Concept Drift Detection Based on Pre-Clustering and Statistical Testing

Jones Sai-Wang Wan, Sheng-De Wang

Department of Electrical Engineering, National Taiwan University, Taiwan
{r04921087, sdwang}@ntu.edu.tw

Abstract

Stream data processing has become an important issue in the last decade. Data streams are generated on the fly and possibly change their data distribution over time. Data stream processing requires some mechanisms or methods to adapt to the changes of data distribution, which is called the concept drift. Concept drift detection can be challenging due to the data labels are not known. In this paper, we propose a drift detection method based on the statistical test with clustering and feature extraction as preprocessing. The goal is to reduce the detection time with principal component analysis (PCA) for the feature extraction method. Experimental results on synthetic and real-world streaming data show that the clustering preprocessing improve the performance of the drift detection and feature extraction trade-off an insignificant performance of detection for speedup for the execution time.

Keywords: Concept drift, Stream data mining, Drift detection, Unsupervised

1 Introduction

Over the decades, stream data mining for knowledge discovery has drawn much attention in real-world applications. Stream data mining is a kind of real-time data analysis, where data is produced continuously at a rapid rate. Stream data mining is challenging because of the data stream characteristics, such as infinite length, limited availability of data labels or delayed labeling, concept drifts, and concept evolution [1].

According to Schlimmer et al. [2], concept drifts occur when the data distribution change over time in dynamic environments: the target class or the concept evolves within the feature space crosses previously defined decision boundaries of the classifier. Concept drift could be defined as $P_t(x \mid y) \neq P_{t+n}(x \mid y)$, where x represents a data instance in the data stream, y is the target class and concept drift occurred between time t and $t+n$ [3]. Therefore, probabilities of the class may change over time. For example, weather forecast models changes with the seasons, and stock market prediction models may change over time because of political or economy news. Those changes in feature space may lead to the decline of the classification accuracy.

There are two types of drifts: virtual drift and real concept drift [4]. Virtual drift refers to the drift that only changes in distribution of data, and real drift refers to the drift that changes the target concept. Moreover, drifts can also be separated according to the speed of change: abrupt drifts when concept drifts happen in sudden like switch; gradual drifts when concept drifts occur slowly in a long period as Figure 1 shows. Also, a previous concept from the stream may reoccur, known as the reoccurring concept.

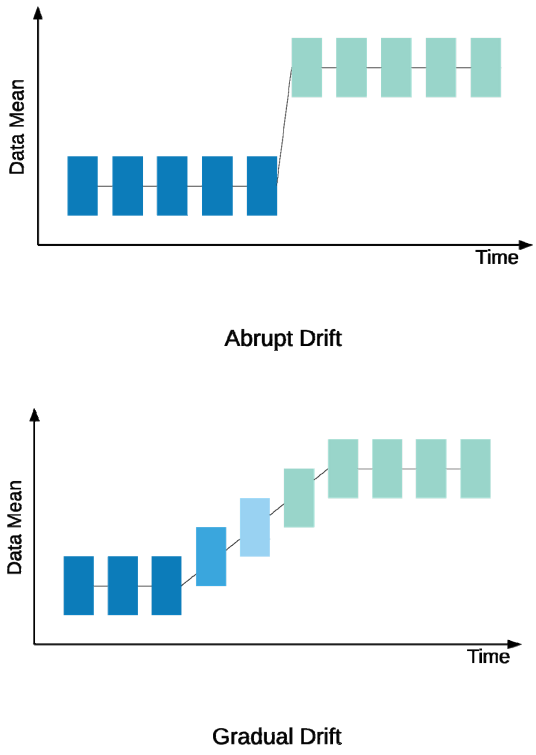

Figure 1. Example of different type of drifts according to their drifting speed

When concept drifts occurred, we should update the

*Corresponding Author: Sheng-De Wang; E-mail: sdwang@ntu.edu.tw
DOI: 10.3966/160792642021032202020

classifier models to uphold the prediction accuracy of the classification. There are two main approaches to handling concept drifts: active approaches and passive approaches. Active approaches rely on the concept drift detection to trigger updates of the classifier. On the other hand, passive approaches keep updating the classifier constantly. Passive approaches can keep the model up-to-date, while they might result in a higher computational cost for updating the model unnecessary when the speed of drift is slow.

Moreover, in many real-world applications, data labels are not available or data labels are available in an extreme cost or high delay. Therefore, unsupervised drift detection methods are required although the supervised detection methods have more accurate detection rate.

The objective of this paper is to propose an unsupervised concept drift detection method based on the feature distributions in data with clustering algorithm as preprocessing. Also, it lessens the execution time of drift detection with feature extraction techniques.

This paper is organized as follows: In Section 0, we give a brief survey of related work. Section 0 explains our proposed work in detail. In Section 0, we evaluate the method on both synthetic and real-world datasets. Finally, Section 0 presents our conclusions and proposes the future work.

2 Related Work

In this section, we are going to review some existing approaches to handling concept drifts. When concept drifts occurred, we should update the classification model to keep the model adapted to the current concept. There are two main approaches to handling concept drifts: active approaches and passive approaches.

2.1 Passive Approaches

Via updating the model incrementally, passive approaches have two most common methods: Adaptive Windowing (ADWIN) [5] and the Very Fast Decision Tree (VFDT) proposed by Domingos and Hulten [6].

ADWIN is using sliding window techniques to detect drifts. By enlarging the detection window, a drift is detected when the average difference between two sub-windows exceeds the threshold ε_{cut}, which is defined by the Hoeffding bound that was evaluated from the Hoeffding's inequality. Although ADWIN required the drift detectors to perform the tests along all the dimensions of data, it provides a major architecture to detect drifts by comparing data in windows from the stream.

In VFDT, a decision tree is grown by using the Hoeffding bound to guarantee the high asymptotic similarity to corresponding batch trees in the stream. The decision tree is constantly updated with the latest concept since the new leaves are generated with the labeled data from the current window. It also discards the outdated-concept data in a constant rate. VFDT shows its adaptability to the dynamic environment in streaming data mining.

2.2 Active Approaches

Active approaches to handling concept drifts are to update the classifier only when the change detector is triggered by a concept drift. When all class labels are available, we prefer supervised approaches to detect concept drifts, as supervised approaches can achieve more accurate detection than an unsupervised one. Drift Detection Method (DDM) detects the drifts based on the significantly increased prediction error rate of the classification model [7]; Early Drift Detection Method (EDDM) is similar to DDM with considering the error distance between the ground-truth labels and the predictions [8]; Statistical Test of Equal Proportions (STEPD) detects drifts by comparing the recent prediction accuracy and the overall prediction accuracy with the chi-square test: A drift is detected when the observed significance level is below the threshold [9].

Unsupervised concept drift detection methods, which do not depend on the classification models but base on the feature distributions of data, work properly when accessing the data label is unavailable. Statistical hypothesis tests like the Kolmogorov-Smirnov test [10], the Page-Hinkley test (PHT) [11-12] are applied to determine whether recent data are within the same concept from the original data. The null distribution of the statistic is calculated under the null hypothesis (H_0) which drew the samples from the same distribution. A drift is detected if the null hypothesis is rejected when the statistic metric exceeds a specified significance level. An unsupervised detection method might lead to a higher false alarm rate because of the lack of label information, which also means that the detection accuracy is also decreasing.

Souza et. al. [13] propose a framework called Stream Classification Algorithm Guided by Clustering (SCARGC), which consists of a clustering step followed by a classification step applied repeatedly in a closed loop fashion. It clusters the incoming data into certain clusters and compares their centroid distances with the reference centroid from past windows, then labels the data in the whole cluster by a simple nearest neighbor algorithm. Kappa coefficients are employed as the significance test between the classifier predictions after the labeling step. It provides an accurate unsupervised method to detect incremental drifts, although extra computational cost is required for clustering in each window steps.

Fukui [14] uses PHT and DDM as the ensemble change detection method and Self-organizing maps (SOM) as the clustering model for the non-density

based approaches for the concept drift detection. Since the approach is an ensemble method with DDM, it requires the accessibility of labeled data to detect drifts with monitoring the prediction accuracy of the model. Moreover, this approach requires updating the clustering model when a drift is detected. It is time-consuming because training an SOM model is slow. However, using a larger window size and adjusting the detection parameter can be an approach to balance the trade-off between detection delay and false detection.

3 Proposed Method

In this section, we introduce a statistical test with clustering and feature extraction as preprocessing steps for concept drift detection on data streams. The goal of the feature extraction step is to lessen the time in the clustering and the detection test in the case of high dimensional feature spaces. The architecture of the proposed method is shown in Figure 2, where a PCA feature extractor and a clustering method is applied before the task of concept drift detection.

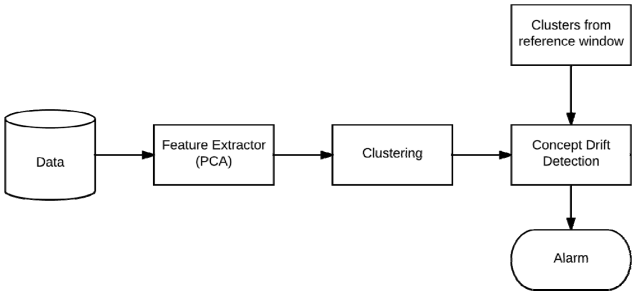

Figure 2. Proposed architecture of drift detection

3.1 Feature Extraction

The time complexity of the clustering and statistical test stages is data-dimension dependent, that is, the processing time increases as the dimensions of the feature space increases. To speed up these stages, feature extraction is applied to reduce the underlying feature dimensions.

The principal component analysis (PCA) is employed in this paper since Kuncheva [15] proved that feature extraction through PCA is favorable for data in the multidimensional stream to detect changes. Other dimension reduction techniques, like linear discriminant analysis (LDA), Independent Component Analysis (ICA), and Multidimensional scaling (MDA) can also be used according to the characteristic of the data stream.

However, feature extraction may result in information loss for classification that affects the accuracy of drift detection, especially in small-dimensional data. Therefore, we decided the feature extraction step is only applied when the reduced dimension is larger than 1.

3.2 Clustering

Since the distribution of data is dynamic in streaming, data may have imbalance in windows. Data imbalance can cause the major class in the window to dominate the concept drift detection. As a result, the drift detection may not sensitive to detect the concept drift in minor class. The goal of data clustering before concept drift detection is to reduce the impact to drift detector performance under imbalanced data. Thus, statistical tests are applied to each group after the unsupervised clustering.

In order to handle imbalance data window in the stream, we choose K-means as the clustering algorithm because of its simple complexity as compared with Affinity propagation (AP). Alternatively, AP does not require to specify or estimate the number of clusters. As a result, AP is more adaptive to the data distributions than K-means. However, in general AP requires a longer time to converge in clustering.

3.3 Drift Detection

By applying the clustering method to the data from the incoming window, the result is referred as the reference window or the reference cluster. The differences of distributions between the current window and the reference window are compared by applying the statistical hypothesis test. The Anderson-Darling (AD) test is used in our drift detection method for the statistical hypothesis test. It is a non-parametric hypothesis test that tests whether two samples come from the same distribution. The AD statistic quantifies a distance between the cumulative distribution functions (CDF) of two samples, which defined as:

$$A^2 = -N - S \tag{1}$$

where,

$$S = \sum_{i=1}^{N} \frac{(2i-1)}{N} [\ln F(Y_i) + \ln(1 - F(Y_{N+1-i}))] \tag{2}$$

F is the cumulative distribution function of the specified distribution. Note that Y_i denotes the ordered data, i=1, N.

The null distribution of the statistic is calculated under the null hypothesis (H_0), where data in windows are drawn from the same distribution. The null hypothesis would be rejected if the statistical return exceeds a specified significance level and it results in a detected concept drift. Therefore, we can use the p-value as the threshold of the drift detection test.

3.4 Data Balancing

Data distribution in windows is dynamic and therefore imbalance data handling is important when those data are used to update the classification model. Under-sampling and over-sampling are common

techniques to handle imbalance data, where under-sampling means removing data from the major class and over-sampling means adding data to the minor class.

We use Synthetic Minority Over-sampling Technique (SMOTE) [16] ensembles with Tomek links [17] as over-sampling followed by under-sampling method proposed in [18]. SMOTE is used to create synthetic data points with taking a feature vector between one of *k* neighbors:

$$x_{new} = x_i + (x'_{neighbor} - x_i) * \delta, \delta \in \{0,1\} \quad \textbf{(3)}$$

In order to remove data from the major class, instances from both classes are chosen to form pairs such that:

$$(x_i, x_j), x_i \in x_{major}, x_i \in X_{minor} \quad \textbf{(4)}$$

and there have no instances, x_k , with:

$$dist(x_j, x_k) < dist(x_i, x_j) \text{ or}$$
$$dist(x_j, x_k) < dist(x_i, x_j) \quad \textbf{(5)}$$

The pairs are defined as Tomek links. By removing Tomek linked point, it cleans up the instances that are noise or near the decision boarder.

As Algorithm 1 shown in Figure 3, initial reference data would be transformed by PCA and divided into clusters with the K-Means clustering algorithm. In the online stage, incoming data are transformed and mapped into clusters with reference PCA and K-Means models. Next, AD-tests are applied on reference and incoming instances in each cluster. Including the classification model, both feature extraction and clustering models are reconstructed when a concept drift is detected. Ninety percent of variance dismissed (K) as the research proposed in [14], and the number of clusters would be equal to the amount of class in labels.

Algorithm 1. Drift Detection Method

Input: Data Stream DS,
Percentage variance dismissed $K \in (0,1)$ (default $K = 0.9$),
Confidence coefficient *a* (default $a = 0.05$),
Cluster number *c* (default *c* = number of class in label),
Window size *w*,
Initial training data $\{X_0, y_0\}$

Output: Drift alarm

1. $PCA(X, K)$:
2. **if** $|X| * (1-K) \geq 1$ **do**
3. **return** transformed X with keeping $(1-K)$ feature dimension
4. **end if**
5. **return** X
6. $driftHandle(X, y)$:
7. Build classifier with $\{X, y\}$
8. Transformed feature space $X' \leftarrow PCA(X, K)$
9. Cluster X' into *c* clusters

Initialization:

10. Build classifier with $\{X_0, y_0\}$
11. Transformed feature space $X'_0 \leftarrow PCA(X_0, K)$
12. Cluster X'_0 into *c* clusters

Streaming:

13. **while** DS is available **do**
14. predict cluster with x'_t to cluster map *c*
15. **for** each cluster **do**
16. $drift_{AD} = ADtest(X'_t, X'_{ref}, a)$
17. **if** $drift_{AD}$ = True **then** #drift detected
18. $driftHandle(X_t, y_t)$ #update classifier/FE/cluster model
19. **end if**
20. **end for**
21. **end while**

Figure 3. Algorithm of drift detection

4 Experimental Evaluation

In this section, we evaluate our proposed concept drift detection method on both synthetic and real-world data streams. We implemented the proposed framework in Python by using the scikit-learn library for the classification and PCA algorithms, and the SciPy library for AD-test in drift detection.

4.1 Datasets

(1) *Synthetic datasets*

a. SEA dataset: is proposed by Street and Kim [19], which is generated 100,000 random points in 3 dimension feature spaces with 2 classes. All features are ranged between 0 and 10 but only the first two features are relevant. Data point could be divided into 10 blocks, each data point in blocks belongs to class 1 if $relevantfeature_1 + relevantfeature_2 \leq threshold$, else belong to class 2. The threshold may drift as 8, 9, 7 and 9.5 repeatedly along 10 blocks, also 10% of noise is inserted into the dataset.

b. Rotating Hyperplanes dataset: is a synthetic data generated using the equation: $\sum_{i=1}^{d} a_i x_i = a_0$, as Fan proposed in 2004 [20]. Data points fulfill $\sum_{i=1}^{d} a_i x_i \leq a_0$ as positive, others would classify as negative. It can simulate concept drift in high dimensional feature space with parameter *k* specifies the total number of dimensions and *t* would be the magnitude of the change of weight a_0 .

c. Benchmark dataset: is proposed by Souza in 2015 [13], containing 16 datasets with incremental drift over time. More detail description of these datasets can be found in the paper website.

(2) *Real-World datasets*

a. Electricity Market (Elec2) dataset: was first described by Harries [21]. This data has collected the information from Australian New South Wales (NSW) Electricity Market. It contains 45312 electricity price records dated from May 1996 to December 1998, identifying the change of price in every 30 minutes with 7 variables: day of a week, timestamp, electricity demand/supply in NSW/ Victorian States and scheduled electricity transfer between states.

b. COVTYPE: is a forest cover type dataset for 30x30 meter cells which is obtained from US Forest Service Region 2 Resource Information System (RIS) data [22], containing 12 features to determine the type of forest cover with 7 classes.

4.2 Experiment 1. Performance of Preprocessing

In order to validate the availability of the proposed method, we first conduct the experiments to compare different methods, which shown below:

(1) Static: The model of the classifier is initialized with the reference data from the data stream and it will no longer be updated.

(2) Sliding: The model of the classifier is initialized with the reference data from the data stream and being updated in every window with the previously labeled data in the streaming phase.

(3) Statistical Test for Drift Detection Method (SDDM): In the streaming phase, the AD-test is employed to compare the difference between the reference window and the current window in the feature space. A drift alarm is triggered when the null hypothesis is rejected in the AD-test.

(4) Proposed Method: Clustered Statistical Test for Drift Detection Method (CSDDM): After the initial build, the classification model is updated when a drift is detected. The AD-test compares the difference between the reference window and the current window along the extracted feature spaces in each cluster.

We evaluated each of above methods on data streams from datasets using Linear Discriminant Analysis as the base classification model. Limiting the number of drifts close to the actual amount in synthetic datasets, the threshold of different approaches is tuned. In addition, the window size was set as $|w|=100$ for SEA and Hyperplane datasets; $|w|=336$ for the Elec2 dataset, which equals to records for a week; $|w|=1000$ for the CoverType dataset, and $a=0.05$ for both CSDDM and SDDM. We have measured the prediction accuracy of the classifier as the performance metric. By carrying out 10 runs of the experiments, the average result is shown in Table 1.

Table 1. Performance of methods on datasets

Dataset		Static	SCARGC	SDDM	CSDDM
SEA	Accuracy	0.8604	0.8407	0.8604	0.8712
	#Drifts	0	102	105	102
	Label Used (%)	0.10	10.30	10.60	10.30
Hyperplane $\{k=9, t=1.0\}$	Accuracy	0.7153	0.8477	0.8358	0.8429
	#Drifts	0	12	10	12
	Label Used (%)	0.01	0.1267	0.11	0.1320
Elec2	Accuracy	0.6999	0.7265	0.6992	0.7091
	#Drifts	0	42	41	41
	Label Used (%)	1.22	52.03	51.23	51.62
CoverType	Accuracy	0.5856	0.7903	0.8141	0.8147
	#Drifts	0	357	348	326
	Label Used (%)	0.34	60.24	60.24	56.23

The result shows our proposed method (CSDDM) has better performance than the statistical test (SDDM) with the same amount of drifts. Hence, CSDDM has more accurate concept drift detection results than SDDM, where most of the model updates are used to adapt the concept drift rather than the noise. However, SCARGC performs better than CSDDM in Hyperplane and Elec2 datasets.

4.3 Experiment 2. Performance of Proposed Method

The purpose of the second experiment was to compare our approaches to existing methods proposed in [13] for incremental evolving data in benchmark datasets. For all datasets, SDDM and CSDDM were using the following parameters: $|w|=300,\ a=0.05$. The average accuracy achieved by the methods over the stream is shown in Table 2. From these results, we observed that the classification accuracy of our proposed method is similar to SCARGC.

As the number of data points for each concept is known, we can calculate the number of expected drifts and measure both the false alarm rate (FPR) and the true positive rate (TPR). The false alarm rate is shown in Table 3 and the true positive rate is shown in Table 4. Figure 4 shows the ROC chart on the 4CRE-V1 dataset. The result shows that our approach performs better in detecting concept drifts than SCARGC and SDDM.

Table 2. Accuracy for methods on benchmark datasets

Dataset	Static	Sliding	SCARGC	SDDM	CSDDM
1CDT	98.57	99.92	99.87	99.86	99.92
2CDT	54.35	92.91	92.91	94.99	92.91
1CHT	93.40	99.43	99.31	99.38	99.43
2CHT	54.55	87.30	87.40	87.01	86.31
4CR	25.18	99.99	99.97	99.58	99.99
4CRE-V1	25.76	97.78	97.64	94.91	97.72
4CRE-V2	26.68	92.49	92.48	88.85	92.32
5CVT	44.19	86.99	86.99	87.08	86.93
1CSurr	65.01	91.91	91.91	91.79	91.89
4CE1CF	96.96	94.85	94.85	95.02	94.60
UG_2C_2D	46.76	96.00	96.00	95.82	96.03
MG_2C_2D	50.68	87.84	87.84	87.74	87.81
FG_2C_2D	64.98	84.38	82.17	84.30	84.43
UG_2C_3D	59.79	95.14	95.15	95.10	95.14
UG_2C_5D	67.93	93.10	93.10	93.02	93.10
GEARS_2C_2D	95.81	95.74	95.76	95.74	95.73
Overall Avg.	60.66	93.49	93.33	93.14	93.41

Table 3. False Positive Rate for methods on benchmark datasets

Dataset	SCARGC	SDDM	CSDDM
1CDT	0.14	0.71	0.93
2CDT	0.93	0.93	0.93
1CHT	0.29	0.71	1.00
2CHT	1.00	0.86	0.86
4CR	0.05	0.09	0.51
4CRE-V1	0.39	0.22	0.55
4CRE-V2	0.92	0.19	0.43
5CVT	1.00	0.48	0.66
1CSurr	0.99	0.34	0.62
4CE1CF	0.97	0.11	0.38
UG_2C_2D	0.82	0.17	0.27
MG_2C_2D	0.78	0.12	0.34
FG_2C_2D	0.51	0.12	0.34
UG_2C_3D	0.63	0.19	0.31
UG_2C_5D	0.91	0.27	0.43
GEARS_2C_2D	0.83	0.12	0.23
Overall Avg.	0.70	0.35	0.55

Table 4. True Positive Rate for Methods on Benchmark Datasets

Dataset	SCARGC	SDDM	CSDDM
1CDT	0.15	0.82	1.00
2CDT	1.00	0.97	1.00
1CHT	0.26	0.72	1.00
2CHT	1.00	0.92	0.82
4CR	0.09	0.07	0.50
4CRE-V1	0.63	0.65	0.99
4CRE-V2	0.94	0.19	0.62
5CVT	1.00	0.65	0.74
1CSurr	1.00	0.30	0.73
4CE1CF	0.97	0.14	0.42
UG_2C_2D	0.80	0.29	0.43
MG_2C_2D	0.81	0.12	0.32
FG_2C_2D	0.50	0.11	0.33
UG_2C_3D	0.74	0.29	0.48
UG_2C_5D	0.94	0.30	0.54
GEARS_2C_2D	0.84	0.11	0.22
Overall Avg.	0.73	0.42	0.63

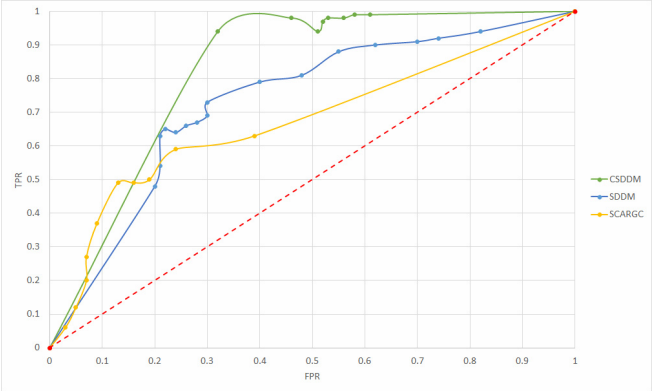

Figure 4. ROC of methods in 4CRE-V1 dataset

4.4 Experiment 3. Execution time of Proposed Method

Table 5 shows the average time in seconds required for datasets: UG-2C-2D, UG-2C-3D, MG-2C-2D, which are measured over 10 runs. We have conducted the measurement in a 1.2GHz workstation with 16GB RAM. It shows that our proposed method is faster than SCARGC due to the clustering for each window in SCARGC.

Table 5. Time Costs (in Minutes) Spent by the Algorithms

Dataset	SCARGC	CSDDM
UG-2C-2D	1.62	0.73
UG-2C-3D	3.23	1.49
MG-2C-2D	3.71	1.52

To demonstrate the execution time of the proposed method in the case of high dimensional feature spaces, we performed an experiment with the synthetic Hyperplane dataset by increasing the number of attributes from 10 to 200 in 20 steps. The number of drifting attributes maintains at 20% during the experiments and the magnitude of a drift is set as 1.0, with each dataset containing 100,000 of instances. The parameters of drift detection are set as |w|=990, a=0.05. The experiment was repeated 10 times and the average execution time is recorded and shown in Figure 5. Although the execution time of our proposed method is larger than SDDM due to the extra preprocessing of clustering and feature extraction, our proposed method is faster than SCARGC. The clustering for each window in SCARGC is time-consuming.

5 Conclusion

In this paper, we have presented an unsupervised method for concept drift detection in the data stream. The proposed method is based on the AD-test with clustering and feature extraction as preprocessing. The use of PCA for the feature extraction can reduce the detection time in the case of high dimensional feature

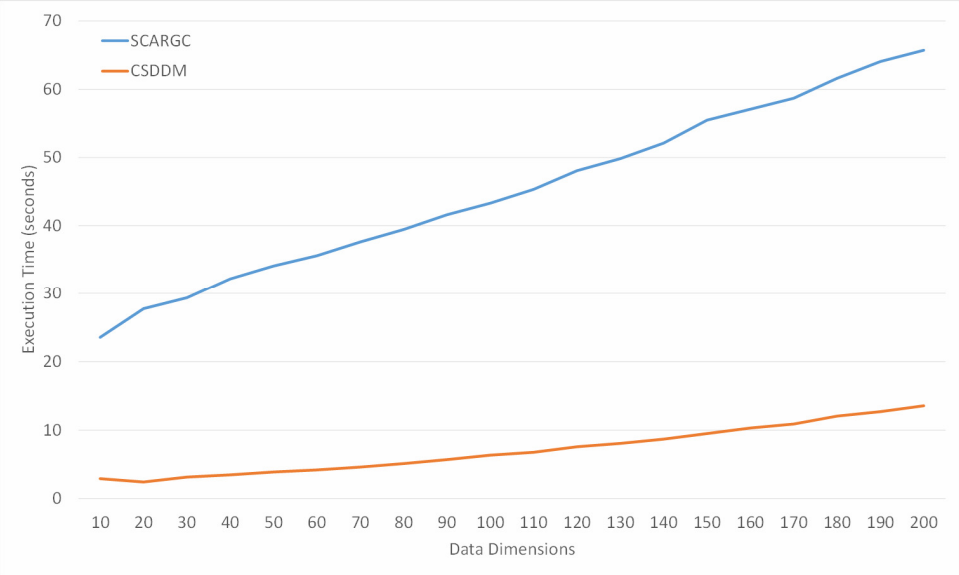

Figure 5. Execution time of methods in increasing dimension

spaces. Experiments using both synthetic data and real-world data show that the proposed method achieved a better classification accuracy as compared with SDDM, which is without unsupervised clustering as the preprocessing step.

As an unsupervised approach, the proposed detector (CSDDM) cannot detect the real concept drift as accurate as a supervised one. It detects both real concept drift and virtual concept drift, causing unnecessary model updates. However, our proposed method provides a drift detection method with lower false alarm rate and competitive classification accuracy as compared with other unsupervised approaches.

References

[1] A. Haque, L. Khan, M. Baron, Semi Supervised Adaptive Framework for Classifying Evolving Data Stream, *Advances in Knowledge Discovery and Data Mining, PAKDD*, Ho Chi Minh City, Vietnam, 2015, pp. 383-394.

[2] J. C. Schlimmer, R. H. Granger, Incremental learning from noisy data, *Machine Learning*, Vol. 1, No. 3, pp. 317-354, September, 1986.

[3] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, A. Bouchachia, A Survey on Concept Drift Adaptation, *ACM Computing Surveys*, Vol. 46, No. 4, pp. 44:41-44:37, April, 2014.

[4] G. Widmer, M. Kubat, Effective Learning in Dynamic Environments by Explicit Context Tracking, in *Proceedings of the European Conference on Machine Learning, ser. ECML '93*, Vienna, Austria, 1993, pp. 227-243.

[5] A. Bifet, R. Gavaldà, Learning from Time-Changing Data with Adaptive Windowing, *Proceedings of the 2007 SIAM International Conference on Data Mining*, 2007, pp. 443-448. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/1.9781611972771.42.

[6] P. Domingos, G. Hulten, Mining high-speed data streams, in *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD '00*, Boston, Massachusetts, USA, 2000, pp. 71-80. [Online]. Available: http://doi.acm.org/10.1145/347090.347107.

[7] J. Gama, P. Medas, G. Castillo, P. Rodrigues, Learning with Drift Detection, *Advances in Artificial Intelligence – SBIA 2004*, 2004, Sao Luis, Maranhao, Brazil, pp. 286-295. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-28645-5_29.

[8] M. Baena-Garcia, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, R. Morales-Bueno, Early drift detection method, in *Fourth International Workshop on Knowledge Discovery from Data Streams*, 2006, pp. 77-86.

[9] K. Nishida, K. Yamauchi, Detecting Concept Drift Using Statistical Testing, *International Conference on Discovery Science (DS 2007)*, 2007, Sendai, Japan, pp. 264-269. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-75488-6_27.

[10] D. M. dos Reis, P. Flach, S. Matwin, G. Batista, Fast Unsupervised Online Drift Detection Using Incremental Kolmogorov-Smirnov Test, *in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD '16*, San Francisco, CA, USA, 2016, pp. 1545-1554. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939836.

[11] D. V. Hinkley, Inference About the Change-Point in a Sequence of Random Variables, *Biometrika*, Vol. 57, No. 1, pp. 1-17, April, 1970.

[12] H. Mouss, D. Mouss, N. Mouss, L. Sefouhi, Test of Page-Hinckley, an approach for fault detection in an agro-alimentary production system, in *2004 5th Asian Control Conference*, Vol. 2, Melbourne, Victoria, Australia, 2004, pp. 815-818.

[13] V. M. A. Souza, D. F. Silva, J. Gama, G. E. A. P. A. Batista, Data Stream Classification Guided by Clustering on Nonstationary Environments and Extreme Verification Latency, *SIAM International Conference on Data Mining*, Britsh Columbia, Canada, 2015, pp. 873-881.

[14] Y. Sakamoto, K. I. Fukui, J. Gama, D. Nicklas, K. Moriyama, M. Numao, Concept Drift Detection with Clustering via Statistical Change Detection Methods, *Seventh International Conference on Knowledge and Systems Engineering (KSE)*, Ho Chi Minh City, Vietnam, 2015, pp. 37-42.

[15] L. I. Kuncheva, W. J. Faithfull, PCA Feature Extraction for Change Detection in Multidimensional Unlabeled Data, *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 25, No. 1, pp. 69-80, January, 2014.

[16] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, SMOTE: Synthetic minority over-sampling technique, *Journal of Artificial Intelligence Research*, Vol. 16, No. 1, pp. 321-357, January, 2002.

[17] I. Tomek, Two modifications of CNN, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 6, No. 11, pp. 769-772, November, 1976.

[18] G. E. A. P. A. Batista, A. L. C. Bazzan, M. C. Monard, Balancing Training Data for Automated Annotation of Keywords: a Case Study, *Workshop on Bioinformatics*, Macaé, RJ, Brazil, 2003, pp. 10-18.

[19] W. N. Street, Y. Kim, A streaming ensemble algorithm (SEA) for large-scale classification, *Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, California, USA, 2001, pp. 377-382.

[20] W. Fan, Systematic data selection to mine concept-drifting data streams, *The Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Seattle, WA, USA, 2004, pp. 128-137.

[21] M. Harries, *SPLICE-2 Comparative Evaluation: Electricity Pricing*, The University of New South Wales, Technical Report UNSW-CSE-TR-9905, July, 1999.

[22] J. A. Blackard, UCI Machine Learning Repository: Covertype Data Set, 1998.

Biographies

Jones Sai-Wang Wan received degree of Master of Science in Electrical Engineering (Computer Science group), National Taiwan University, 2017. He was a software engineer at Tread-Mirco, Taiwan and is now a software engineer with En-trak, Hong-Kong, focusing on data streaming and machine learning.

Sheng-De Wang received the B.S. degree from National Tsing Hua University, Hsinchu, Taiwan, in 1980, and the M. S. and the Ph. D. degrees in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1982 and 1986, respectively. Since 1986 he has been on the faculty of the department of electrical engineering at National Taiwan University, Taipei, Taiwan, where he is currently a professor. From 1995 to 2001, he also served as the director of computer operating group of computer and information network center, National Taiwan University. He was a visiting scholar in Department of Electrical Engineering, University of Washington, Seattle during the academic year of 1998-1999. From 2001 to 2003, He has been served as the Department Chair of Department of Electrical Engineering, National Chi Nan University, Puli, Taiwan. His research interests include embedded systems, internet computing and security, and intelligent systems.