An EM-Based Algorithm for Clustering Data Streams in Sliding Windows

Conference Paper · April 2009

DOI: 10.1007/978-3-642-00887-0_18 · Source: DBLP

CITATIONS
17

READS
57

5 authors, including:

Xuan Hong Dang
Nanyang Technological University
15 PUBLICATIONS **219** CITATIONS

SEE PROFILE

Vincent C. S. Lee
Monash University (Australia)
108 PUBLICATIONS **1,513** CITATIONS

SEE PROFILE

Wee Keong Ng
Nanyang Technological University
435 PUBLICATIONS **5,109** CITATIONS

SEE PROFILE

Kok-leong Ong
Deakin University
46 PUBLICATIONS **252** CITATIONS

SEE PROFILE

An EM-Based Algorithm for Clustering Data Streams in Sliding Windows

Xuan Hong Dang[1], Vincent Lee[1], Wee Keong Ng[2], Arridhana Ciptadi[2], and Kok Leong Ong[3]

[1] Monash University, Australia
{xhdang;vincent.lee}@infotech.monash.edu
[2] Nanyang Technological University, Singapore
{awkng,arri0001}@ntu.edu.sg
[3] Deakin University, Australia
leong@deakin.edu.au

Abstract. Cluster analysis has played a key role in data understanding. When such an important data mining task is extended to the context of data streams, it becomes more challenging since the data arrive at a mining system in one-pass manner. The problem is even more difficult when the clustering task is considered in a sliding window model which requiring the elimination of outdated data must be dealt with properly. We propose SWEM algorithm that exploits the Expectation Maximization technique to address these challenges. SWEM is not only able to process the stream in an incremental manner, but also capable to adapt to changes happened in the underlying stream distribution.

1 Introduction

In recent years, we are seeing a new class of applications that changed the traditional view of databases as a static store of information. These applications are commonly characterized by the unbounded data streams they generate (or receive), and the need to analyze them in a continuous manner over limited computing resources [2,4]. This makes it imperative to design algorithms that compute the answer in a continuous fashion with only one scan of the data stream, whilst operating under the resource limitations. Among various data mining tasks, clustering is one of the most important tasks. Research in data stream clustering reported so far has mostly focused on two mining models, the landmark window [1,5] and the forgetful window [3,4]. While these two mining models are useful in some data stream applications, there is a strong demand to devise novel techniques that are able to cluster the data streams in a sliding window model which is the most general and also the most challenging mining model since it considers the elimination of outdated data.

We propose in this paper algorithm SWEM (clustering data streams in a time-based Sliding Window with Expectation Maximization technique) to address the above challenges. SWEM consists of two stages which are designed to strictly address the problem of constrained memory usage and one-pass processing over

X. Zhou et al. (Eds.): DASFAA 2009, LNCS 5463, pp. 230–235, 2009.

data streams. Furthermore, we develop in SWEM two important operations, namely splitting and merging micro components, in order to automatically adapt to changes happened frequently in stream distributions. Various experimental results confirm the feasibility of our proposed algorithm.

2 Problem Formulation

We focus on the time-based sliding window model. Let $TS_0, TS_1, \ldots, TS_i$ denote the time periods elapsed so far in the stream. Each time period contains multiple data points $x_i = \{x_i^1, x_i^2, ..., x_i^d\}$ (in d-dimensional space) arriving in that interval. Given an integer b, a time-based sliding window $\mathcal{SW}$ is defined as the set of records arriving in the last b time periods $\mathcal{SW} = \{TS_{i-b+1}, \ldots, TS_{i-1}, TS_i\}$. TS_i is called the latest time slot and TS_{i-b} is the expiring one. We also assume that the stream evolves with time and data points are generated as a result of a dynamic statistical process which consists of k mixture models. Each model corresponds to a cluster that follows a multivariate normal distribution. Consequently, any cluster $C_h, 1 \leq h \leq k$, is characterized by a parameter: $\phi_h = \{\alpha_h, \mu_h, \Sigma_h\}$ where α_h is the cluster weight, μ_h is its vector mean and Σ_h is its covariance matrix. Accordingly, our clustering problem is defined as the process of identifying parameters $\Phi_G = \{\phi_1, ..., \phi_k\}$ that optimally fit the current set of data points arriving in the last b time periods in the stream.

3 Algorithm Description

Initial Phase: We compute m distributions (also called micro components) modelling the data within each time slot of the sliding window. Let $\Phi_L = \{\phi_1, ..., \phi_m\}$ be the set of parameters of these local components where each MC_ℓ, $1 \leq \ell \leq m$, is assumed to follow a Gaussian distribution characterized by $\phi_\ell = \{\alpha_\ell, \mu_\ell, \Sigma_\ell\}$. For the initial phase where $SW = \{TS_0\}$, the initial values for these parameters will be randomly chosen.

In our framework, each data point belongs to all components yet with different probabilities. Given x, its probability (or weight) in a component ℓ^{th} is: $p(\phi_\ell|x) = \frac{\alpha_\ell \times p_\ell(x|\phi_\ell)}{p(x)} = \frac{\alpha_\ell \times p_\ell(x|\phi_\ell)}{\sum_{i=1}^{m} \alpha_i \times p_i(x|\phi_i)}$, in which $p_\ell(x|\phi_\ell) = (2\pi)^{-d/2}|\Sigma_\ell|^{-1/2}$ $\exp\left[-\frac{1}{2}(x-\mu_\ell)^T \Sigma_\ell^{-1}(x-\mu_\ell)\right]$. We also assume data points are generated independently and thus, the probability of n records in TS_0 is computed by the product: $p(TS_0|\Phi_L) = \prod_{x_i \in TS_0} p(x_i|\Phi_L) = \prod_{i=1}^{n} \sum_{\ell=1}^{m} \alpha_\ell \times p_\ell(x_i|\phi_\ell)$, and in log likelihood form $Q(\Phi_L) = |TS_0|^{-1} \log \prod_{x \in TS_0} \sum_{h=1}^{m} \alpha_\ell \times p_\ell(x_i|\phi_\ell)$ which defines the average log likelihood measure.

In the first stage, SWEM employs the EM technique to maximize $Q(\Phi_L)$. Once the algorithm converges, the set of micro components are approximated by keeping a triple $T_\ell = \{N_\ell = |S_\ell|, \theta_\ell = \Sigma_{x_i \in S_\ell} x_i, \Gamma_\ell = \Sigma_{x_i \in S_\ell} x_i x_i^T\}$ for each MC_ℓ (where S_ℓ is the set of data points assigned to MC_ℓ to which they have the highest probability). The important property of T_ℓ is that it is sufficient to compute the mean and covariance of MC_ℓ. Concretely, $\mu_\ell = N_\ell^{-1}\theta_\ell$ and

$\Sigma_\ell = N_\ell^{-1}\Gamma_\ell - N_\ell^{-2}\theta_\ell \times \theta_\ell^T$. Furthermore, its additive property guarantees the mean and covariance matrix of a merged component can be easily computed from the triples of each member component. With these sufficient statistics, SWEM computes the k global clusters $\phi_h \in \Phi_G$ in the second stage:

E-step: $p(\phi_h|T_\ell) = \frac{\alpha_h^{(t)} \times p_h(\frac{1}{N_\ell}\theta_\ell|\mu_h^{(t)}, \Sigma_h^{(t)})}{\sum_{i=1}^{k} \alpha_i^{(t)} \times p_i(\frac{1}{N_\ell}\theta_\ell|\mu_i^{(t)}, \Sigma_i^{(t)})}$

M-step: $\alpha_h^{(t+1)} = \frac{1}{n}\sum_{\ell=1}^{m} N_\ell \times p(\phi_h|T_\ell)$; $\mu_h^{(t+1)} = \frac{1}{n_h}\sum_{\ell=1}^{m} p(\phi_h|T_\ell) \times \theta_\ell$;

$\Sigma_h^{(t+1)} = \frac{1}{n_h}\left[\sum_{\ell=1}^{m} p(\phi_h|T_\ell)\Gamma_\ell - \frac{1}{n_h}\sum_{\ell=1}^{m}\left(p(\phi_h|T_\ell)\theta_\ell\right)\left(p(\phi_h|T_\ell)\theta_\ell\right)^T\right]$

where $n_h = \sum_{\ell=1}^{m} N_\ell \times p(\phi_h|T_\ell)$.

Incremental Phase: In this phase SWEM utilizes the converged parameters in the previous time slot as the initial values for the mixture models' parameters. This helps minimize the number of iterations if the stream's characteristic does not vary much. However, in case the stream's distribution significantly changes, it is necessary to re-locate components. We develop splitting and merging operations in order to *discretely* re-distribute components in the entire data space.

An MC_ℓ is split if it is large enough and has the highest variance sum (i.e., its data are most spread). Assume dimension e having largest variance is chosen for splitting, parameters for two resulting components MC_{ℓ_1} and MC_{ℓ_2} are approximated by: $\mu_{\ell_1}^e = \int_{\mu_\ell^e - 3\sigma_\ell^e}^{\mu_\ell^e} x \times p_{\ell,e}(x|\phi_\ell)dx$; $\mu_{\ell_2}^e = \int_{\mu_\ell^e}^{\mu_\ell^e + 3\sigma_\ell^e} x \times p_{\ell,e}(x|\phi_\ell)dx$; $(\sigma_{\ell_1}^e)^2 = \int_{\mu_\ell^e - 3\sigma_\ell^e}^{\mu_\ell^e} x^2 \times p_{\ell,e}(x|\phi_\ell)dx - (\mu_{\ell_1}^e)^2$; $(\sigma_{\ell_2}^e)^2 = \int_{\mu_\ell^e}^{\mu_\ell^e + 3\sigma_\ell^e} x^2 \times p_{\ell,e}(x|\phi_\ell)dx - (\mu_{\ell_2}^e)^2$. For other dimensions, their means and variances are kept unchanged. On the other hand, two components are merged if they are small and close enough. The closeness is measured based on Mahalanobis distance: $Avg(D_{i,j}) = \frac{1}{2}[(\mu_i - \mu_j)^T \Sigma_j^{-1}(\mu_i - \mu_j) + (\mu_j - \mu_i)^T \Sigma_i^{-1}(\mu_j - \mu_i)]$. Parameters for the merging component is computed relied on the additive property: $\alpha_\ell = \alpha_{\ell_1} + \alpha_{\ell_2}$; $\mu_\ell = \frac{\alpha_{\ell_1}}{\alpha_{\ell_1}+\alpha_{\ell_2}} \times \mu_{\ell_1} + \frac{\alpha_{\ell_2}}{\alpha_{\ell_1}+\alpha_{\ell_2}} \times \mu_{\ell_2}$; $\Sigma_\ell = \frac{\Gamma_\ell}{n(\alpha_{\ell_1}+\alpha_{\ell_2})} - \frac{\theta_\ell \times \theta_\ell^T}{(n(\alpha_{\ell_1}+\alpha_{\ell_2}))^2}$. In that, $\theta_\ell = n \times (\alpha_{\ell_1} \times \mu_{\ell_1} + \alpha_{\ell_2} \times \mu_{\ell_2})$; $\Gamma_\ell = n[\alpha_{\ell_1}(\Sigma_{\ell_1} + \mu_{\ell_1}\mu_{\ell_1}^T) + \alpha_{\ell_2}(\Sigma_{\ell_2} + \mu_{\ell_2}\mu_{\ell_2}^T)]$.

Expiring Phase: This phase is applied when the window slides and the oldest time slot is expired from the mining model. $\Phi_G = \{\phi_1, \ldots, \phi_k\}$ is updated by subtracting the statistics summarized in $\Phi_L = \{\phi_1, \phi_2, ..., \phi_m\}$ of the expiring time slot. SWEM controls this process by using a fading factor λ $(0 < \lambda < 1)$ to gradually remove these statistics. At each iteration t, it reduces the weight of each expiring MC_ℓ by $N_\ell^{(t)} = \lambda^{(t)} N_\ell$. Equivalently, the reducing amount denoted by $r_\ell^{(t)}$ is $r_\ell^{(t)} = (1-\lambda)\lambda^{(t-1)} N_\ell$. The following theorem guarantees the number of iterations t can be any arbitrary integer while the total reducing weight on each expiring component approaches its original value.

Theorem 1. *Let t be an arbitrary number of iterations used by the SWEM algorithm. Then for each expiring micro component MC_ℓ:* $\lim_{t\to\infty} \sum_t r_\ell^{(t)} = N_\ell$

In specific, E-step updates: $p(\phi_h|T_\ell) = \frac{\alpha_h^{(t)} \times p_h(\frac{1}{N_\ell}\theta_\ell|\mu_h^{(t)}, \Sigma_h^{(t)})}{\sum_{i=1}^{k} \alpha_i^{(t)} \times p_i(\frac{1}{N_\ell}\theta_\ell|\mu_i^{(t)}, \Sigma_i^{(t)})}$

Table 1. Average log likelihood values returned by stdEM, SWEMw/oG and SWEM

	D2.K10.N100k			D4.K5.N100k			D10.K4.N100k		
TS	stdEM	w/oG	SWEM	stdEM	w/oG	SWEM	stdEM	w/oG	SWEM
2	-10.436	-10.512	-10.512	-19.252	-19.276	-19.276	-47.846	-47.869	-47.869
4	-10.427	-10.446	-10.446	-19.192	-19.215	-19.215	-47.933	-48.010	-48.010
6	-10.451	-10.604	-10.716	-19.164	-19.220	-19.326	-47.702	-47.712	-47.726
8	-10.444	-10.700	-10.735	-19.188	-19.226	-19.245	-47.859	-47.884	-47.886
10	-10.439	-10.523	-10.579	-19.202	-19.247	-19.258	-47.759	-47.820	-47.873

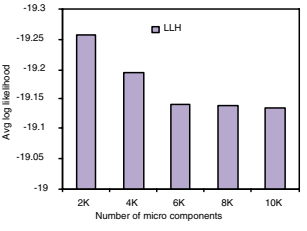

Fig. 1. Micro Components vs. Accuracy

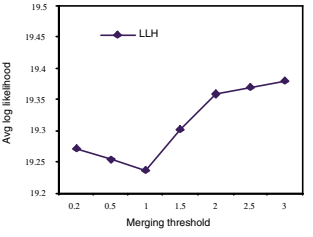

Fig. 2. Merging threshold $Avg(D_{i,j})$ vs. Accuracy

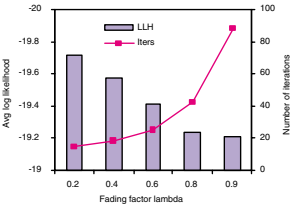

Fig. 3. Fading Factor λ vs. Accuracy

M-step: $n_G^{(t+1)} = n_G^{(t)} - \sum_{\ell=1}^{m} r_\ell^{(t+1)}$;$n_h^{(t+1)} = \alpha_h^{(t)} \times n_G^{(t)} - \sum_{\ell=1}^{m} p(\phi_h|T_\ell) \times r_\ell^{(t+1)}$; $\alpha_h^{(t+1)} = \frac{n_h^{(t+1)}}{n_G^{(t+1)}}$ where $\mu_h^{(t+1)} = \frac{\theta_h^{(t+1)}}{n_h^{(t+1)}}$; $\Sigma_h^{(t+1)} = \frac{1}{n_h^{(t+1)}} \left[\Gamma_h^{(t+1)} - \frac{1}{n_h^{(t+1)}} \theta_h^{(t+1)} \theta_h^{(t+1)^T} \right]$, where $\theta_h^{(t+1)} = \theta_h^{(t)} - \sum_{\ell=1}^{m} p(\phi_h|T_\ell) \times r_\ell^{(t+1)} \times \frac{\theta_\ell}{N_\ell}$.

4 Experimental Results

Our algorithms are implemented using Visual C++ and the experiments are conducted on a 1.9GHz Pentium IV PC with 1GB memory space running Windows XP platform.

Clustering Quality Evaluation: Using the notations and method described in [1], three datasets $D2.K10.N100k$, $D4.K5.N100k$ and $D10.K4.N100k$ are generated. Unless otherwise indicated, the following parameters are used: $b = 5$; $m = 6K$ (where K is the number of global clusters), $Avg(D_{i,j}) = 1$, $\lambda = 0.8$, and each $TS_i = 10k$ data points. Table 1 shows the average log likelihood results returned by our experiments. It is observed that the clustering results of SWEM are very close to those of stdEM (a standard EM algorithm working without any stream constraints). Analogically, SWEM's clustering quality is almost identical to that of SWEMw/oG (a algorithm that derives global models by simply re-clustering all sets of micro components). This result verifies the efficiency of our gradually reducing weight technique.

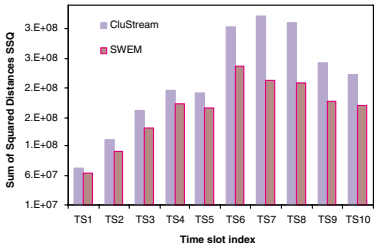

Fig. 4. Clustering quality comparison

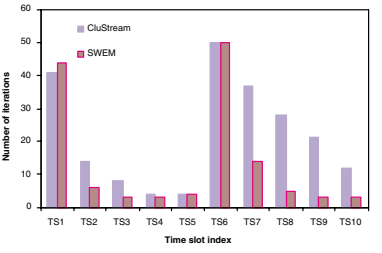

Fig. 5. Execution time comparison

Parameter Sensitivity: Using $D4.K5.N100k.AB$, a combination of two completely different stream distributions, we test the sensitivity of SWEM to various parameters. Figures 1, 2 and 3 report the accuracy of SWEM when the number of MC_ℓ, merging threshold $Avg(D_{i,j})$, and fading factor λ are respectively varied. We make the following observations. The average log likelihood becomes stable when $m = 6K$ (indicating that m need not be set too large, yet SWEM is still able to achieve high clustering quality); $Avg(D_{i,j})$ is set around 1 and λ is between 0.8 and 0.9.

Comparison with CluStream: It is observed from figures 4 and 5 that both the quality and execution time of SWEM are better than those of CluStream. This is understood by the difference in two algorithms' design. Whenever a significant change happens in the stream's distribution, SWEM re-distributes the set of micro components in the entire data space by using split and merge operations, CluStream, instead, simply creates a new cluster for a new point which cannot be absorbed by any clusters and merge other old ones. This causes the clusters' weights very imbalance and usually leads to a poor approximation of CluStream. Analogously, while SWEM minimizes the number of iterations by using the converged parameters of the previous time interval, CluStream always tries to update new data points into the set of micro clusters being maintained so far in the stream. This degrades the performance of Clustream and requires it more time to converge.

5 Conclusions

In this paper, we have addressed the problem of clustering data streams in one of the most challenging mining model, the time-based sliding window. We proposed SWEM algorithm that is able to compute clusters with a strictly single scan over the stream and work within confined memory space. Importantly, two techniques of splitting and merging components were developed to address the problem of time-varying data streams. SWEM has a solid mathematical background as it is designed based on the EM technique. Such a mathematically sound tool has been shown to be stable and effective in many domains despite the mixture models it employs being assumed to follow Gaussian distributions.

References

1. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: Aberer, K., Koubarakis, M., Kalogeraki, V. (eds.) VLDB 2003. LNCS, vol. 2944, pp. 81–92. Springer, Heidelberg (2004)
2. Babcock, B., Babu, S., Datar, M., Motwani, R., Widom, J.: Models and issues in data stream systems. In: PODS, pp. 1–16 (2002)
3. Cao, F., Ester, M., Qian, W., Zhou, A.: Density-based clustering over an evolving data stream with noise. In: Jonker, W., Petković, M. (eds.) SDM 2006. LNCS, vol. 4165. Springer, Heidelberg (2006)
4. Chen, Y., Tu, L.: Density-based clustering for real-time stream data. In: SIGKDD Conference, pp. 133–142 (2007)
5. Aoying, Z., Feng, C., Ying, Y., Chaofeng, S., Xiaofeng, H.: Distributed data stream clustering: A fast em-based approach. In: ICDE Conference, pp. 736–745 (2007)