On Tracking Behavior of Streaming Data: An Unsupervised Approach

Niloofar Mozafari mozafari@cse.shirazu.ac.ir
Department of Computer Science and Engineering
and Information Technology
Shiraz University
Shiraz, Iran

Sattar Hashemi s_hashemi@shirazu.ac.ir
Department of Computer Science and Engineering
and Information Technology
Shiraz University
Shiraz, Iran

Ali Hamzeh ali@cse.shirazu.ac.ir
Department of Computer Science and Engineering
and Information Technology
Shiraz University
Shiraz, Iran

Abstract

In the recent years, data streams have been in the gravity of focus of quite a lot number of researchers in different domains. All these researchers share the same difficulty when discovering unknown pattern within data streams that is concept change. The notion of concept change refers to the places where underlying distribution of data changes from time to time. There have been proposed different methods to detect changes in the data stream but most of them are based on an unrealistic assumption of having data labels available to the learning algorithms. Nonetheless, in the real world problems labels of streaming data are rarely available. This is the main reason why data stream communities have recently focused on unsupervised domain. This study is based on the observation that unsupervised approaches for learning data stream are not yet matured; namely, they merely provide mediocre performance specially when applied on multi-dimensional data streams.

In this paper, we propose a method for **Track**ing **Ch**anges in the behavior of instances using **C**umulative **D**ensity **F**unction; abbreviated as *TrackChCDF*. Our method is able to detect change points along unlabeled data stream accurately and also is able to determine the trend of data called *closing* or opening. The advantages of our approach are three folds. First, it is able to detect change points accurately. Second, it works well in multi-dimensional data stream, and the last but not the least, it can determine the type of change, namely *closing* or *opening* of instances over the time which has vast applications in different fields such as economy, stock market, and medical diagnosis. We compare our algorithm to the state-of-the-art method for concept change detection in data streams and the obtained results are very promising.

Keywords: Data Stream, Trend, Concept Change, Precision, Recall, F1 Measure, Mean Delay Time.

1. INTRODUCTION

Recently, data streams have been extensively investigated due to the large amount of applications such as sensor networks, web click streams and network flows [1], [2], [3], [4], [5]. Data stream is an ordered sequence of data with huge volumes arriving at a high throughput that must be analyzed in a single pass [6], [7]. One of the most important challenges in data streams and generally in many real world applications is detecting the concept change [6]. In general, the process of transition from one state to another is known as the concept change [8]. In data streams where data is generated form a data generating process, concept change occurs when the distribution of the generated data changes [9], [10].

The problem of concept change detection in time-evolving data has been explored in many previous researches [15], [16], [17], [24]. They are mainly focused on labeled data streams, but nowadays, data streams consist of unlabeled instances and rarely the assumption of having data label is realistic. However, there is some respectable works to detect concept changes in unlabeled data streams [8], [10], [24] and the existing approaches merely offer a mediocre performance on data stream having high dimension. So, in this paper, we are trying to propose a new method for Tracking Changes in the behavior of instances using Cumulative Density Function; abbreviated as *TrackChCDF*. It is able to detect change points in unlabeled data streams accurately and also it can track the behavior of instances over the time. To briefly the advantages of our approach are three folds. First, it is able to detect change points accurately. Second, it works well in multi-dimensional data stream, and the last but not the least, it can determine the type of change, namely *closing* or opening of instances over the time which has vast applications in different fields such as economy, stock market, and medical diagnosis. We compare our algorithm to the state-of-the-art method for concept change detection in data streams and the obtained results are very promising.

The reminder of this paper is organized as follows: we outline the previous works in Section 2. Section 3 presents the proposed algorithm. In Section 4, we report the experimental results and the paper concludes with Section 5.

2. RELATED WORK

In general, change is defined as moving from one state to another state [8]. There are some important works to detect changes where some of them detect changes with statistical hypothesis testing and multiple testing problems [11]. In the statistical literature, there are some works for change point detection [12]. However, most of the statistical tests are parametric and also needs the whole data to run [13], [14]. These methods are not applicable in the data stream area, because they require storing all data in memory to run their employed tests [14].

Popular approaches for the concept change detection uses three techniques including (1) sliding window which is adopted to select data points for building a model [15], [16]. (2) Instance weighting which assumes that recent data points in window are more important than the other [17], [18]. (3) Ensemble learning which is created with multiple models with different window sizes or parameter values or weighting functions. Then, the prediction is based on the majority vote of the different models [19], [20], [21], [22]. Both sliding window and instance weighting families suffer from some issues: First, they are parametric methods; the sliding window techniques require determining window size and instance weighting methods need to determine a proper weighting function. Second, when there is no concept change in the data stream for a long period of time, both of sliding window and instance weighting methods would not work well because they do not take into account or give low weights to the ancient instances [10]. The ensemble methods try to overcome the problems that sliding window and instance weighting are faced with by deciding according to the reaction of multiple models with different window sizes or parameter values or weighting functions. However, these techniques need to determine the number of models in the ensemble technique.

Another family of concept change detection methods is based on density estimation. For example, Aggarwal's method [23] uses velocity density estimation which is based on some heuristics instead of classic statistical changes detectors to find changes. As another major works in this family, we could mention Kifer's [24] and Dasu's works [25] which try to determine the changes based on comparing two probability distributions from two different windows [24], [25]. For example, in [24] the change detection method based on KS test determines whether the two probability density estimations obtained from two consequent different windows are similar or not.

However, this method is impractical for high dimensional data streams and also needs to determine the proper window size. Dasu et al. propose a method for change detection which is related to Kulldorff's test. This method is practical for multi-dimensional data streams [25]. However, this method relies on a discretization of the data space, thus it suffers from the curse of dimensionality.

Another major work is proposed by Ho et al. [8], [10]. In Ho's method, upon arrival of new data point, a hypothesis test takes place to determine whether a concept change has been occurred or not. This hypothesis test is driven by a family of martingales [8] which is based on Doob's Maximal Inequality [8]. Although Ho's method detects changes points accurately, it can only detect some types of changes to be detailed in Section 4. Moreover, it is not able to determine the type of changes.

3. PROPOSED METHOD

The problem of concept change detection in time-evolving data is formulated as follows: we are given a series of unlabeled data points *D = {z_1, z_2, …, z_n}*. *D* can be divided into *s* segments D_i where *{i=1, 2 , .., s}* that follow different distribution. The objective of a detection approach is basically to pinpoint when the distribution of data changes along unlabeled data stream.
In our proposed method, in the first step, we rank instances according to their differences to a mean of instances. It determines how much a data point is different from the others. Namely, the Euclidean distance of each instance with mean of instances is calculated using the Equation 1:

$$D_i(Z, z_n) = \sum_{i=1}^{m} \left| z_i - C(Z \cup \{z_n\}) \right| \tag{1}$$

Where m is the number of dimension and $C(Z \cup \{Z_n\})$ indicates the mean of the union of previous instances and new received instance z_n. The value of *D* is high when the new instance is farther from the representative of previous ones. Then, we calculate the number of instances that are farther than the last received instance from the mean of instances.

$$G = \frac{\#\{i : D_i > D_n\}}{n} \tag{2}$$

In this formula, *D* is obtained using Equation 1 and *n* indicates the number of instances. The changes of *G* toward higher values can be deemed as data points are running away from their representative. In contrast, having data close to their representative conveys that the sequences of *G* are approaching smaller values. In other words, the sequences of *G* approach to the smaller values upon widening of data distribution. Conversely, these sequences approach to the higher value when the data distribution is going to be contracted. To be illustrative, suppose that instances are produced from a distribution function whose standard deviation gets high near 2000. We calculated values of *G* for these instances. As Figure 1 shows, the sequences of *G* approach to the smaller values near 2000. We repeat this experiment for the case of *closing* instances. Figure 3 illustrates the values of *G* when the instances get closed near 2000. According to these figures if the sequences of *G* get high value in an interval of time, it means a change is occurred and the type of change is *closing* and it vice versa for the case of opening. Thus we need a temporary to store sequence of *G* to track the behavior of instances over the time. To do that, in the second step, we calculate Cumulative Density Function (CDF) of *G*. Figures 2 and 4 respectively illustrate CDF of *G* when the instances either get opened or get closed around instance 2000. As these figures illustrate if the sequences of G approach to the smaller value, the Cumulative Density Function (CDF) of *G* gets a small value and vice versa. Thus we can use this property to detect change points and also determine the type of change, namely *closing* or opening. In other words, this property can help tracking the changes in the behavior of instances over the time.
In the next step, we calculate *CH* using Equation 3. In this formula *G* is obtained using Equation 2 and t_w is the smoothing parameter.

$$CH_n = CH_{n-1} + \left(\sum_{t}^{t_c} G_t + \sum_{t}^{t_c} G_{t-t_w}\right) / t_w \tag{3}$$

If the difference of two consecutive *CH* is greater than a threshold, it means the distribution of data is going to be closed over the time, namely, there is a change in the data stream and the type of change is *closing*. If the difference of two consecutive *CH* is smaller than a threshold, it means the distribution of data would be opened, namely, there is a change in the data stream and the type of change is *opening*.

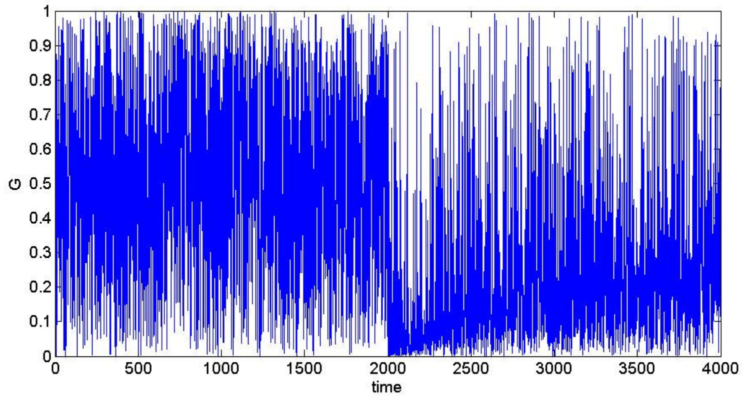

FIGURE 1: the values of *G* when the instances get open near 2000. The horizontal and vertical axes respectively show time and *G* that is calculated by Formula 2. The sequences of *G* approach to the smaller values when the data distribution would be opened.

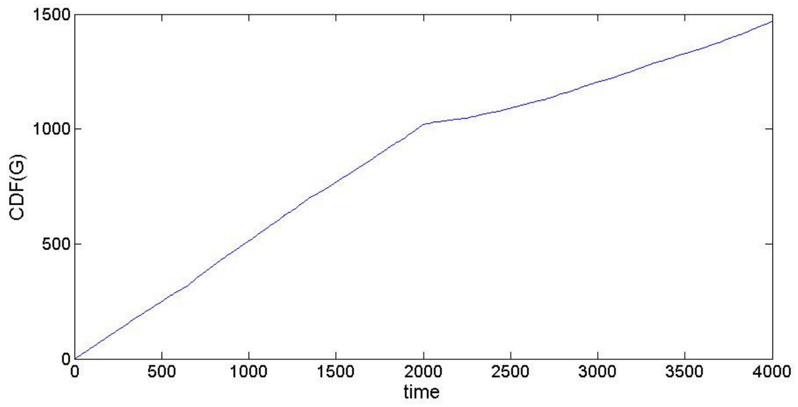

FIGURE 2: the values of CDF of *G* when the instances get open near 2000. The horizontal and vertical axes respectively show time and CDF of *G*. If the two successive slopes of the sequences of CDF are smaller than a threshold, it means the distribution of data would be opened, namely, there is a change in the data stream and the type of change is *opening*.

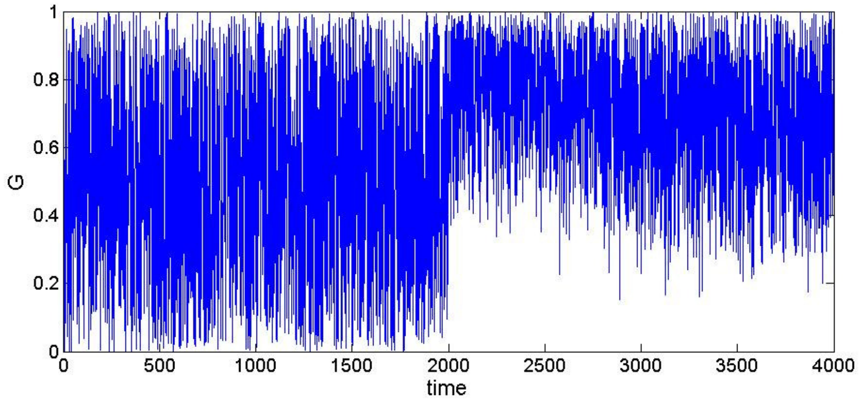

FIGURE 3: the values of *G* when the instances get close near 2000. The horizontal and vertical axes respectively show time and *G* that is calculated by Formula 2. The sequences of *G* approach to the higher values when the data distribution would be closed.

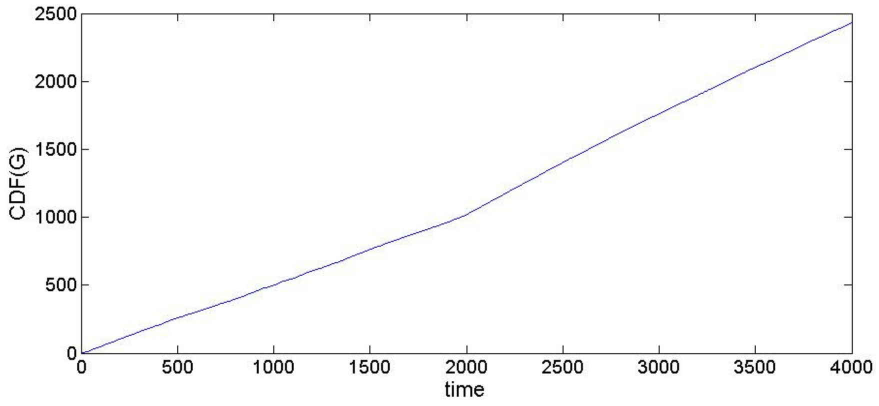

FIGURE 4: the values of CDF of *G* when the instances get close near 2000. The horizontal and vertical axes respectively show time and CDF of *G*. If the two successive slopes of the sequences of CDF are greater than a threshold, it means the distribution of data would be closed, namely, there is a change in the data stream and the type of change is *closing*.

To have a general overview of our algorithm for better understanding, the main steps of our algorithm can be capsulated in the following steps:
The Euclidean distance of each instance to the mean of pervious seen instances is calculated using Equation (1).
G; the number of instances that their distance to the mean is greater than the distance of the last received instance to the mean of instances; is counted using Equation (2).
CH is calculated using Equation (3).
If the difference of two successive *CH* is greater than a threshold, there exists a change and the type of change is *closing*. If the difference of two consecutive *CH* is smaller than a threshold, there exists a change and the type of change is opening.

4. EXPERIMENTAL RESULTS AND DISCUSSION

This section composes of two subsections, precisely covering our observation and analysis. The first subsection presents the experimental setup and description of used evaluation measures. The latter one presents and analyses the obtained results.

4.1 Experimental Setup

This section introduces the examined data set and the used evaluation measures respectively.

Data Set. To explore the advantages of *TrackChCDF*, we conduct our experiments on a data set which was used previously in Ho's work [8], [10]. In this data set, change is defined as the change in the generating model. This change is simulated by varying the parameters of the function generates the data stream. According to this definition, we construct a data set with 20000 instances that changes occur after generating each 2000 instances. Thus, this data set includes ten segments. In each segment, instances are sampled from a Gaussian distribution. We change standard deviation after each 2000 instances. Therefore, this data set has nine change points in instances 2000, 4000, 6000, 8000, 10000, 120000, 140000, 160000, and 180000. Figure 5 illustrates the behavior of data streams over the time. As this figure shows the type of changes are {*opening, opening, opening, closing, closing, opening, closing, closing, closing*}.

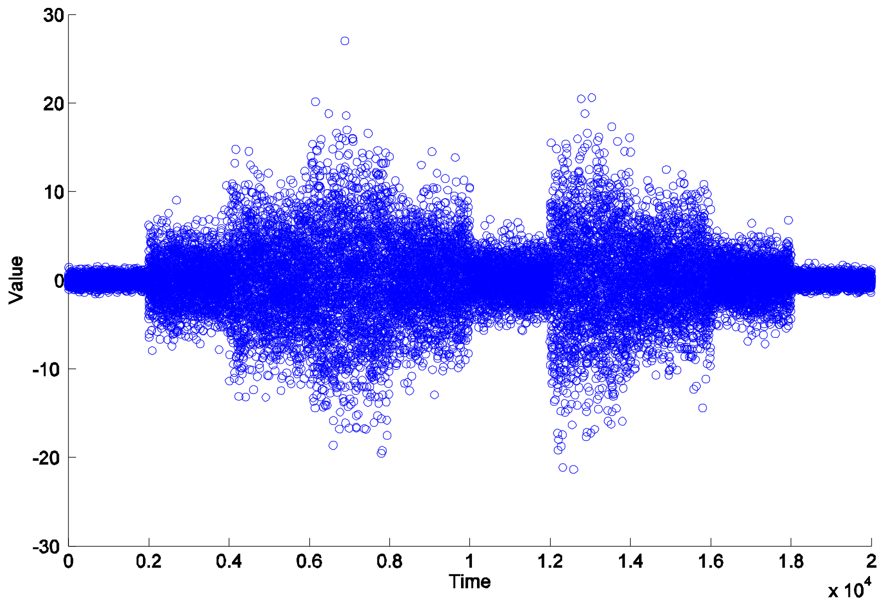

FIGURE 5: the behaviour of instances in the stream data. The X axis shows time and the Y axis indicates the value of instance in each time.

Evaluation Measures. We assess our method with three measurements criterion which is well known in the context [8], [10]: 1) the precision, that is the number of corrected detections divided by the number of all detections. 2) Recall that is the number of corrected detections divided by the number of true changes. 3) F1, that represents a harmonic mean between recall and precision. Following is the definitions of these measurements.

$$\text{Precision} = \frac{\text{Number of Corrected Detections}}{\text{Number of Detections}} \quad (3)$$

$$\text{Recall} = \frac{\text{Number of Corrected Detections}}{\text{Number of True Changes}} \quad (4)$$

$$\text{F1} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \quad (5)$$

As the precision and recall measures are related by F1, if F1 measure gets higher value, we can ensure that precision and recall are reasonably high.

4.2 Results and Discussion

In this section, the results of applying our method on the studied data set are analyzed. As mentioned previously, this data set is created using ten overlapping Gaussian distributions. To apply concept change in this data set, we change the standard deviation of data after generating each 2000 instances. We compare our method with Ho's method [8], [10]. Table I shows the result of applying Ho's method and our method on this data set. To be more accurate, we ran this experiment 50 times and evaluated our method with three measurements; precision, recall, and F1.

TABLE I: comparison between Ho's method and the proposed method on num-ds data set.

Ho's Approach			TrackChCDF		
Precision	Recall	F1-measure	Precision	Recall	F1-measure
0.9850	0.3357	0.5006	0.9728	0.8556	0.9104

The Precision measure of *TrackChCDF* is slightly less than the precision of Ho's method and the Recall measure is significantly higher than Ho's method because our method detects all the change points whereas Ho's method detects smaller number of them. Also, as F1 is the balance between Recall and Precision; we can ensure that Precision and Recall are reasonably high if F1 gets high value. As *TrackChCDF* has the higher F1 in comparison to Ho's method, we can conclude that the proposed method certainly detects the true change points in addition to a few number of false change points. But, according to the value of precision, these false change points are not extortionary. Ho's method only detects those change points where data get away from the mean of data distribution. In Ho's method, changes can be detected when p_values [10] get small. The p_values will be small when the number of strangeness data [8] increases over coming numerical data, this increasing occurs when data gets away from the centre of data, i.e. the mean of data distribution. Therefore, when data is close to the centre of data, the number of strangeness data decrease, the p_value increases and the martingale value [8] would not be large enough to detect these kinds of changes. In contrast, such changes can be detected in *TrackChCDF* because it analyzes the behavior of *G* sequences by obtaining its CDF. If the sequences of G approach to the smaller value, the CDF of G sequences gets a small value and vice versa.
To explore the ability of *TrackChCDF*, when the dimension of instances increases, we increase the number of dimensions in the studied data set and investigate the behavior of our method against Ho's method. Figure 6 illustrates the Precision of *TrackChCDF* and Ho's method in different dimensions. The horizontal axis shows the dimension of data set and the vertical one represents the mean of Precision measurements in 50 independent runs respectively. Both methods have high Precision. It means that when they alarm the existence of a change point, it is a true change point and they have low false alarm in average.

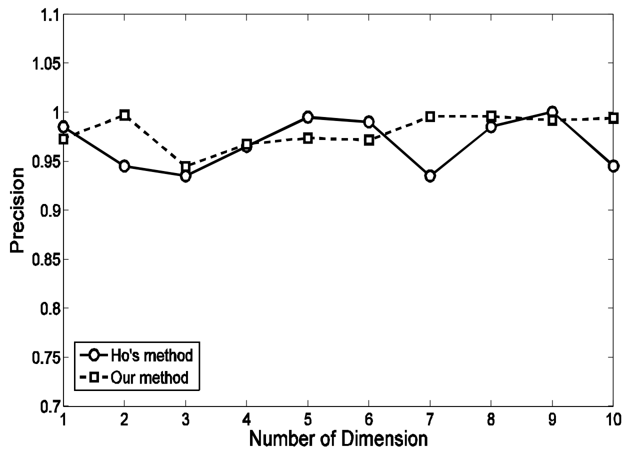

FIGURE 6: the accuracy of *TrackChCDF* and Ho's method in different number of dimensions. The horizontal axis shows dimension of data set and the vertical one represents the mean of precision measurements in 50 run respectively.

Figure 7 illustrates the Recall of *TrackChCDF* in comparison to Ho's method in different dimensions. The horizontal axis shows the dimensions of data set and the vertical one represents the mean of Recall measurements in 50 independent runs respectively. Ho's method can only detect those change points where the data get away from the mean of data distribution. In other words, it just detects the change points when the underlying data distribution will be open along the time. Thus, it detects the first three change points and it cannot detect the change points when the distribution of data gets near the mean of data, namely closing change types. So, Ho's method cannot detect the fourth and fifth change points. Also, that method cannot detect the sixth change point in the instances of 12000 in spite of its type, *opening*, because this change occurs after two closing change points taking place in instances 8000 and 10000 respectively. It should be mentioned that in Ho's method, changes can be detected when p_values [10] get small. The p_values will be small when the number of strangeness data [8] increases through coming numerical data, this increasing occurs when data gets away from the center of data, i.e. the mean of data distribution. Therefore, when two closing type changes occur sequentially and after that an *opening* change happens, in Ho's method the two closing change causes the p_value sequences to get high value and the next *opening* change is slow down the p_value sequences but this reduction is not enough to be able to show this type of change in this manner. In contrast, *TrackChCDF* can detect such change precisely because it analyzes the behavior of G sequences by obtaining its CDF. Consequently, we can easily monitor the behavior of data distribution, including *opening* and closing changes happens in any combination, along the time. If the sequences of G approach to the smaller value, the CDF of G gets a small value and vice versa. Therefore, *TrackChCDF* has a higher Recall in comparison to Ho's method.

As Precision and Recall are related by F1, if F1 measure gets high value, we can ensure that Precision and Recall are reasonably high. Figure 8 illustrates the mean of F1 in 50 time experiments. Our method has higher F1 in comparison to Ho's method because Recall of our method is significantly higher that Ho's method.

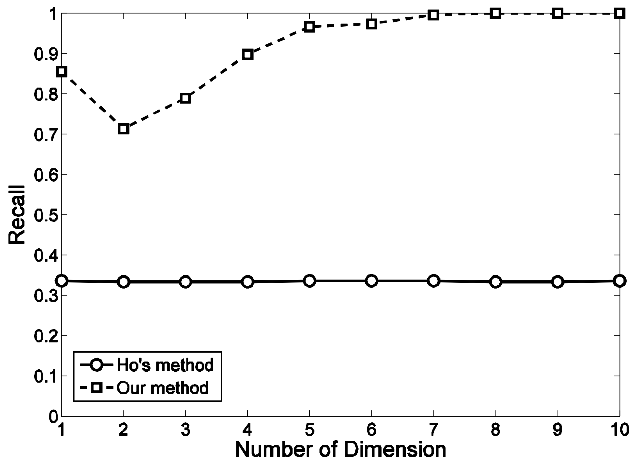

FIGURE 7: the accuracy of *TrackChCDF* and Ho's method in different number of dimensions. The horizontal axis shows dimension of data set and the vertical one represents the mean of recall measurements in 50 run respectively.

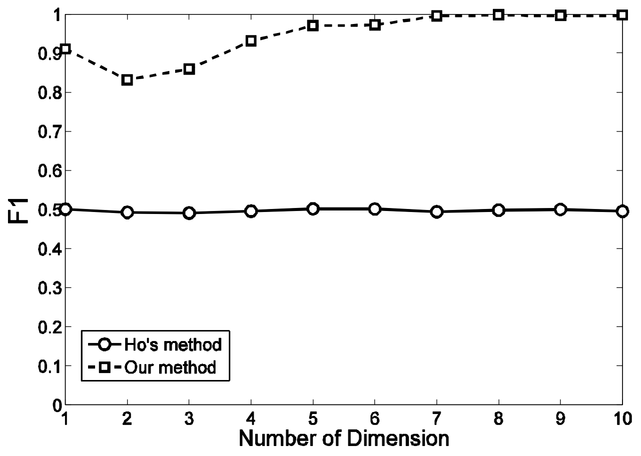

FIGURE 8: the accuracy of *TrackChCDF* and Ho's method in different number of dimensions. The horizontal axis shows dimension of data set and the vertical one represents the mean of F1 measurements in 50 run respectively.

To show the effectiveness of our method in the other point of view, we compare our method with Ho's method by calculating the mean delay time. This measure, i.e. delay time, shows the difference between the true time of occurrence of a change and the time in which ours can alarm an existence of that change. In other words, it shows how much delay time the underlying method has. Figure 9 shows the mean delay time of *TrackChCDF* in comparison to Ho's method. It should be mentioned that, for all ten experiments in each method, we perform 50 trials for each of these ten experiments. In this figure, we observe the mean delay time decreases with increasing the number of dimensions. With increasing the number of dimensions, the amount of changes increases because change applies in each dimension.

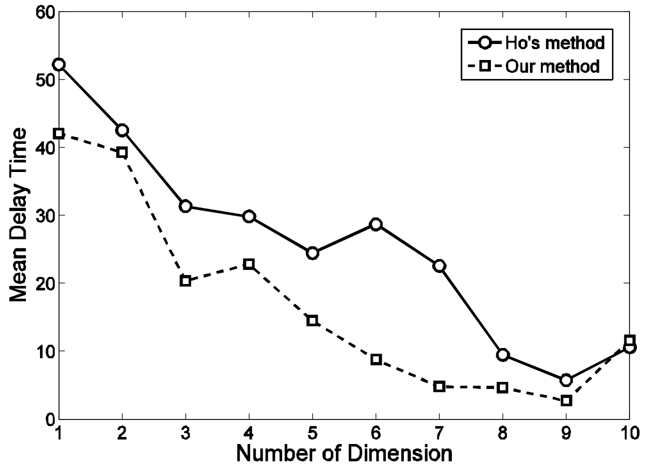

FIGURE 9: the accuracy of *TrackChCDF* and Ho's method in different number of dimensions. The horizontal axis shows dimension of data set and the vertical one represents the mean of Mean delay time in 50 run respectively.

Finally, it could be said that our method is able to determine the type of change in data stream. If the difference of two consecutive slopes of CDF is greater than a threshold, it means that there is a change and also this indicates that instances are going to be closed along the time. Also, if the difference of two successive slopes of CDF is smaller than a threshold, there exists a concept change in the distribution of data generating model and the instances are going to be opened along the time. According to this property, unlike Ho's method, *TrackChCDF* is able to determine not only the existence of change but also the type of such change, being closing or *opening* type.

5. CONSLUSION & FUTURE WORK

Recently data streams have been attractive research due to appearance of many applications in different domains. One of the most important challenges in data streams is concept change detection. There have been many researchers to detect concept change along data stream, however, majority of these researches devote to supervised domain where labels of instances are known a priori. Although data stream communities have recently focused on unsupervised domain, the proposed approaches are not yet matured to the point to be relied on. In other words, most of them provide merely a mediocre performance specially when applied on multi-dimensional data streams. In this paper, we propose a method for detecting change points along unlabeled data stream that is able to determine trend of changes in data streams as well. The abilities of our model can be enumerated as: (1) it is able to detect change points accurately. (2) It is able to report the behavior of instances along the time. (3) It works well in multi-dimensional data stream. We compare our algorithm to the state-of-the-art method for concept change detection in data streams and the obtained results are very promising. As a future work, we will incorporate our method into data clustering schemes.

6. REFERENCES

1. B. Babcock, S. Babu, R. Datar, R. Motwani and J. Widom. “Models and Issues in Data Stream Systems”, *in proceedings of ACM Symp, Principles of Databases Systems (PODS)*, pp. 1-16, 2002.

2. W. Fan. “Systematic Data Selection to Mine Concept Drifting Data Streams”, *in Proceedings of ACM SIGKDD*, pp. 128-137, 2004.

3. X. Liu, J.Guan, P. Hu. “Mining Frequent Closed Item Sets from a Landmark Window Over Online Data Streams”, *in journal of computers and mathematics with applications*, vol. 57, pp. 927-936, 2009.

4. C.C.Aggarwal, J. Han, J. Wang, P.S. Yu. “On Demand Classification of Data Streams”, *in proceedings of ACM SIGKDD*, pp. 503-508, 2004.

5. T. Jiang, Y. Feng, B. Zhang. “Online Detecting and Predicting Special Patterns over Financial Data Streams”, *in Journal of Universal Computer Science*, vol. 15, pp. 2566-2585, 2009.

6. J. Han, M. Kamber. “Data Mining: Concepts and Techniques”, *Morgan Kaufmann*, 2001.

7. O. Nasraoui, C. Rojas. “Robust Clustering for Tracking Noisy Evolving Data Streams”, *in Proceedings of Sixth SIAM International Conference of Data Mining (SDM)*, 2006.

8. S. S. Ho, H. Wechsler. “A Martingale Framework for Detecting Changes in Data Streams by Testing Exchangeability”, *in IEEE transactions on pattern analysis and machine intelligence*, 2010.

9. S. S. Ho. “A Martingale Framework for Concept Change Detection in Time Varying Data Streams”, *in Proceeding of 22th International Conference on Machine Learning*, L. D. Raedt and S. Wrobel, Eds., ACM, pp. 321–327, 2005.

10. S.-S. Ho, H. Wechsler. “Detecting Changes in Unlabeled Data Streams Using Martingale”, *in Proceeding 20th International Joint Conference on Artificial Intelligence*, M. Veloso, pp. 1912–1917, 2007.

11. P.J. Bickel, K. Doksum, “Mathematical Statistics: Basic Ideas and Selected Topics”, *Holden-Day*, Inc., 1977.

12. E. Carlstein, H. G. Muller, D. Siegmund editors. “Change point problems”, *Institute of Mathematical Statistics*, Hayward, California, 1994.

13. J. Glaz, N. Balakrishnan Editors. “Scan Statistics and Applications”, Boston, 1999.

14. J. Glaz, J. Naus, S. Wallenstein. “Scan Statistics”, Springer, New York, 2001.

15. R. Klinkenberg, T. Joachims. “Detecting Concept Drift with Support Vector Machines”, *in Proceedings of 17th International Conference on Machine Learning*, P. Langley, Ed. Morgan Kaufmann, pp. 487–494, 2000.

16. G. Widmer, M. Kubat. “Learning in the Presence of Concept Drift and Hidden Contexts”, *in Machine Learning*, vol. 23, no. 1, pp. 69–101, 1996.

17. R. Klinkenberg. “Learning Drifting Concepts: Examples Selection VS Example Weighting”, *in Intelligent Data Analysis, Special Issue on Incremental Learning Systems capable of dealing with concept drift*, vol. 8, no. 3, pp. 281–300, 2004.

18. F. Chu, Y. Wang, C. Zaniolo. “*An Adaptive Learning Approach for Noisy Data Streams”*, *in proceedings of 4th IEEE international conference on Data Mining. IEEE Computer Society*, pp. 351–354, 2004

19. J. Z. Kolter, M. A. Maloof. “Dynamic Weighted Majority: A New Ensemble Method for Tracking Concept Drift*”, in proceedings of 3th IEEE international conference on Data Mining, IEEE Computer Society*, pp. 123–130, 2003.

20. H. Wang, W. Fan, P. S. Yu, J. Han. “Mining Concept Drifting Data streams Using Ensemble Classifiers”, *in proceedings of 9th ACM SIGKDD international conference on Knowledge Discovery and Data Mining*, L. Getoor, T. E. Senator, P. Domingos, and C. Faloutsos, Eds. ACM, pp. 226–235, 2003.

21. M. Scholz, R. Klinkenberg. “Boosting Classifiers for Drifting Concepts”, *in Intelligent Data Analysis*, vol. 11, no. 1, pp. 3-28, 2007.

22. O. Bousquet, M. Warmuth. “Tracking a Small Set of Experts by Mixing Past Posteriors”, *in Journal of Machine Learning Research*, vol. 3, pp. 363-396, 2002.

23. C. C. Aggarwal. “A framework for Change Diagnosis of Data Streams”, *in proceedings of ACM SIGMOD international conference on Management of Data*, pp. 575–586, 2003.

24. D. Kifer, S. Ben-David, J. Gehrke. “Detecting Change in Data Streams”, *in proceedings of 13th international conference on Very Large Data Bases*, M. A. Nascimento, M. T. O¨ zsu, D. Kossmann, R. J. Miller, J. A. Blakeley, and K. B. Schiefer, Eds. Morgan Kaufmann, pp. 180–191, 2004.

25. T. Dasu, S. Krishnan, S. Venkatasubramanian, K. Yi. “An Information Theoretic Approach to Detecting Changes in Multi Dimensional Data Streams”, in Interface, 2006.