Categorizing Concepts for Detecting Drifts in Stream

Sharanjit Kaur *
Deptt. of Computer Science
University of Delhi
Delhi, India

Vasudha Bhatnagar
Deptt. of Computer Science
University of Delhi
Delhi, India

Sameep Mehta
IBM India Research Lab
New Delhi
India

Sudhir Kapoor
Hindu College
University of Delhi
Delhi, India

Abstract

Mining evolving data streams for concept drifts has gained importance in applications like customer behavior analysis, network intrusion detection, credit card fraud detection. Several approaches have been proposed for detection of concept drifts in the context of supervised learning in data streams. Recently, researchers have been looking into the problem of identifying concept drifts in unlabeled data streams. Prevalent approaches study the evolution of streaming clusters using intrinsic and extrinsic characteristics of the discovered clusters, where each cluster is considered a concept.

In this paper we model an unlabeled, uniform data stream as a stochastic poisson process and study the arrival pattern of data points to analyse the nature of an evolving concept (cluster). Each concept is modeled as stochastic poisson process and is individually observed for arrival rates of the incoming data points. A random sample of arrival rates is collected for each concept and appropriate non-parametric tests are applied to infer the nature of evolution for the concept. Concept drift in the stream can be inferred by the overall behavior of the concepts. We also propose a taxonomy of various types of concept behaviors and inter-relation among them. Experiments have been performed to demonstrate feasibility, validity and scalability of the proposed method.

Keywords: Unlabeled data streams, concept drift, clustering, stochastic poisson process

1 Introduction

It is a well known and accepted fact that the underlying data generation mechanism for streaming datasets changes (may be slowly) over time. For example, change in customer buying preferences, change in students interest in choosing subjects depending on current market demand and change in number of network packets received at the server. These changes are typically characterized by outliers, change points or concept drifts. Outliers and change points indicate abrupt change in data whereas concept drift is a relatively slower process.

Researchers are interested in *concept drift* detection because, often, the cause of the change is not known a-priori. Consequently, *concept drift* has a degrading effect on the learned model, and eventually on the target objective too [22]. Presence of concept drift therefore, requires revision of the learned model to obtain accurate results.

Concept drifts have been extensively studied in the context of supervised learning. The objective is to maintain the desired accuracy level of the predictor, by adapting the learner to the changing concepts [2, 11, 12, 23]. Each class is treated as a concept and the change in data distribution of the class is modeled as concept drift [8]. However, in unlabeled data, non-availability of class label accentuates the problem because there is no known distinction between underlying concepts. In such settings, clusters have been used to model concepts and study of their evolution has been used to detect concept drift [5, 13, 19, 20].

We propose a taxonomy of concepts based on the changes (drifts) they have undergone and present a state space diagram for transitions. We also provide physical significance of such transitions. For example, about a decade ago science subjects were highly popular among undergraduates university students, while finance courses enjoy similar popularity currently. Thus analyzing student data for last ten years would reveal *Science* as a diminishing concept whereas *Economics* and *Commerce* as emerging concepts. However, *Literature* is a consistent concept as it still maintains the popularity level as it had earlier. The taxonomy can be extremely useful for obtaining actionable insights from the data. For example, educationists can take proactive measures to popularize science courses by offering placement in reputed national science laboratories.

In this work incoming stream is modeled as a stochastic *poisson* process characterized by its rate and can be split into multiple child processes. Each

*corresponding author, email id: skaur@cs.du.ac.in

15^{th} *International Conference on Management of Data*
COMAD 2009, Mysore, India, December 9–12, 2009

child stochastic poisson process is mapped to a concept. Each process generates data with a specific distribution and is characterized by the rate of data generation.

The proposed algorithm processes the incoming data and captures detailed data distribution in grid based synopsis. Clustering is performed to discover concepts existing in the stream. The arrival rate for each concept is sampled at random interval, thereby generating a sample of iid observations on arrival rates with unknown distribution. Non-parametric statistical tests are applied on this sample to categorize the concepts as *emerging, diminishing, transitional, random or consistent.*

To summarize, the key contributions of the article are:

- An algorithm to facilitate detection of concept drifts in an unlabeled and smooth data stream modeled as a collection of *stochastic poisson processes*. The concepts in the stream are modeled as *child stochastic poisson processes* and are obtained by clustering the stream.
- A taxonomy of concepts based on the changes they undergo and a transition model, which concepts follow during their life times.
- Use of a novel approach by sampling arrival rate for each concept for studying its evolution. Subsequently, non-parametric tests are used on the samples to infer about the nature of evolution of each concept, and categorize it accordingly.
- Experimental analysis to demonstrate the feasibility, validity and scalability of the proposed method.

Section 2 describes modeling of stream as stochastic poisson process. Sections 3 and 4 delineate the main contributions of the paper. The detailed algorithm is given in Section 5 and related work is discussed in Section 6. Experiments are reported in Section 7, and Section 8 concludes the paper.

2 Stream as a Stochastic Poisson Process

A data stream $\mathcal{S}$ is a real-time, continuous, ordered sequence of data instances [9]. Consider a data stream with a uniform arrival rate λ and let $N(t)$ denote the total number of points that have arrived up to time t. Note that stream $\mathcal{S}$ satisfies:

1. $N(t) \geq 0$.
2. $N(t)$ is integer valued.
3. if $s < t$, then $N(s) \leq N(t)$.
4. For $s < t$, $N(t) - N(s)$ equals the number of points that have arrived in the interval (s,t].

Satisfaction of the above conditions makes stream a counting process $\mathcal{P}(t)$ [17]. Further, $\mathcal{P}(t)$ also satisfies:

1. $\mathcal{P}(0) = 0$.
2. The process $\mathcal{P}(t)$ has independent increments i.e. the number of data instances that arrived in disjoint time intervals are independent.
3. $P(\mathcal{P}(t+h) - \mathcal{P}(t) = 1) = \lambda * h + o(h)$.
4. $P(\mathcal{P}(t+h) - \mathcal{P}(t) \geq 2) = o(h)$ where $o(h)$ defines a function f s.t. $\lim_{h \to 0} \frac{f(h)}{h} = 0$.

Accordingly, $\mathcal{P}(t)$, which characterizes the smooth data stream $\mathcal{S}$, can be considered as a stochastic poisson process (SPP) with parameter λ [17].

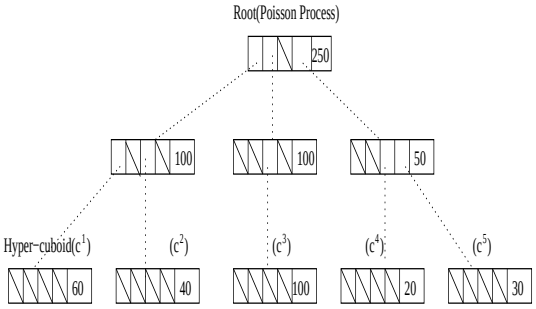

Figure 1: Grid representation of the splitting of stochastic poisson process in a two dimensional data stream with $g = 4$

2.1 Grid Structure for Analyzing Stream

The incoming data in the stream $\mathcal{S}$ is processed into a grid like trie structure G, which is a collection of hyper-cuboids (cells) in bounded data space. Each hyper-cuboid represents a data region of pre-specified granularity[1] in the data space. The structure maintains a detailed data distribution of all points received in stream. Such a structure is capable of supporting both the categorical and numeric data.

Given the dimension set $\Delta = \{a_1, \ldots, a_d\}$ for d-dimensional numeric dataset. Let l_i and h_i respectively be the lowest and the highest data values along dimension a_i, as known to the domain expert. The range $r_i = [l_i, h_i]$ of a_i is divided into g equi-width intervals $[l_i^1, h_i^1], (l_i^2, h_i^2], \ldots, (l_i^g, h_i^g]$, such that $l_i^1 = l_i, h_i^g = h_i$, where g is a user defined parameter. We keep g same for all dimensions to simplify notation, even though there is no practical or implementational limitation in this respect. In case of categorical data, g is the size of the domain and will vary for different attributes.

Each hyper-cuboid (cell c^m) represents a data region $(I_{1,q^1}^m \wedge I_{2,q^2}^m \wedge \ldots \wedge I_{d,q^d}^m)$ where I_{i,q^j}^m refers to

[1] Attempts have been made to get rid of the need to predetermine the granularity of the data space in [15, 16]. However, the computational cost of these approaches is prohibitive.

interval $q^j (1 \leq q^j \leq g)$ of i^{th} dimension of the c^m. At any instance, the grid maintains only those hyper-cuboids which have at least one point in the corresponding data region. A hyper-cuboid, referred as a cell in the rest of the paper, maintains statistical information required for subsequent computations.

2.2 Non-homogeneous SPP for Cells

As mentioned earlier, stream $\mathcal{S}$ is modeled as a SPP $\mathcal{P}(t)$ with parameter λ (arrival rate). The points arriving in $\mathcal{S}$ are assigned to appropriate cell as per their data values. Conceptually, each cell c^m in the grid represents a data region that receives points and can be considered as child SPP of the root process ($\mathcal{P}(t)$). However, the arrival rates for the cells are not constant because the assumption of uniform rate may not hold for individual cells, even though it is valid for $\mathcal{P}(t)$. In fact depending on the change in data characteristics i.e. concept drift, the arrival rate for each cell may vary in an unpredictable manner and gives clues about the changes in data distribution. Because of this reason, the SPP's at cell level are non-homogeneous in nature with parameter $\lambda^m(t)$, which denotes the rate of arrival of points in the c^m at time t [17].

At time t, the process $\mathcal{P}(t)$ at the root of the grid gets splitted into m processes where m is the number of cells in the grid. In case the stream is stationary without concept drift, then $\sum_m \lambda^m(t) = \lambda$. Since the assumption of stationarity is not true, in the case of continuous non-ending data stream $\sum_m \lambda^m(t) \rightarrow \lambda$ as $m \rightarrow \infty$ [17]. Figure 1 shows grid synopsis for SPP with five subprocesses (cells) for 2-dimensional data stream. Here $g = 4$ and the numbers in the cells denote the count of the data points arrived.

2.3 Modeling Concepts in Stream

At time t, a cell c^m in the grid represents a *concept* at the lowest level of abstraction and stores arrival time of the first point (f^m), number of points contained (n^m) and arrival rate $\lambda^m(t) (= \frac{n^m}{t - f^m + 1})$. As time progresses, the number of cells in the grid increases denoting increase in the number of concepts in $\mathcal{S}$. Monitoring all these concepts for detecting concept drift is computationally expensive.

In order to reduce the number of units under observation and to make computations tractable, the cells in grid are clustered to yield concepts at higher level of abstraction. These concepts are expected to be semantically more meaningful and hence representative. Connected component analysis is used to generate clusters. The additive property of SPP is exploited here to associate a SPP with each cluster [17]. Therefore, SPP $P^k(t)$ for the cluster C^k is the additive outcome of the SPPs corresponding to the constituent cells.

For cluster C^k with N^k cells, i) arrival time of first point (ϕ^k), ii) number of points (η^k) and iii) arrival rate (Λ^k) at time t are computed as follows:

$$\phi^k = min(f^1, f^2, \cdots, f^{N^k}) \ldots\ldots$$
$$\eta^k(t) = \sum_{m=1}^{N^k} n^m \quad (1)$$
$$\Lambda^k(t) = \frac{\eta^k}{t - \phi^k + 1}$$

The arrival rate of each cluster is sampled periodically for its categorization.

3 Identifying Concept Drift

Concept drift in stream indicates data evolution; essentially a change in the underlying data distribution. It is an aggregated effect of the changes in the concepts existing in the stream. Hence, the task of ascertaining the concept drift in the stream is accomplished by inferring the changing trends in each concept individually and combining the overall effect of change.

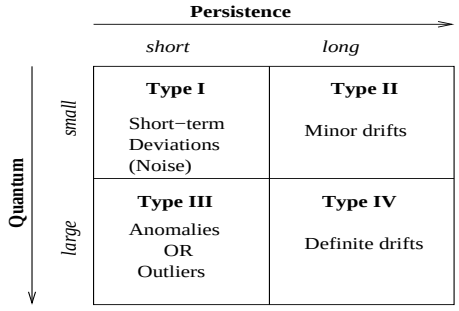

Figure 2: Nature of changes in concepts

The nature of concept drift in stream can be expressed as combination of quantum and duration or the persistence of change. Thus identification of concept drift rests on definition of 'how much' change and for 'how long' duration. Figure 2 shows a matrix of nature of changes in concepts with respect to quantum and duration.

Type I consists of small quantum of changes (*statistically non-significant*) observed for short duration that are natural to expect in any data generation process. Such non-persistent changes do not require attention and must be ignored by a concept drift detection algorithm (CDDA). Type III changes indicate aberration or an unexpected event in data generation process. An example of type III change is a sudden and quantum jump in the patient admission during the spread of an epidemic. Type III changes that characterize swift, large and short drifts are strong indications of anomalies or outliers, and may require immediate action. A good CDDA must be able to identify type III concept drifts.

Type II and IV are persistent changes in a sense that they are observable over a relatively longer period

of time. They differ primarily in the time they take to build a gradient which is observable. Consequently, their detection is influenced by the periodicity of observation. Developing an algorithm which is hedged from the periodicity of observation is the ultimate goal of the researchers in this area.

3.1 Methodology in Detail

As the stream begins, the incoming data points are processed and concepts in the stream are discovered. As mentioned earlier, each concept is generated by a stochastic poisson process with a corresponding sample of arrival rates. Recall that the arrival rate of a process (concept) is computed using statistics stored in its constituents processes (cells) (Eq. 1).

An incoming data point in the stream is added to the appropriate concept by the online component of the algorithm. Each existing concept is sampled for its arrival rate. If a new concept gets created during the processing, it is noted and a corresponding new sample set is created. When the sample of desired size s has been obtained, non-parametric tests are applied to categorize the corresponding concept as described later. The categorization may also be done when demanded by the user.

Depending on the user specified time period for detection of concept drift, an average sampling periodicity Φ is determined such that a sample of size $s \geq 30$ (statistically large [10]) can be obtained. Actual sampling is performed at random intervals with average sampling periodicity Φ. Randomness ensures that the arrival rates are independently and identically distributed (iid) over time, which is a necessary requirement for the statistical tests used for categorization.

Occurrence of Type I drift may introduce noise in the collected sample. Smoothing, a noise reduction technique commonly used in signal processing is employed to mitigate this effect. We use 3-points sliding-average smoothing function which is defined as follows:

$$\hat{p}_j = \frac{(p_{j-1} + p_j + p_{j+1})}{3},\ 2 \leq j \leq n-1 \qquad (2)$$

where $\hat{p}_j$ and p_j are j^{th} points in the smoothed signal and original signal respectively, and n is the total number of points in the sample.

Type III drifts, characterized by outliers and anomalies (noise) are inherently identified and possibly removed during CCA based approach for clustering [3]. However if it is prominent, it is captured by the algorithm as evident in experiment described in Section 7.3.2.

3.2 Categorization of Concepts

For each concept, its iid sample of arrival rates, whose distribution is *unknown* is examined for a trend using non-parametric statistical tests. A newly discovered concept may have sample of size $< s$. Hence it is not prudent to comment upon its nature, till the sample is complete. During this period, the concept is considered 'Novel' (N).

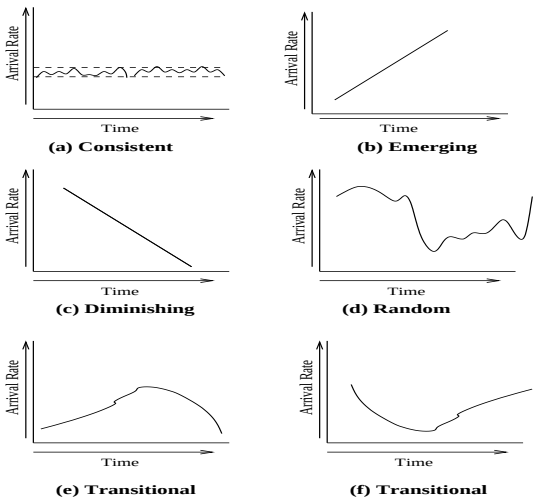

Figure 3: Categorization of concepts on the basis of their arrival rates

At a later time, when the sample is complete, the concept is categorized as one of the following.

1. Consistent Concept: A concept is said to be consistent (C) if arrival rate in the corresponding process does not vary significantly (Figure 3a). This indicates that concept is receiving points at nearly uniform rate and as per the expectation.

2. Emerging Concept: A concept with increasing arrival rate is said to be an emerging (E) concept (Figure 3b). The increasing arrival rate indicates that concept is firming up as number of points supporting the concept are increasing with time. Such concepts contribute to concept drift.

3. Diminishing Concept: A concept with decreasing arrival rate is said to be a diminishing (D) concept (Figure 3c). Such concept fades away with time and contributes to the drift. A diminishing concept indicates a clear loss of support in terms of the data points which belong to it.

4. Transitional Concept: Transitional concept reflects instability of emerging or diminishing trend during observed duration (Figure 3e, 3f). Usually, a concept which has increasing arrival rates followed by decreasing arrival rates or vice-versa is categorized as a transitional (T) concept.

5. Random Concept: A concept which cannot be categorized as any of the above mentioned concepts is called random (R) concept. The sample

of its arrival rates does not exhibit any statistically significant pattern(Figure 3d).

Though arrival rate is the primary characteristic of the behavior of the concept over time, some other attributes are also important. For instance, η - the number of points and N - the number of cells in a concept also characterize it and influence categorization. η is an indicator of the membership of cluster, while N is the indicator of the volume of the cluster. Each of the three features Λ, η and N can either vary i.e. increase ($\uparrow$) or decrease ($\downarrow$) or remain unchanged ($\doteq$), overtime.

Let $\Lambda(i)$, $\eta(i)$ and $N(i)$ denote the three attributes of a concept at time t_i. Each of them can have three possible transitions at time t_{i+1} i.e. increase ($\uparrow$), decrease ($\downarrow$) or unchanged ($\doteq$). Thus at time t_{i+1}, attributes of a concept can have 27 possible transitions based on which it can be categorized. After enumerating each of these transitions, it can be seen that 20 of these transitions are not feasible. For example, a transition with no change in the number of points in a concept ($\eta(i) = \eta(i+1)$), but with a change in the number of cells is not feasible. A careful analysis reveals that only five attribute transitions are possible, which determine the category (state) of a concept as shown in Table 1.

States	Attributes Λ	η	N
Consistent (C)	$\doteq$	$\uparrow$	X
Emerging (E)	$\uparrow$	$\uparrow$	X
Diminishing (D)	$\downarrow$	$\doteq$	$\doteq$
Diminishing (D)	$\downarrow$	$\uparrow$	$\doteq$
Diminishing (D)	$\downarrow$	$\uparrow$	$\uparrow$

Table 1: Five possible states of a concept based on changes in its three attributes; $\doteq$ denotes unchanged; $\uparrow$ denotes increase; $\downarrow$ denotes decrease; X denotes don't care

Changes in both η and N have been effectively used for cluster evolution studies earlier [19, 20]. We have used only the arrival rate for categorization, as it is a good representative of other two attributes.

3.3 Transition of Concepts States

At any time t, the state of a concept is represented by its category. A concept may change its state during its lifetime as shown in Figure 4. A concept begins life as a novel concept, when it is first discovered but may not have a complete sample for testing the nature of its evolution. Subsequently a novel concept can become either consistent, emerging or diminishing concept depending upon the rate at which the incoming points in the stream join the concept. The emerging concept may either retain its state or change to consistent or diminishing. A consistent concept persists as long as the arrival rates of the points joining the concept remains nearly constant, otherwise the state transition takes place. Transitional and random states can be observed in between transitions to any of the shown stable states.

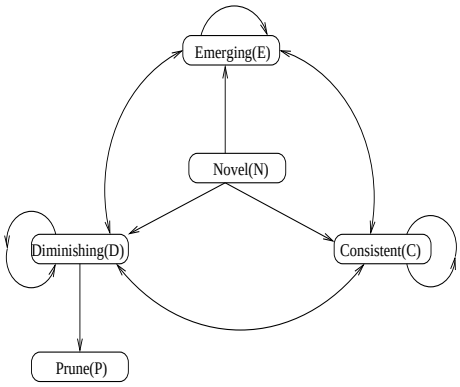

Figure 4: Semantics of state transitions

Concept which is diagnosed (possibly repeatedly) diminishing is retained till such time when no points are received in last sample. Detailed semantics of the states transition are given in Table 2.

Transition	Semantics
N→D	A novel concept appeared and is fading now.
N→E	A novel concept is still emerging and more data in the stream is favouring this concept
N→C	A novel concept is stable now
D→D	A diminishing concept is weakening further
D→E	A diminishing concept is getting revived More data points are exhibiting distribution favourable to this concept
D→C	A diminishing concept is stable now
D→P	A diminishing concept has not recived points in last $s/2$ observations and is pruned
E→E	An emerging concept is strengthening further
E→D	An emerging concept does not receive new points and becomes weak
E→C	An emerging concept is stable now

Table 2: Semantics of state transitions

After categorization of concepts, the knowledge discovered is aggregated to quantify the concept drift in the stream. There can be multiple ways in which aggregation can be carried out, each of which may be application dependent. The methodology proposed by Choudhary et al. [7] can be adopted for quantification of concept drift.

4 Tests for concept categorization

Let $L^k = \{\Lambda^k(1), \Lambda^k(2), \cdots, \Lambda^k(s)\}$ be the sample of smoothed arrival rates for a concept C^k ($s \geq 30$). An overview of the applied statistical tests is given in the algorithm later. The tests for categorization are detailed below.

4.1 Consistent Concept

Intuitively, a consistent concept has nearly constant arrival rates in L^k and variance is a good enough test for capturing variation. However, it is difficult for the

naive user to set the threshold for accepting the consistency of the concept. The Coefficient of Variation (CV) is an alternative measure which requires threshold (θ_v) to be given in percentage, and hence is intuitive.

CV is a statistical measure for computing the relative dispersion of data points in a data sequence around the mean and is computed as $CV = \frac{\sigma}{\mu} * 100$. Lower value of CV indicates higher similarity in data. Computed CV for L^k, if found less than $\theta_v\%$ indicates consistent concepts.

4.2 Random Concepts

One sample run test for ups and downs is a useful check for non-randomness in a sequence of observations and requires minimum of formal assumptions [10, 14]. The test is based on the premise that if an observed value in the sequence is influenced by its position or by its preceding or succeeding observations, the process is not truly random.

The test is derived from the theory of runs, where a *run* is a succession of identical letters/symbols which is preceded and followed by same symbol or no symbol at all. Too few runs imply a definite grouping or a trend, where as too many runs indicate erratic behavior in the sequence [10]. In the test, the null hypothesis of randomness (H_0) is tested against the alternate hypothesis (H_1) of non-randomness. The test is inferred at α level of confidence and if H_0 is accepted then there is an evidence of randomness in the sequence.

Given a sequence L^k, set symbol '1' if $\Lambda^k(i) < \Lambda^k(i+1), (0 \leq i < s)$, else set symbol '0'; ignoring the equal case. This leads to a sequence of 1's and 0's of size $< s$. Defining a run as a sequence of same symbols, let u be the total number of runs, n_1 be the number of 1s and n_2 be the number of 0s. For $s > 25$, it has been shown that u is approximately normally distributed with mean μ' and variance σ' [14], where

$$\mu' = \frac{2 * (n_1 + n_2) - 1}{3}$$
$$\sigma' = \sqrt{\frac{16(n_1 + n_2) - 29}{90}}$$

z-score is computed as $z = \frac{u - \mu'}{\sigma'}$ and H_0 is rejected if $z \leq l_\alpha$ or $z \geq h_\alpha$ where l_α and h_α are low and high values at α level of significance for normal distribution.

4.3 Emerging and Diminishing Concepts

Rejection of H_0 indicates absence of evidence of randomness in the sequence L^k, or alternately presence of a trend. The natural follow-in task in sequence is to determine the type of trend.

Sen's slope test, a nonparametric alternative for estimating a slope for a univariate time series, is a well known test to detect increasing or decreasing trend in the series [18]. It uses the slope as a change in measurement with respect to time. The test is more robust as it is not affected by gross data errors and outliers [18].

Given L^k, slope between two readings $\Lambda^k(i)$ and $\Lambda^k(j)$ is computed as $\frac{\Lambda^k(j) - \Lambda^k(i)}{j - i}$, where $1 \leq i < j \leq s$. This leads to computation of $\frac{s(s-1)}{2}$ slope values, whose median is computed and compared with specified confidence interval. L^k exhibits a trend if median is statistically different from zero and lies within the confidence interval [4]. Further, signs of slopes are used to report the trend as diminishing or emerging. In case there is not sufficient evidence of the trend, the corresponding concept is reported as *transitional*. The rationale for the decision follows from the fact that the previous test has already ruled out randomness in the arrival rate. A transitional concept usually has an emerging and a diminishing trend within same sample.

5 CCDD Algorithm

CCDD (**C**ategorizing **C**oncepts for **D**etecting **D**rifts) begins with an empty grid structure G and as data points stream in, G gets populated. After a prespecified gestation period, concepts are discovered and arrival rate for each concept is computed. This task denotes the start of building of the sample L^k. These concepts are updated at random time intervals (average sampling periodicity - Φ) and the arrival rates are observed for each concept. Once the complete sample for a concept is obtained, it can be categorized.

5.1 Notations

The application of connected component analysis on G at time t_i delivers a set $\mathcal{C}_i = \{C_i^1, C_i^2, \cdots, C_i^{K_i}\}$ of K_i concepts. Each concept C_i^j is a tuple $< id, \eta, N, \Lambda, \phi, type >$, where id is the concept identifier and *type* refers to category of the concept.

5.2 Description

Let $\mathcal{C}_0$ consists of K_0 concepts discovered at time t_0. Subsequently, all incoming points are inserted in G, till it is time to sample the arrival rates. Statistical information maintained in cells are used to update three attributes of each concept in $\mathcal{C}_i$, thereby leading to updated clustering scheme $\mathcal{C}_{i+1}$. This task may result into discovery of new concept, which are marked 'N'. Such concepts arise because some of the recently added cells in G do not belong to any of the existing concepts. Such cells may either collectively give rise to a new concept or may individually be precursor of new concepts. Two or more existing concepts may also get merged due to newly added cells and additive property of poisson process is used to compute a sample of arrival rates for merged concepts. In case, pruning

of stale cells leads to splitting of concepts, the statistics for splitted concepts are computed afresh using the statistics of constituents cells using Eq. 1.

This process of inserting points in the G and collecting observations at random intervals of time is repeated till either a sample of size s is attained or there is demand for categorization.

Algorithm : CCDD: Categorizing Concepts for Detecting Drifts
Input : Points in Stream $\mathcal{S}$, Sample size s.
Output : All concepts in stream alongwith the categories (either on demand or periodically).

begin
$\mathcal{C}_0$=Initialization(G) //initial concepts
$j = 0$
while more points in $\mathcal{S}$ **do**
  $\mathcal{B}$=random() //with Φ as average sampling periodicity
  Process and insert all incoming points received within time-period Φ in G
  Using G, update existing concepts $\mathcal{C}_j$ and insert new concepts, if any.
  j=j+1 //Next observation
  if ($DemandForDrift$) OR (sample complete) **then**
    Categorize_concepts($\mathcal{C}_j$)
    Quantify drift using concepts and their categories in $\mathcal{C}_{j-1}$ and $\mathcal{C}_j$
  end if
end while
end

Function : Categorize_concepts()
Input : Concepts $\mathcal{C}_i$
Output : Concept Type for each concept.

begin
for each concept C_j^i in $\mathcal{C}_i$ **do**
  if sample size of i^{th} concept $< s$ **then**
    Set $type$='N'
  else
    Compute Co-efficient of Variation (CV)
    if ($CV <= \theta_v$) **then**
      Set $type$='C'
    else
      Apply Runs 'ups and downs' test // evidence for randomness
      if (non-random) **then**
        Apply Sen's Slope Method to detect increasing and decreasing slope
        if slope is negative **then**
          Set $type$='D'
        else
          if slope is positive **then**
            Set $type$='E'
          else
            Set $type$='T'
          end if
        end if
      else
        Set $type$='R'
      end if//non-random
    end if//CV
  end if//sample size
end for//each concept
end

6 Related work

Recently, the idea of concept drift has been applied to unsupervised learning to detect evolution of clusters [1, 5, 13, 19, 20]. We describe some of the recent algorithms proposed to capture concept drift and follow it by comparison of the proposed methodology with some of these algorithms.

6.1 Capturing Cluster Transition

Kernel functions are used by Aggrawal [1] to model clusters and variation in kernel density is used for reporting cluster transitions. Changes are reported based on the computation of the change 'velocity' and finding the location with the highest 'velocity' using assumption of fixed trajectory. Temporal and spatial velocity profiles are maintained at periodic intervals and are used to produce three types of cluster changes viz. dissolution, coagulation and shift. However, the method cannot be used in environment like text stream mining, where feature space may change with time [19].

MONIC framework [19] proposes a cluster transition model which tracks the cluster changes, not on the basis of topological properties of clusters, but the contents of the stream obtained by periodic clustering. It tracks two types of cluster transitions viz. external and internal. The transitions are used in making conclusions like mutation, stability and life time etc. of a cluster. External transition imply relationship of a cluster to the rest of clustering, whereas internal transition is detected using form and content of the cluster. An aging function is used to reduce impact of history data on current trends.

An Online Novelty and Drift Detection Algorithm (*OLINDDA*) detects concept as cluster from a one-class perspective. This implies that initial model is learned based only on examples of a single class that represents normal concept [20]. The algorithm detects a novel concept or report concept drift if a normal concept undergoes a change and a new class emerges. A validated cluster which appears very far from the global boundary of all clusters is reported as a novel concept where as small transition within normal clusters is detected as concept drift.

HE-Tree, an entropy-based clustering algorithm reports changes in underlying clustering structure for categorical stream as cluster transitions [5]. Three types of cluster transitions viz. emerging clusters, disappearing clusters and expanding clusters are detected. As number of clusters K may vary with time, *BK Plot* method is used to identify the best K clusters in categorical data and to detect first two types of transitions. It uses incremental entropy to find similarity between two clusters and to report third form of cluster transition. Change in physical characteristics has been used to detect cluster transitions with respect to last clustering.

The method proposed by Lin et al. [13] uses notion of a concept cycle to indicate the concept drift. All clusters formed within a cycle are treated as one concept. A linear regression test is used to predict the next time stamp for occurrence of a new dense cluster which is compared with real next time stamp. The formation of a new dense cluster after the predicted time indicates the beginning of a new concept. But,

Data set	Window1		Window2		Window3		Window4	
	Data (30k)	CT	Data (30k)	CT	Data (30k)	CT	Data (30k)	CT
FD1	C1	C1:C	C2	C1:D,C2:C	C3	C2:D,C3:C	C4	C3:D,C4:C
FD2	C1,C2,C3 (15k,10k,5k)	C1:D C2,C3:N	C1,C3 (25k,5k)	C1:E C2,C3:D	C3,C1 (15k,15k)	C1,C3:T	C1,C3,C1,C3,$\cdots$ ((1k,1k),$\cdots$,15 times)	C1,C3:C

Table 3: Description of Synthetic datasets for validation; CT: Categorization Type; $\Phi = 1000$; $s = 30$

in real life applications like customer buying pattern, there may be multiple concepts within a time cycle which this method cannot capture.

6.2 Comparison

Our work is comparable to the algorithms proposed in [1], *MONIC* [19] and *HE-Tree* [5] as they capture concept drifts in multiple concepts by comparing two consecutive clustering schemes. Topological properties are used in [1, 5] to report concepts transitions whereas *MONIC* [19] distinguishes between internal and external transitions using the contents of underlying data streams.

CCDD algorithm introduces a novel perspective on concept evolution, by considering the arrival rate of the data points in a concept as an indicator of changing data characteristics. This facilitates concise capturing of new and emerging concepts on one hand, disappearing and diminishing on the other. At the same time, consistent and transitional data characteristics are captured effectively. In several applications, including market basket analysis and stock market analysis, arrival rate is more effective indicator compared to change in physical characteristics of concepts like shape, size etc.. This approach thus complements the existing frameworks for concept transitions.

7 Experiment Section

In this section, we describe the experiments performed to evaluate various aspects of proposed algorithm (CCDD) on a synthetic data set and a real data set. All experiments are performed on Intel Centrino processor with 256 MB RAM, running stand-alone Linux (kernel 2.4.22-1). The algorithm is implemented in ANSI C with no optimizations, and compiled using g++ compiler (3.3.2-2). In the experiments, the timing results are averaged over multiple runs.

7.1 Data Set Description

The data sets used for experiments are described below.

1. Synthetic Data in the experiments is generated using *ENCLUS* data generator [6], which generates pre-specified number of clusters with user defined cardinality. The number of dimensions is specified by the user.

2. Intrusion Detection data set (*Kdd cup 99*) consists of a series of TCP connection records, each of which can either correspond to a normal connection or an intrusion. An intrusion is from one of the 22 attack classes. The dataset has 494,020 observations, each consisting of 42 attributes (34 continuous and 8 categorical) [21]. We performed experiment using 34 continuous attributes.

Synthetic data set has been used to validate the framework, because it facilitates simulation of concepts drifts by generating, merging, splitting of concepts as described in Section 7.2. Kdd cup data set consists of attacks of varied sizes. Attack classes like Neptune and Smurf are biggest classes and appear in chunks whereas some of the small attack classes are scattered. Appearance and disappearance of attack classes with normal class affects the evolution of corresponding concepts (attacks) which is captured by the proposed framework as emerging, diminishing, consistent concepts.

7.2 Validation of CCDD algorithm

We validate the proposed framework on both synthetic and real data sets keeping sampling periodicity (Φ) fixed so as to obtain at least one complete sample ($s = 30$) at the end of each window.

7.2.1 Validation using synthetic data set

Clusters were generated using *ENCLUS* and stored in separate files. Initially, a data file (FD1) was created with four concepts in a sequence where each concept was of size 30,000. The data file was created to affirm the expected output of the algorithm in simplest scenario. On execution, one consistent concept was reported in each window with one diminishing concept which was generated in last window (Table 3, Row 1). As expected, all the concepts were identified and categorized correctly.

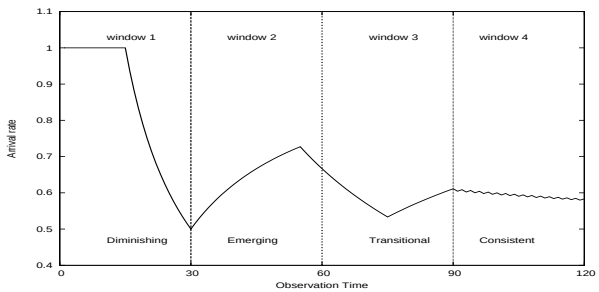

Figure 5: Transition of concept C1 in data set FD2

Later, a second data file (FD2) was created to validate the capability of the algorithm to categorize multiple concepts at same time. Data was partitioned into four windows each of size 30k as before. Three concepts (C1,C2,C3) of respective sizes (15k,10k,5k) were embedded in first window. Records of these concepts were varied in next windows to simulate concepts transitions as shown in the second row of the Table 3.

This data file was streamed in for experimentation and categorization was done after each window for $s = 30$ with sample periodicity $\Phi = 1000$. All the concepts were reported correctly as expected. Figure 5 shows the arrival pattern of concept C1 which transits thru different categories. These transitions match with expected categorization (column CT of Row 2 in Table 3) at the end of each window. This validated the proposal and the programing aspect.

7.2.2 Validation using real data set

Subsequently, we validated algorithm on Kdd cup data set by capturing transition in concepts. The original Kdd cup data set was transmuted such that extracted data had two concepts viz. Smurf attack and Neptune attack. Order of appearance of these attacks was not disturbed, to facilitate repeatability of experiments and to capture natural evolution of clusters in the data. We executed the algorithm with $\Phi = 1000$ and $s = 30$. Table 4 shows the obtained results. The first column shows total records in the stream at the time of categorization. Second and third column show the number of records of respective attack types. CId column denotes the id of the discovered cluster (concept). $x.y.z$ denotes that during the observation period, concepts x, y and z got merged to yield one concept. CT column shows the categorization of the concepts.

Records			Concepts categorization			
			Smurf		Neptune	
TR*	Smurf	Neptune	CId	CT	CId	CT
30	11,258	18,742	1.2.3.4	T	5	N
60	22,753	37,247	1.2.3.4	T	5	T
90	48,878	41,122	1.2.3.4.7	E	5	D
					6	N
120	78,878	41,122	1.2.3.4.7	E	6	D
150	108,878	41,122	1.2.3.4.7	C	-	-
180	138,878	41,122	1.2.3.4.7	C	-	-
210	168,878	41,122	1.2.3.4.7	C	-	-
240	198,878	41,122	1.2.3.4.7	C	-	-
270	223,545	46,455	1.2.3.4.7	C	8	N
300	224,364	75,636	1.2.3.4.7	D	8.9.10	D
					11	N
					12	N
330	243,969	86,031	1.2.3.4.7	C	8.9.10	D
					11	D
					12	D
360	273,969	86,031	1.2.3.4.7	C	-	-

Table 4: Concepts transition; TR*: Total records in thousands, CId: Concept Id, CT: Category

During the first observation period, four concepts were discovered, which ultimately got merged (Concept Id 1.2.3.4). As a new (N) concept of Neptune attack (CId 5) was also forming during this observed period, the concept of Smurf was categorized as transitional (T). Figure 6 shows the arrival rates corresponding to these observed concepts.

During the second observation period, records for Neptune attack were received initially, thereby increasing its arrival rates and decreasing arrival rates for Smurf concept. Later, Smurf concept received more data points which increased its arrival rate whereas arrival rates of Neptune concept decreased (Figure 7). As arrival rates of both concepts were not stabilized, these were reported as transitional concepts which is verified from Figure 7.

Interesting drift can be noted after 120,000 records, when Smurf concept was still emerging but old Neptune concept (CId 6) became diminishing. This small concept eventually vanished and Smurf concept became consistent in the next categorization (Figure 8). This can be vetted from the left hand side column of the table, which shows total absence of Neptune records up to 240,000. The Smurf concept remained consistent even though a small concept of Neptune (CId 8) appeared at 270,000; but became diminishing on appearance of new concepts of Neptune at 300,000. Three Neptune concepts (CIds 8, 9 and 10) got merged, decreasing its arrival rate whereas two more were reported separately as novel concepts. These new concepts became diminishing at 330,000 and vanished subsequently, again making Smurf consistent. Figure 9 shows the appearance of these two novel concepts of Neptune in one window, which became diminishing in adjacent window. *Capturing of such small concepts demonstrates the capability of the algorithm to capture Type III drifts* (Section 3) .

7.3 Sensitivity Analysis

The next set of experiments was run with the same synthetic data file FD2 (Table 3), in a more realistic environment, where Φ was randomized. This was a preliminary test where the objective was to assess the sensitivity of the proposed algorithm with respect to i) sampling periodicity and ii) sample size. This experiment was designed to answer questions like "does the inference change if sampling periodicity changes?" or "does the inference change if the sample size increases?". This investigation is important because determining the sampling periodicity in an application is difficult for the user. Intuitively, the inference is not totally independent of the sampling periodicity. However, we found that varying Φ in wide ranges does not alter the inference regarding the category of a concept in FD2.

7.3.1 Analysis using synthetic data set

Rigorous test was performed on another synthetic data set which simulated rapid variation of concepts within a single observation window. Each chunk corresponds

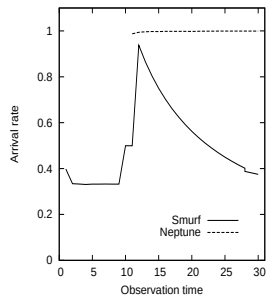

Figure 6: Categorization after 30,000 records

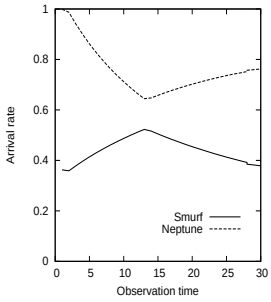

Figure 7: Categorization after 60,000 records

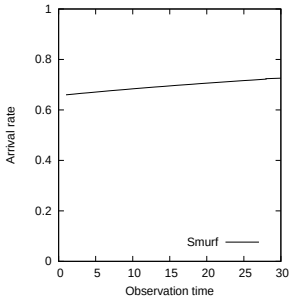

Figure 8: Categorization after 150,000 records

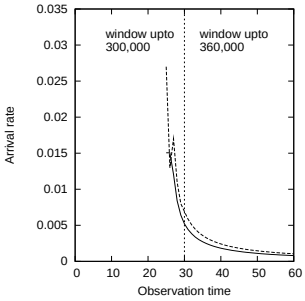

Figure 9: Categorization for 300,000-360,0000 records (Type III drift)

Data (size*)	Chunk1	Chunk2	···	···	Chunk i	···	···	Categorization
DS1 (350k)	$C1$(5k)$C2$(5k) (10k)	$C1$(4.8k)$C2$(5.2k) (10k)	··· ···	··· ···	$C1$(3k)$C2$(7k) (10k)	··· ···	··· ···	$C1$:D, $C2$:E
DS2 (400k)	$C1$(5k)$C2$(5k)$C3$(2k) (12k)	$C1$(5.2k)$C2$(4.8k)$C3$(2k) (12k)	··· ···	··· ···	$C1$(7k)$C2$(3k)$C3$(2k) (12k)	··· ···	··· ···	$C1$:E, $C2$:D $C3$: C
DS3 (500k)	$C1$(4k)$C2$(4k)$C3$(2k) (10k)	$C1$(4.2k)$C2$(3.8k)$C3$(2k) (10k)	··· ···	··· ···	$C1$(6k)$C3$(2k)$C4$(2k) (10k)	··· ···	··· ···	$C1$:E, $C2$:D $C3$:C, $C4$:N

Table 5: Description of Synthetic datasets

to one observation of arrival rate in concepts. We generated five data files each having 50,000 records in one cluster. Three data sets (DS1, DS2, DS3) with (350,000, 400,000 and 500,000) points respectively, were generated using these data files. The doctored datasets are shown in Table 5. Dataset DS1 consisted of data chunks each of size 10,000 points where each chunk had only two types of concepts: C1 (diminishing) and C2 (emerging). DS2 had data chunks each of size 12,000 points and consisted of three type of concepts such that C1 was emerging, C2 was diminishing and C3 was consistent. Dataset DS3 had four types of concepts. Data records of each chunk were shuffled to randomize the appearance of records of each cluster.

The categorization obtained with $\Phi = 10,000$ and $s = 30$ for each data set was used as reference for comparison (Table 5, last column). Each data set was streamed in a separate experiment, and each experiment was repeated with Φ and s varying in wide ranges as shown in Table 6. Further, this range was kept same for DS1 and DS3, in which the chunk size was same.

The concepts were categorized correctly for wide variation of Φ (1000 to 20,000) for DS1 and DS3, and (1000 to 24,000) for DS2. However, categorization for very small periodicities (100 to 2000) was initially incorrect, in which the concepts were reported as 'R' or 'C'. Highly frequent sampling of arrival rates in the concepts initially leads to unpredictable fluctuations in arrival rates making the patterns either random or consistent. However, after the concepts have stabilized (few complete chunks have been processed) the arrival rates accumulate and then small variations do not alter the decision.

Data	Φ	s	Categorization
DS1	1000-20000	30-50	Correct
	100-2000	30-200	R/C initially Correct* afterwards
DS2	100-24000	30-45	Correct
	100-2000	30-200	R/C initially Correct* afterwards
DS3	1000-20000	30-50	Correct
	100-2000	30-200	R/C initially Correct* afterwards

Table 6: Effect of sampling periodicity on concept categorization, *: after first few data chunks have been sampled; R: Random, C: Consistent

	I $s = 30$ $\Phi = 1000$		II $s = 30$ $\Phi = [500, 1000]$		III $s \in [30, 50]$ $\Phi = 1000$	
TR	SF	NP	SF	NP	SF	NP
30	T	N	T	N	-	-
60	T	T	T—D	N	T—D	N
90	E	D,N	E—N	D,N	E	D,N
120	E	D	E—E	D	E	D
150	C	-	C—E	-	E	-
180	C	-	C	-	C	-
210	C	-	C	-	C	-
240	C	-	C	-	C	-
270	C	N	C	N	C	N
300	D	D,N,N	D	D,N,N	D	D,N,N
330	C	D,D	C	D,D	C	D,D
360	C	-	C	-	C	-

Table 7: Sensitivity analysis by varying s and Φ; TR: Total records in thousands, SF: Smurf, NP: Neptune

7.3.2 Analysis using real data set

We performed sensitivity analysis of CCDD algorithm on Kdd cup data by varying sampling periodicity (Φ) and sample size (s). The data file generated for validation was used for comparison purpose. Table 7 shows the categorization of Smurf (SF) and Neptune (NP) with varying Φ and s.

Column I in the table shows actual concept categorization reported with fixed $\Phi = 1000$ and $s = 30$ and has been taken from Table 4 . Column II and III shows the categorization reported by varying Φ and s. The observation is same as that in the previous experiment. The categorization of concepts gets perturbed initially, but as the concepts stabilize, the correct inferences are drawn.

7.4 Scalability Tests

Per-point processing time, sampling time and categorization time are critical for the performance of the algorithm. Per-point processing time in grid is constant and of order ($O(d)$). Time for categorization depends on sample size used in statistical test computation for categorization. Increasing Φ implies reduction of sample size and hence categorization time. However, the sampling time depends on the number of concepts existing in the stream since arrival rates of each concept has to be recorded. The scalability of the algorithm is also tested on both synthetic data as well read data set.

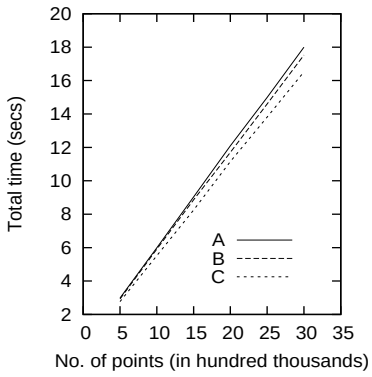

Figure 10: Effect of sampling periodicity on total processing time, A: $200 \leq \Phi \leq 1000$; B: $500 \leq \Phi \leq 1000$; C:$1000 \leq \Phi \leq 3000$

7.4.1 Scalability with synthetic data set

First experiment was conducted on the DS1, which was repeatedly streamed to create a stream of 3,500,000 data points. We varied Φ in small ranges and found linear scalability for total time which includes processing time, sampling time and categorization time. Observations from this experiment shows that total processing time reduces marginally with reduction in Φ (Figure 10).

7.4.2 Scalability with real data set

Next experiment aimed to study scalability with respect to the number of concepts. We used Kdd cup data for this purpose because of the abundance of the number of clusters. Though there are only 23 classes, the number of clusters is very large which keep on appearing and disappearing. This cluster evolution is nicely captured by the algorithm.

First 350,000 records of KDD cup data file were used in experiment. The sampling time and the number of concepts were observed with periodicity varying Φ ($3000 \leq \Phi \leq 6000$). The experiment was repeated multiple times to collect more observations so as to average out the effect of randomness of Φ.

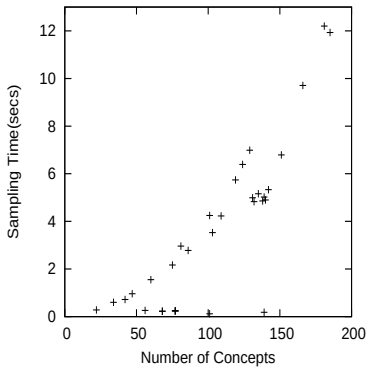

Figure 11: Variation of sampling time with the concepts

Figure 11 shows scatter plot of the sampling times with the changing number of concepts where sampling time includes updation time for concepts and observation time for next reading. A general linear trend is observed which affirms the intuition that if more concepts are present in the stream, then the time required to record the arrival rate will be more because of updation of concepts to incorporate new statistics corresponding to updated cells in grid. However, little noise between 50-150 along x-axis raises some pertinent questions. On investigating, we observed that these were the locations in the data file where all the records were of the same class and hence, were going to the same concept. Consequently, minimal number of concepts were being updated explaining near zero sampling time.

8 Conclusion

In this paper, we presented CCDD (Categorizing Concepts for Detecting Drifts) algorithm, which captures concept drift in an unlabeled data stream by monitoring arrival pattern of data points. A taxonomy of various types of concept drifts has been presented and

transitions that occur in the life-cycle of a concept are modeled.

The basic premise of the algorithm is that the arrival rate of points in a concept is a good indicator of its evolution. Accordingly, the algorithm samples each concept in the stream for respective arrival rates. The iid sample is subjected to non-parametric statistical tests to infer about the nature of evolution. Experimentation had been done to demonstrate feasibility, validity and scalability of the algorithm and the results were found to be encouraging. The algorithm also was found to be nearly non-sensitive to the sampling periodicity and sample size.

Acknowledgment We acknowledge with gratitude Prof. Sharma Chakravarthy (University of Texas, Arlington) for his insightful comments. We also thank anonymous reviewers for valuable suggestions to improve the paper. This work was supported by research grant no. Dean(R)/R&D/2008/185 from University of Delhi, Delhi, India.

References

[1] C. C. Aggarwal. A Framework for Diagnosing Changes in Evolving Data streams. In *Proceedings of ACM SIGMOD*, 2003.

[2] M. Baena-Garcia, J. del Campo-Avila, R. Fidalgo, A. Bifet, R. Gavalda, and R. M. Bueno. Early Drift Detection Method. In *Proceedings of the Fourth International Workshop on Knowledge Discovery from Data Streams with ECML/PKDD*, pages 77–86, 2006.

[3] V. Bhatnagar, S. Kaur, and A. Chaturvedi. Gradation Framework for Anomaly Detection in Streams. *ISAST Transactions on Intelligent System*, 1:55–63, 2008.

[4] H. Bulut, B. Yesilata, and M. Irfan. Trend Analysis for Examining the Interaction between the Atatrk Dam Lake and its Local Climate. *International Journal of Natural and Engineering Sciences*, 1(3):115–123, 2008.

[5] K. Chen and L. Liu. Detecting the Change of Clustering Structure in Categorical Data Streams. In *Proceedings of SDM*, pages 502–506. SIAM, 2006.

[6] C. H. Cheng, A. W. Fu, and Y. Zhang. Entropy Based Subspace Clustering for Mining Numerical Data. In *Proceedings of the Fifth ACM SIGKDD*, pages 84–93. ACM, 1999.

[7] R. Choudhary, S. Mehta, and A. Bagchi. On Quantifying Changes in Temporally Evolving Dataset. In *Proceedings of the Seventeenth CIKM*, pages 1459–1460. ACM, 2008.

[8] G. Forman. Tackling Concept Drift by Temporal Inductive Transfer. In *Proceedings of the Twenty Ninth SIGIR*, pages 252–259. ACM, 2006.

[9] L. Golab and M. Ozsu. Issues in Data Stream Management. *SIGMOD Record*, 32(2), 2003.

[10] M. Kendall. *Kendall's Advanced Theory of Statistics*. Arnold, 2004.

[11] M. M. Lazarescu, S. Venkatesh, and H. H.Bui. Using Multiple Windows To Track Concept Drift . *Intelligent Data Analysis*, 8(1):29–59, 2004.

[12] X. Li, P. S. Yu, B. Liu, and S. K. Ng. Positive Unlabeled Learning for Data Stream Classification. In *Proceedings of the Ninth SDM*. SIAM, 2009.

[13] T. Y. Lin, Y. Xie, A. Wasilewska, and C. J. Liau, editors. *Data Mining: Foundations and Practice*, volume 118 of *Studies in Computational Intelligence Series*. Springer, 2008.

[14] H. R. Neave and P. L. Worthington. *Distribution Free Tests*. Unwin Hyman Ltd., London, UK, 1998.

[15] N. H. Park and W. S. Lee. Statistical Grid-based Clustering over Data streams. *ACM SIGMOD Record*, 33:32–37, 2004.

[16] N. H. Park and W. S. Lee. Cell trees: An Adaptive Synopsis structure for Clustering Multidimensional On-line Data Streams. *Journal of Data and Knowledge Engineering*, 63:528–549, 2007.

[17] S. M. Ross. *Stochastic Process*. John Wiley, second edition, 2004.

[18] P. K. Sen. Estimate of the Regression Coefficient based on Kendall's tau. *Journal of Americal Statistical Association*, 63:1379–1389, 1968.

[19] M. Spillopoulou, I. Ntoutsi, Y. Theodoridis, and R. Schult. The MONIC Framework for Cluster Transistion Detection. In *ACM SIGKDD*, 2006.

[20] E. J. Spinosa, A. P. Carvalho, and J. Gama. OLINDDA: A Cluster-based approach for Detecting Novelty and Concept drift in Data Streams. In *Proceedings of SAC*, pages 448–452, 2007.

[21] University of California at Irvine. UCI Machine Learning Repository. http:/www.kdd.ics.uci.edu/.

[22] G. Widmer and M. Kubat. Learning in the Presence of Concept Drift and Hidden Context. *Machine Learning*, 23(1):69–101, 1996.

[23] D. H. Widyantoro and J. Yen. Relevant Data Expansion for Learning Concept drift From Sparsely Labeled Data. *TKDE*, 17(3), 2005.