LETTER

Concept Drift Detection for Evolving Stream Data

Jeonghoon LEE[†*a)], ***Member and*** **Yoon-Joon LEE**[††b)], ***Nonmember***

SUMMARY In processing stream data, time is one of the most significant facts not only because the size of data is dramatically increased but because the characteristics of data is varying over time. To learn stream data evolving over time effectively, it is required to detect the drift of concept. We present a window adaptation function on domain value (WAV) to determine the size of windowed batch for learning algorithms of stream data and a method to detect the change of data characteristics with a criterion function utilizing correlation. When applying our adaptation function to a clustering task on a multi-stream data model, the result of learning synopsis of windowed batch determined by it shows its effectiveness. Our criterion function with correlation information of value distribution over time can be the reasonable threshold to detect the change between windowed batches.

key words: *stream, stream data, concept drift, change of the characteristics, clustering*

1. Introduction

Progress in various hardware and sensor technology has created new kinds of data management. These data, being generated and growing continuously and rapidly over time, are referred to stream. Stream data has become a challenge to Knowledge Discovery and Data mining (KDD) due to their large size and dynamics in generation. Even various problems in managing and processing of stream data issue from high-dimensional attributes and multi-valued categorical values found in recent stream data.

Many tasks in KDD related stream data have been focused on relatively simple processes like searching and information filtering. In various domains, however, there are needs for more sophisticated tasks like summarizing and clustering to find hidden knowledge of data. Moreover, it is the advent of personal mobile hardware like smart phones and advanced application areas which generate continuous and pervasive data like social networks, that have changed most data considered static into types of stream [1], [2].

In processing stream data, time is one of the most significant facts not only because the size of data is dramatically increased but because the characteristics of data is varying over time. Generally stream data sets are too large to fit in main memory so and linear scan could be a cost-effective access method. Some common stream data like sensor network data and internet packet statistics are transient and do not need to be accumulated on memory. These data should occasionally be processed at a time when being generated and then be discarded for memory space whenever possible. Therefore processes for such stream data should be designed to use compressed data like synopsis or summary. Stream data are seriously affected by time because they emerge in time line and the characteristics of them are subject to be changed. Tasks of stream data require continuous or periodic update the current information to guarantee so that they reflect the latest status of data.

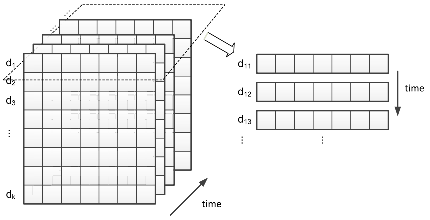

Fig. 1 A multi-stream data model.

First, we propose an adaptation function to determine the size of windowed batches of stream data. We assume that an item in a data set is not only a singleton of data but a series of stream and the whole data is a set of multi-series of streams in a multi-stream data model in Fig. 1. By an example, each stream data generated by an individual in a social network service like micro-blog could be an item in our data model. For effective synopsis reflecting concept of stream with limited subset of data, our adaptation function is used to build up the windowed batches of a series of stream. Second, we present a method for detecting the change of concepts between windowed batches of stream. Our approaches provide reasonable criteria to establish windowed batches of stream for efficient processing based on a clear statistical motivation and make it possible to detect the change of characteristics on linear scan effectively.

The rest of this paper is organized as follows. In Sect. 2 we present background information and our motivation on stream data and detecting concept drift on it and we describe how to model stream data and detect the change of intrinsic characteristics of stream in Sect. 3. In Sect. 4, we evaluate the effectiveness of our approaches by applying them to a clustering task on a synthesized data set. Section 5 con-

Manuscript received December 8, 2010.
Manuscript revised June 14, 2011.

[†] The author is with the Applied Mathematics and Systems Lab. (HPC Team), Ecole Centrale Paris, 92295 Chatenay Malabry, France.

[††] The author is with the Department of Computer Science, Korea Advanced Institute of Science and Technology, 291 Daehak-ro, Yuseong-gu, Daejeon 305–701, Republic of Korea.

[*] Corresponding author.

a) E-mail: jhoon@ymail.com, jhoon@kaist.edu
b) E-mail: yoonjoon.lee@kaist.ac.kr

DOI: 10.1587/transinf.E94.D.2288

cludes the paper by presenting our further remarks.

2. The Change of Data Characteristics

The change of intrinsic characteristics in stream data is phenomenon as known as the concept drift that the distribution of data collected over an extended period is likely to change over time. For example, many companies collect an increasing amount of data like customer profiles and sales figures to find out hidden information and patterns in the customer's behavior. The successful models of data should be adapted according to the customer's behavior, which tends to change.

In the previous problems of concept drift, stream data is represented by a series of batches labeled as -1 and 1 when arriving over time. The goal is to predict the label of the next batch with batches which arrived so far. It is assumed that any subset of the training items from previous batches (1 to t) can be used to predict the label of a batch $t + 1$. The handling of concept drift is often regarded just as the problem of fixed or variable size of time windowed subset on training data. Fast adaptability with a small window is conflict with good generalization with a large window. Some other researches adopt weights according to their age or utility for the classification task.

Unsupervised learning tasks like clustering of stream data have been related to extended works with legacy studies of large scale data and time driven data. Most of them, however, focus on development of incremental algorithm for data generated continuously [3], [4]. In practice, the accurate prediction of the changing concept is infeasible, if no restrictions are imposed on the type of admissible change because a function randomly jumping between the values one and zero cannot be predicted by any learner with more than 50% accuracy [5]. From this point of view, our research focuses on modeling and detecting the change of concepts on multi stream data sets for learning tasks.

3. Our Approach on Evolving Stream Data

In this paper, we study a multi-stream data model and the problem of concept drift detection on it for learning tasks. We start with defining attribute domain D, which is the set of distinct values on a given attribute.

Definition 1: Let $\mathcal{D} = \{D_1, D_2, \ldots, D_k\}$ be a set of bounded domains and $\mathcal{S} = D_1 \times D_2 \times \cdots \times D_k$ an k-dimensional non-numerical space. We refer to $D_1, \ldots, D_k$ as the attributes of $\mathcal{S}$. An attribute domain, D_i, is a set of all distinct values occurring in the attribute i and is defined as follows.

$$D_i = \{d_{i1}, d_{i2}, \ldots, d_{i|D_i|}\} \quad . \tag{1}$$

We consider all values as distinct discrete value to get a discrete value distribution vector for each attribute[†]. Now we define a stream data as follows.

Definition 2: Let a feature vector of each example of a stream data item be $\vec{x} = (x_1, x_2, \cdots, x_k)$, which consists of values as $x_i \in D_i$. A stream data X is a series of examples on batches in windows over time.

$$\cdots, \vec{x}_{(1,t)}, \cdots, \vec{x}_{(f(t),t)}, \vec{x}_{(1,t+1)}, \cdots, \vec{x}_{(f(t+1),t+1)}, \cdots \quad . \tag{2}$$

$\vec{x}_{(i,j)}$ denotes the i-th example of batch j. The size of batch is decided by an adjustment function $f(t)$. For each batch j, the data is distributed with respect to $Ds_j(\vec{x})$. The multi-stream data set S consists of a set of N stream items $S = \{X_1, X_2, \cdots, X_N\}$. According to a kind of changing of concept, the example distribution of $Ds_j(\vec{x})$ and $Ds_{j+1}(\vec{x})$ between windowed batches will show some difference.

The task of clustering on S is to find out groups among N stream items. Unlike other clustering methods on streams which determine group for a new arriving data, our task is to decide how to adjust the changed data and reconstruct groups reflecting the shift of characteristics. The goal of task is to decide the size of the windowed batch to contain sufficient information to generalize the distribution of the example value and detect a significant change of correlation between batches. In practice, multi-stream data are too large to remember the whole data scanned in the past. This scarcity of space necessitates the design of method to utilize information of neighboring batches.

3.1 Window Adaptation Based on Value Statistics

In our stream model, the size of windowed batch is not stationary and adaptive by an adaptation function. A batch of example represents a unit of atomic concept. It should be small enough to get swift adaptability for change and big enough to guarantee sufficient generality in order to avoid unnecessary learning process. Now we should answer to the question of "How many examples should be in a windowed batch?" We solve the problem of this compromising necessity by using a statistical result of the Hoeffding bound(also known as additive Chernoff bound) [6].

Consider a random variable v whose range is R. According to Hoeffding bound, the minimum number of observation n, which assures that their true mean value $\bar{v}$ is within ε of $\bar{v}$ by the probability of $1 - \delta$, is computed by the following equation.

$$\mathit{Hoeffding\_bound}(n) = \frac{1}{2}\left(\frac{R}{\varepsilon}\right)^2 \log\left(\frac{2}{\delta}\right) \tag{3}$$

By example, when the size of an attribute domain is 10, $\varepsilon = 3$, $\delta = 0.1$, the value of $\mathit{Hoeffding\_bound}(n)$ is 16.64, that is, we only need to gather 17 observations to determine the value of the attribute to within 3 of its true value with the probability of 90% without the further observation unless there is a significant change of the nature of the attribute.

In our model, the size of windowed batch is determined

[†] Real values can be transformed into discrete ones by histograms or transforming tools such as the Discrete Fourier Transform.

following a window adaptation function based on domain value (WAV) with parameters ε and δ.

$$WAV(t) = \max\{H(D_{1t}), H(D_{2t}), \cdots, H(D_{kt})\} \tag{4}$$

$H(D_{it})$ is based on Hoeffding bound for i-th attribute of an example in t-th windowed batch. For each windowed batch, the information of the attribute domain value of the first observed example determines the size of batch for the rest. The attribute domain set is reassessed every windowed batch and modified for most recent values over time. Our WAV function is adapted by change of the whole data characteristics not by change of a single example value.

3.2 Detecting the Change by Value Distribution

Now, we detect the change of data characteristics using correlation of data value distributions. If there is the change between two batches of examples, the value distribution of one batch does not correlate with that of the other.

Let a series of examples on a windowed batch t be $\vec{x}_{(1,t)}, \cdots, \vec{x}_{(WAV(t),t)}$ with each $\vec{x} = (x_1, x_2, \cdots, x_k)$ then the value distribution of the i-th attribute on the batch t is

$$\vec{a}_{i,t} = (a_{(i,t)1}, a_{(i,t)2}, \cdots, a_{(i,t)WAV(t)}) \tag{5}$$

The value distribution of an example $\vec{x}_p$ on a windowed batch t is

$$A_{p,t} = \begin{pmatrix} a_{(1,t)1} & \cdots & a_{(1,t)WAV(t)} \\ \vdots & \ddots & \vdots \\ a_{(k,t)1} & \cdots & a_{(k,t)WAV(t)} \end{pmatrix} \tag{6}$$

For detecting the change of characteristics of an example $\vec{x}_p$, we exploit correlation information between $A_{p,t}$ and $A_{p,t+1}$. $A_{p,t+1}$ correlates with $A_{p,t}$ in a significant literature if the characteristics of data remains unchanged and it does not otherwise. In our research, we use Pearson Correlation Coefficient to discover relationship between two windowed batches.

For two vectors v_1, v_2, the correlation between them is computed by the following equation.

$$corr(v_1, v_2) = \frac{\sum v_1 v_2 - \frac{\sum v_1 \sum v_2}{n}}{\sqrt{\sum {v_1}^2 - \frac{(\sum v_1)^2}{n}}\sqrt{\sum {v_2}^2 - \frac{(\sum v_2)^2}{n}}} \tag{7}$$

The criterion to decide effective correlation model is the distributional statistics of variables. Pearson Correlation Coefficient can discover a type of linear correlation for normal distribution effectively. To lessen effect of values following non-normal distribution, however, Rank Correlation Coefficients like Spearman's and Kendall's are more appropriate. Moreover, degree of freedom of multivariate t-distribution is more effective. The unit variable in our data model is for the value of a single attribute varying over windowed batches and is assumed to following to univariate normal distribution.

Our object function for detecting the change of characteristics in value distribution between windowed batches is defined as follows.

Definition 3: Let the value distributions of two windowed batches of with respect to an example p be $A_{p,t}$ and $A_{p,t+1}$ respectively. Both matrices are a form of $k \times m$ where $m = \min(WAV(t), WAV(t+1))$. When the batch size is bigger than m, the matrix is constructed by abstraction data selected randomly. The criterion function to detect the change (DC) between t and $t+1$ for the example p is defined as follows.

$$DC(p_t, p_{t+1}) = \left(\sum_{i=1}^{k} \frac{1 - corr(v_{(p,t)i}, v_{(p,t+1)i})}{2} \right) / k \ . \tag{8}$$

where $v_{(p,t)i}$ is a row vector of $A_{p,t}$ as $v_{(p,t)i} = [a_{p,q}]_{p=i,q=1,\cdots,m}$.

DC function ranges from 1 to 0, the higher value means the more change there is. The threshold value for judging whether the characteristics of data have been changed could be user's parameter depending on the kind of data set and tasks. The threshold is about 0.4 in general and this value is also shown in our experiment.

In our data model, the change of the multi stream data set S is determined by following.

$$DC(S_t, S_{t+1}) = \left(\sum_{p=1}^{N} \lambda(p) DC(p_t, p_{t+1}) \right) / N \ . \tag{9}$$

where $\lambda(p)$ is the weight of each item in data set. The default value is 1 when every item has equal weight.

On the multi-stream data model, we apply our approaches to an unsupervised learning framework (ULEVO) for evolving stream data as shown in Algorithm 1.

Algorithm 1 Procedure of ULEVO

```
Input:
$S$ is a multi-stream data set
$C$ is a learning task
1. For first example of stream data items $S_{(1,t)} = \{X_{(1,t)}, X_{(2,t)}, \cdots, X_{(N,t)}\}$,
   Decide the size of a windowed batch $t$ by calulate $WAV(t)$.
2. Conduct $C$ on a batch of stream data set $S_t$.
3. For a new example after the batch $t$,
   Decide the size of a windowed batch $t+1$ by calulate $WAV(t+1)$.
4. Calculate $DC(S_t, S_{t+1})$.
   If $DC(S_t, S_{t+1}) > \varphi$ then Conduct C on a batch of stream data set $S_{t+1}$.
5. Set t = t+1;
6. Repeat 3 - 6.
```

In our learning framework, one windowed batch of stream data is object to learning process and all data of in each batch are summarized into a single synopsis. For our example task, clustering, the mean of data values can be simple and effective representative.

4. Experimental Evaluation

The purposes of our research are to determine the appropriate size of windowed batches to reduce load of repeating an expensive learning task and to detect the change of characteristics for diminishing a defect in reflecting change of evolving data effectively. We evaluate the soundness of our

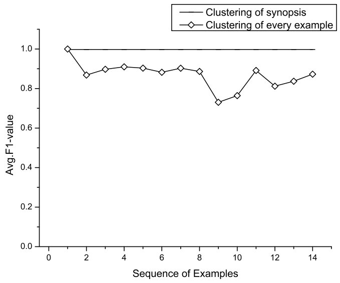

Fig. 2 The result of the clustering with synopsis and the clustering with each example.

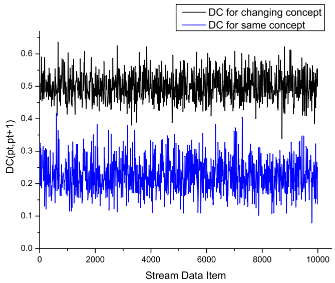

Fig. 3 DC of stream data in the same concept and in the different concepts.

criterion to detect the change and the efficiency of learning by the synopsis of windowed batches by showing that its result is within tolerable error comparing with the result of learning by every single item in stream data sets. For our experiment, we use a clustering method suggested in [7] as a learning task, which is based on k-modes with an effective dissimilarity measure utilizing intrinsic levels of dissimilarity of domain values.

We generated a synthetic multi-stream data set with a series of 10,000 items of 10-dimensional space $U = D_1 \times D_2 \times \cdots \times D_{10}$ in each example. Each attribute of data items has one of 9 relative values(i.e., $|D_i| = 9$ for $1 \leq i \leq 10$) and the whole data items in the first example have been grouped into 3 clusters. In the first arriving example, the attributes $D_1, D_2, \ldots, D_5$ have randomly selected values which are irrelevant to groups in order to guarantee the effect of relative importance of attributes for clustering task and the other attributes $D_6, D_7, \ldots, D_{10}$ have relevant values selected from designated value sets with size of 3 for each cluster to preserve relationship among the values depending on groups. In practice, the value of attribute $i\,(6 \leq i \leq 10)$ is selected in the subset of each attribute domain: $\{d_{i1}, d_{i2}, d_{i3}\}, \{d_{i4}, d_{i5}, d_{i6}\}, \{d_{i7}, d_{i8}, d_{i9}\}$ for Cluster *A, B, C* respectively. In the same context, all items in every example varied within error bound $\varepsilon = |D_i|/3$. After the change occurs, the attributes which have relevant values from the designated value set become are $D_1, D_2, \ldots, D_5$ and the other attributes $D_6, D_7, \ldots, D_{10}$ have irrelevant values. To show performance of clustering, we use $F_1 - value$, the harmonic mean of precision and recall.

In Fig. 2, the result of clustering with a synopsis is not worse than that with each example in a windowed batch, whose size is 14 determined by WAV. Several trials for the different batches show the similar results. For a multi-stream data model, the clustering process should be repeated whenever a new example arrives to keep information up-to-date. However, our window adaption function (WAV) determines the size of sequence which is enough to establish a sound synopsis as far as the concept of data is not changed (values of data varies only within the error bound ε). The data tend to fluctuate dynamically over time and the noise factor could be generated in the process. Our WAV is based on value distribution of data set and can decide the effective size of a batch in the conceptual point of view. Thus the synopsis of the batch could have an effect on eliminating this unnecessary noise factor. For continuously generated data over time, it can be said that the learning by synopsis of individual data is sufficiently effective because it is not just proper for efficient process but could result in the better performance depending the kinds of tasks and data.

Figure 3 compares the criteria values for detecting the change of concept (DC) between batches with the same concept (within the errorbound ε) with those with the changed concept in stream data set. In each circumstance, DC values of the 10000 series of stream data items are shown. When the change occurs, our DC ranges over 0.4. DC for the same concept, however, does under 0.4 on average. The result shows our DC measure is a reasonable criterion of the change for DC based on Pearson Correlation Coefficient ranges from 0 to 1 where 0 means that there is deep correlation between observations and 1 means the opposite.

5. Concluding Remarks

In processing stream data, learning tasks should be adapted to the change of concept to reflect the up-to-date information. We present a window adaptation function (WAV) to determine the appropriate size of windowed batches and a method to detect the change of data characteristics with a criterion function (DC). WAV function utilizes the distribution of domain values and can determine the size of windowed batches which is effective to establish a sound synopsis representing values in them. When applying to a learning task, clustering, it showed that the learning by a synopsis with WAV is efficient and effective. The correlation of data values distribution could be good information to detect the change of them and DC could provide a reasonable threshold to detect the change between windowed batches in stream data set.

For our window adaptation function, there are some user parameters such as an error bound and probability values. These values play important roles in deciding the effective size of batches and the further research of theoretical basis for them is required. Our research focused only on preprocessing part of the whole learning framework. Therefore, practical learning methods to utilize our approaches are needed and we are planning to develop a kind of incremental clustering algorithm for these.

References

[1] M. Azimi, P. Nasiopoulos, and R.K. Ward, "Data transmission schemes for DVD-like interactive TV," IEEE Trans. Multimedia, vol.8, no.4, pp.856–865, 2006.

[2] L.G.P. Alves, F.S.D. Silva, and G. Bressan, "Collabora TV ware," Int. J. Advanced Media and Communication, vol.3, pp.365–382, 2009.

[3] Guha, Meyerson, Mishra, Motwani, and O'Callaghan, "Clustering data streams: Theory and practice," IEEETKDE, vol.15, 2003.

[4] C.C. Aggarwal, J. Han, J. Wang, and P.S. Yu, "A framework for clustering evolving data streams," Proc. 29th VLDB, pp.81–92, 2003.

[5] R. Klinkenberg and T. Joachims, "Detecting concept drift with support vector machines," Proc. 17th Int. Conf. on Machine Learning, pp.487–494, 2000.

[6] W. Hoeffding, "Probability inequalities for sums of bounded random variables," J. American Statistical Association, vol.58, no.301, pp.13–30, 1963.

[7] J. Lee, Y.J. Lee, and M. Park, "Clustering with domain value dissimilarity for categorical data," Advances in Data Mining, pp.310–324, 2009.