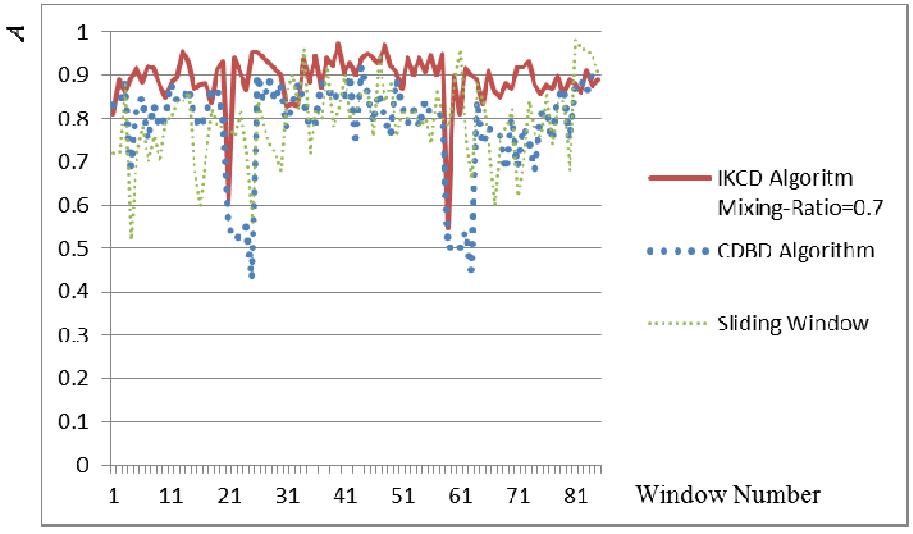

Figure 9: The accuracy of the IKCD, sliding window, and theCDBD algorithms using threshold value (th) = 0.3

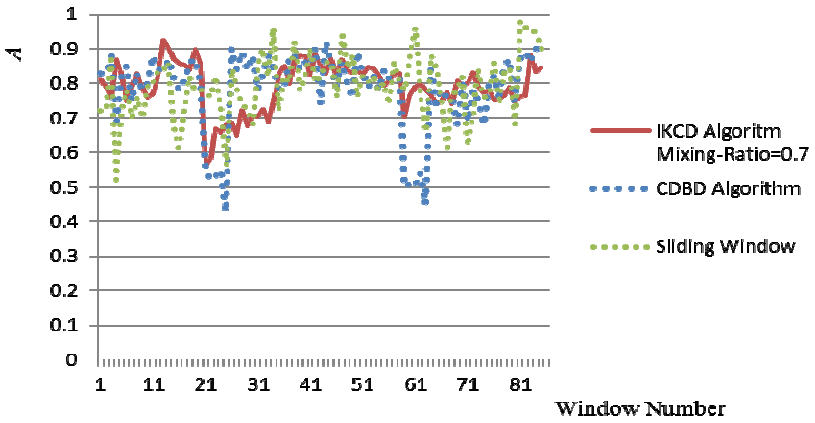

Figure 10: The accuracy of the IKCD, sliding window, andthe CDBD algorithms using threshold value (th) = 0.5

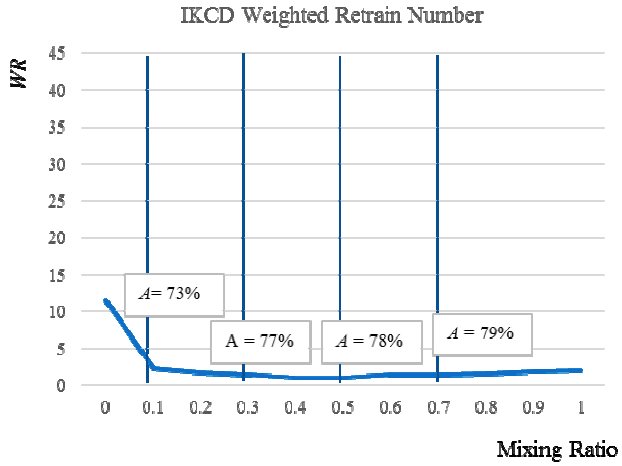

Figure 11: Model weighted retrain of IKCD algorithm using threshold (th) = 0.6

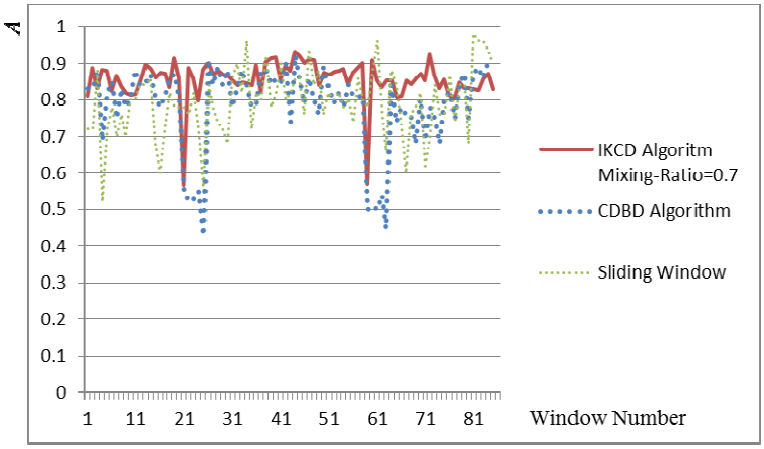

Figure 12: The accuracy of the IKCD, sliding window, and the CDBD algorithms using threshold value (th) = 0.6

4.2 2 0-NGC DATA SETS

CDBD data is only limited to the re-occurring drift, therefore, this research proposed a new dataset composed of 20-Newsgroups called 20-Newsgroups concept drift (20-NGC). It has three versions 20-NGC-S, 20-NGC-R, and 20-NGC-G, which simulates the sudden, re-occurring, and gradual concept drift, respectively.

4.2.1 DATASET

The 20-NGC dataset uses three classes. Two classes of the three are used to form the training data, while the test data is composed from the same two classes in segments that do not contain drift. To introduce the drift, one of the two classes - used in the training - is switched by the third class.

This research simulates three types of the concept drift: re-occurring, sudden, and gradual drifts using a data stream of length 1200 instance. The data classes used for training are "comp.graphics" and "rec.autos", while the class "rec.sport.hockey" is used to introduce the drift [28]

The re-occurring drift takes place three times. The new class completely replaces the drifted class instances. Table 2 shows the structure of the used data. The sudden drift takes place when the new class totally replaces the drifted class. Table 3 shows the structure of the used data. The gradual drift takes place over five steps. In each step, the drifted class instances are decreased by about 20%, while the new class instances are increased by the same ratio. Table 4 shows the structure of the used data.

4.2.2 RESULT

The results are evaluated using: (i) F-measure, (ii) the total accuracy of the system, and (iii) the model retrain count. For each drift type, a parametric study on different values of thresholds and mixing ratio are used to find the best combination of both parameters. The model accuracy is evaluated at different sizes as illustrated in Table 5. The results show that using window size = 200 produces the best performance compared to data with larger window sizes. Moreover, the larger window requires a larger memory. In turn, we use the window_size=200 for the results illustrated thereon.

Table 2: The structure of 20-NGC-R for re-occurring concept drift

		comp.graphics	rec.autos	rec-sport-hockey
Training data	200	100	100	---
Window 1	250	125	125	---
Window 2	250	125	---	125
Window 3	250	125	125	---
Window 4	250	125	---	125
Total data	1200	600	350	250

Table 3: The structure of 20-NGC-S for sudden concept drift

		comp-graphics	rec.autos	rec-sport-hockey
Training data	200	100	100	---
Window1	500	250	250	---
Window 2	500	250	---	250
Total data	1200	600	350	250

Table 4: The structure of 20-NGC-G for gradual concept drift

		Comp.graphics	rec.autos	rec-sport-hockey
Training data	200	100	100	---
Window 1	200	100	80	20
Window 2	200	100	60	40
Window 3	200	100	40	60
Window 4	200	100	20	80
Window 5	200	100	---	100
Total data	1200	600	300	300

Table 5: Illustrates the effect of changing the window size on accuracy of the model

Window size	Accuracy
150	0.951
200	0.967
250	0.963
300	0.962

The results show that using a setup, which consists of about 30% - 40% of priorconcepts with 60% -70% of the new concept introduces the best results. This is due to the nature of news stream. For each class of news, this research classified the features into two classes: (i) fixed features, and (ii) variable features. Fixed features represent the set of features which are constant with the class regardless of the occurrence of a concept drift or not. Variable features represent the set of features that are affected with the drift. The combination strategy allows the model to enforce the weight of the fixed features and hence the model becomes able of efficiently handling the concept drifts in news.

Re-occurring Drift: The combination between threshold and mixing ratio reduces the retrain number and maximizes the average total accuracy of the system. At a threshold th=0.6, the minimum model rebuilding number is achieved, as Figure 13(b) illustrates, and relatively produces a good model accuracy (see Figure 13(a)). However, the F-measure demonstrates a poor performance, as shown in Figure 13(c). In fact, the IKCD algorithm enables the model to deal with drifts without rebuilding the model. Also, the mixing ratio affects the value of both F-measure and model accuracy. Using a mixing ratio of 60% from new data is used in model retrain, the number of retraining is kept to a minimum and the accuracy is maximized. However, the F-measure at this mixing ratio is worse than its value for higher mixing ratio. The mixing ratio selected allows the model to gain the ability to adjust to the new concept, and in the same time retain the knowledge of the old concept. In turn, the model does not need to rebuild itself when the drift reappeared. This reflects on the F-measure

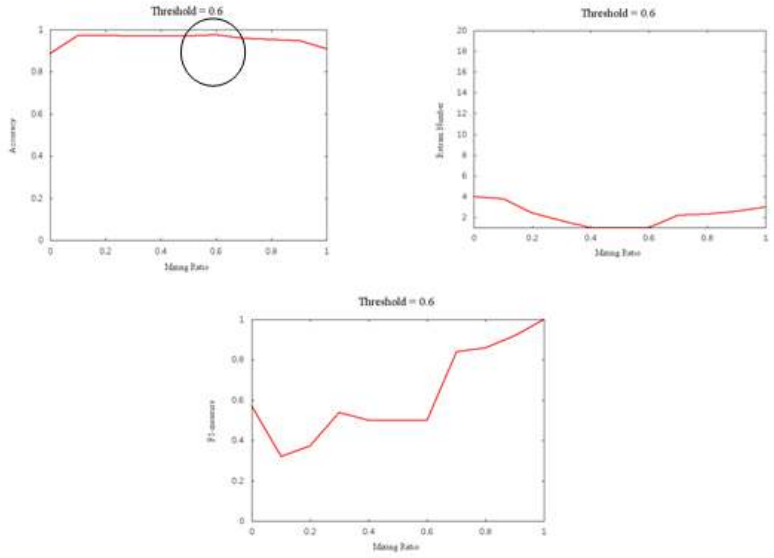

Figure 13: The performance of the re-occurring drift of the 20-NGC at th=0.6 regarding (a) model accuracy, (b) retrain number, and (c) F1-measure.

Gradual concept drift: The IKCD maintains a considerable performance with the gradual concept drift. The model accuracy increases with the increase of mixing ratio, as Figure 14(a) illustrates, and the system reduces the number of retrains and the F-measure as shown in Figure 14(b). However, if the system uses the same mixing-ratio of 60%, similar to the best performing in re-occurring drift, the system maintains an acceptable accuracy.

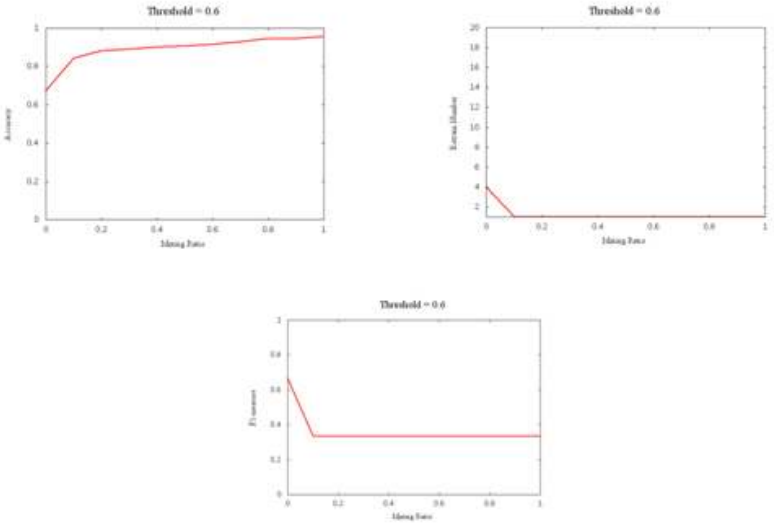

Figure 14: The performance of the gradual drift of 20-NGC at th=0.6 regarding (a) model accuracy, (b) retrain number, and (c) F1-measure

Sudden drift: Sudden drift demonstrates a similar behavior to gradual drift; however, the *F*-measure saturates to its maximum value beyond the mixing ratio of 0.3. Generally, the

IKCD algorithm can maintain a good performance with sudden drift. Figure shows the results of *F*-measure, accuracy and retrain number when the threshold *th=0.6*, respectively.

5 CONCLUSIONS

This paper proposes Incremental Knowledge Concept Drift (IKCD) algorithm which handles the concept drift problem in news recommendation system. The main contributions of the proposed IKCD algorithm are (i) using unlabelled data to build the learning model in addition to (ii) utilizing preserved historical data in adapting that model. IKCD basically includes two phases, the training phase and the adaptive learning phase. The training phase is activated when the drift indicator triggers the existence of a drift. The adaptive learning phase is activated after the model rebuilding is complete and lasts until the arrival of a subsequent trigger. The training phase starts with extracting important features from text instances that are implied in the retraining window. Then clustering technique is applied and the resulting clusters are used in the training process of the learner. Adaptive learning phase added the incremental learning ability to IKCD technique. It detects the drift and formulates retraining window to be used in the training phase.

We conduct two main experiments; first one uses the CDBD dataset which is limited to the re-occurring drift. The target of the first experiment is to compare the proposed concept drift detection and handling technique to CDBD and sliding window algorithms. The second experiment considers the different concept drift types using proposed new dataset. The results show that IKCD enhances the ability of the model to deal with concept drift, especially with re-occurring drift, and that the number of model retraining has decreased. A parametric study was performed using different values of thresholds and different values of mixing ratio. News data streams are the main focus of this research, and 20 news-groups dataset has been used to simulate different types of concept drift. The results show that using a mixing ratio 30% to 40% from old concept with the new concept enables the model to introduce better performance with news concept drift. The results are compared with respect to the results of CDBD algorithm and sliding window algorithm. IKCD outperforms CDBD and Sliding window algorithm. Relative to the CDBD algorithm, IKCD reduces the number of retrain required by the model. With respect to sliding window, IKCD enhances the accuracy of the model.

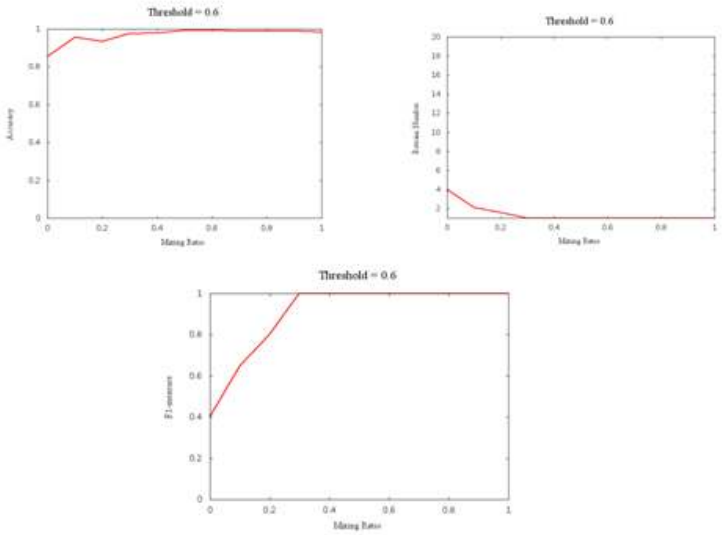

Figure 15: The performance of the sudden drift of 20-NGC, the effect at th=0.6 regarding (a) model accuracy, (b) retrain number, and (c) F1-measure

REFERENCES

[1] F. Ricci, B. Shapira, and L. Rokach, Recommender systems handbook, Second edition. 2015.

[2] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," Knowledge-Based Syst., vol. 46, pp. 109–132, 2013.

[3] A. Adomavicius, Gediminas and Tuzhilin, "Towards the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," IEEE Trans. Knowl. \&Data Eng., no. 6, pp. 734--749, 2005.

[4] I. Žliobaitė, M. Pechenizkiy, and J. Gama, "An Overview of Concept Drift Applications," pp. 91–114, 2016.

[5] J. D. Leskovec, Jure and Rajaraman, Anand and Ullman, Mining of massive datasets. 2014.

[6] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," ACM Comput. Surv., vol. 46, no. 4, pp. 1–37, 2014.

[7] A. Karpatne, "Predictive Learning with Heterogeneity in Populations," 2017.

[8] S. Wang, B. Zou, C. Li, K. Zhao, Q. Liu, and H. Chen, "CROWN: A Context-aware RecOmmender for Web News," Proc. - Int. Conf. Data Eng., vol. 2015–May, pp. 1420–1423, 2015.

[9] Y. Kadwe and V. Suryawanshi, "A Review on Concept Drift," IOSR J. Comput. Eng., vol. 17, no. 1, pp. 20–26, 2015.

[10] A. and B. Šili´c, "Exploring classification concept drift on a large news text corpus," in International Conference on Intelligent Text Processing and Computational Linguistics, 2012, pp. 428--437.

[11] M. M. {Gaber, "Advances in data stream mining," Wiley Interdiscip. Rev. Data Min. Knowl. Discov., vol. 2, no. 1, pp. 79--85, 2012.

[12] E. Lughofer, "On-line active learning: A new paradigm to improve practical useability of data stream modeling methods," Inf. Sci. (Ny)., vol. 415, pp. 356--376, 2017.

[13] B. J. Hammer, Hugo Lewi and Yazidi, Anis and Oommen, "On the classification of dynamical data streams using novel 'Anti-Bayesian' technique," Pattern Recognit., vol. 76, pp. 108--124, 2018.

[14] S.-L. Nguyen, Thi Thu Thuy and Nguyen, Tien Thanh and Liew, Alan Wee-Chung and Wang, "Variational inference based bayes online classifiers with concept drift adaptation," Pattern Recognit., vol. 81, pp. 280--293, 2018.

[15] G. Desrosiers, Christian and Karypis, A comprehensive survey of neighborhood-based recommendation methods. 2011.

[16] I. Žliobaitė, "Learning under Concept Drift: an Overview," pp. 1–36, 2010.

[17] D. Brzezinski and J. Stefanowski, "Reacting to Different Types of Concept Drift :," vol. 25, no. 1, pp. 81–94, 2014.

[18] A. Tsymbal, "The problem of concept drift: definitions and related work," Comput. Sci. Dep. Trinity Coll. Dublin, vol. 106, no. 2, 2004.

[19] G. Widmer and M. Kubat, "Effective learning in dynamic environments by explicit context tracking," Eur. Conf. Mach. Learn. (ECML 1993), vol. 667, pp. 227–243, 1993.

[20] D. Klinkenberg, Ralf & Renz, Ingrid & Ag, "Adaptive Information Filtering: Learning in the

Presence of Concept Drifts," 1999.

[21] P. R. L. Almeida, L. S. Oliveira, A. S. Britto, and R. Sabourin, "Adapting dynamic classifier selection for concept drift," Expert Syst. Appl., vol. 104, pp. 67–85, 2018.

[22] Y. Sun, K. Tang, Z. Zhu, and X. Yao, "Concept Drift Adaptation by Exploiting Historical Knowledge," IEEE Trans. Neural Networks Learn. Syst., vol. 29, no. 10, pp. 4822–4832, 2018.

[23] I. Khamassi, M. Sayed-Mouchaweh, M. Hammami, and K. Ghédira, "Self-Adaptive Windowing Approach for Handling Complex Concept Drift," Cognit. Comput., vol. 7, no. 6, pp. 772–790, 2015.

[24] L. I. Kuncheva, "Classifier ensembles for changing environments," in International Workshop on Multiple Classifier Systems, 2004, pp. 1--15.

[25] C. J. Tsai, C. I. Lee, and W. P. Yang, "Mining decision rules on data streams in the presence of concept drifts," Expert Syst. Appl., vol. 36, no. 2 PART 1, pp. 1164–1178, 2009.

[26] W. F. Hsiao and T. M. Chang, "An incremental cluster-based approach to spam filtering," Expert Syst. Appl., vol. 34, no. 3, pp. 1599–1608, 2008.

[27] R. Bifet, Albert and Gavalda, "Learning from time-changing data with adaptive windowing," in Proceedings of the 2007 SIAM international conference on data mining, 2007, pp. 443--448.

[28] P. Lindstrom, B. Mac Namee, and S. J. Delany, "Drift detection using uncertainty distribution divergence," Evol. Syst., vol. 4, no. 1, pp. 13–25, 2013.

[29] Y. Kim and C. H. Park, "An efficient concept drift detection method for streaming data under limited labeling," IEICE Trans. Inf. Syst., vol. E100D, no. 10, pp. 2537–2546, 2017.

[30] A. Liu, J. Lu, F. Liu, and G. Zhang, "Accumulating regional density dissimilarity for concept drift detection in data streams," Pattern Recognit., vol. 76, pp. 256–272, 2018.

[31] F. Ricci, L. Rokach, and B. Shapira, Recommender Systems Handbook. 2015.

[32] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," Proc. 2003 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - NAACL '03, vol. 1, no. June, pp. 173–180, 2003.

[33] J. O. JOSEPHSEN, "Hypertensjon og hjertets st??rrelse.," Nord. Med., vol. 56, no. 37, pp. 1335–1339, 1956.

[34] Wael H. Gomaa and Aly A. Fahmy, "A Survey of Text Similarity Approaches," Int. J. Comput. Appl., vol. 68, no. 13, pp. 13–18, 2013.

[35] A. Bifet et al., "Early Drift Detection Method," 4th ECML PKDD Int. Work. Knowl. Discov. from Data Streams, vol. 6, pp. 77–86, 2006.

[36] P. Gama, Joao and Medas, Pedro and Castillo, Gladys and Rodrigues, "Learning with drift detection," in Brazilian symposium on artificial intelligence, 2004, pp. 286--295.

[37] B. M. Sundheim, "Tipster/MUC-5 information extraction system evaluation," Proc. a Work. held Fredericksburg, Virginia Sept. 19-23, 1993 -, p. 147, 1993.