Clustering Massive Text Data Streams by Semantic Smoothing Model

Yubao Liu[1], Jiarong Cai[1], Jian Yin[1], and Ada Wai-Chee Fu[2]

[1] Department of Computer Science of Sun Yat-Sen University, Guangzhou, 510275, China
liuyubao@mail.sysu.edu.cn, kelvin2004_cai@163.com,
issjyin@mail.sysu.edu.cn
[2] Department of Computer Science and Engineering, the Chinese University of Hong Kong,
Hong Kong
adafu@cse.cuhk.edu.hk

Abstract. Clustering text data streams is an important issue in data mining community and has a number of applications such as news group filtering, text crawling, document organization and topic detection and tracing etc. However, most methods are similarity-based approaches and use the TF*IDF scheme to represent the semantics of text data and often lead to poor clustering quality. In this paper, we firstly give an improved semantic smoothing model for text data stream environment. Then we use the improved semantic model to improve the clustering quality and present an online clustering algorithm for clustering massive text data streams. In our algorithm, a new cluster statistics structure, cluster profile, is presented in which the semantics of text data streams are captured. We also present the experimental results illustrating the effectiveness of our technique.

Keywords: Semantic Smoothing, Text Data Streams, Clustering.

1 Introduction

Clustering text data streams is an important issue in data mining community and has a number of applications such as news group filtering, text crawling, document organization and TDT (topic detection and tracing) etc. In such applications, text data comes as a continuous stream and this presents many challenges to traditional static text clustering [1].

The clustering problem has recently been studied in the context of numeric data streams [2, 3]. But, the text data streams clustering research is only on the underway stage. In [4], an online algorithm framework based on traditional numeric data streams clustering approach is presented for categorical and text data streams. In [4], the concept of cluster droplet is used to store the real-time condensed cluster statistics information. When a document comes, it would be assigned to the suitable cluster and then the corresponding cluster droplet is updated. This framework also distinguishes the historical documents with the presents by employing a fading function to describe the temporal locality attribute. The single pass clustering method for online event

R. Alhajj et al. (Eds.): ADMA 2007, LNAI 4632, pp. 389–400, 2007.

detection of text data streams is presented in [5] that is also an extension of traditional numeric data streams clustering. Different from [4], in [5], the time window is used to describe the changes of text data streams. Different from [4, 5], the idea of [6] uses the static spherical k-means text clustering method for online text data streams clustering and it only keeps cluster centers instead of cluster droplets. When a document comes, it would not be clustered at once. Instead, it is accumulated as a portion of a segment. When a segment (a fixed number of documents) forms, the online spherical k-means algorithm would be employed and then update the cluster centers. In [9], a feature-pivot clustering technique is presented for the detection of a set of bursty events from a text stream.

Recently, [7] argues that a text document is often full of class-independent "general" words (such as stop words that may shared by different classes of two documents) and short of class-specific "core" words (such as the related topic words occur in the document), which often leads to poor document clustering quality. In [7], model-based clustering approaches based on semantic smoothing that is widely used in information retrieval (IR) [8] is presented for efficient text data clustering. Actually, most existing clustering algorithms for text data streams are similarity-based approaches and often employ the heuristic TF*IDF scheme to discount the effect of "general" words. As shown in [7], semantic smoothing model is often better than TF*IDF scheme in improving the clustering quality. Inspired by semantic smoothing model, in this paper, we extend semantic smoothing model for text data streams environment and use the semantic smoothing model to improve the clustering quality. We also present an online clustering algorithm (short for *OCTS*) for clustering massive text data streams. In our algorithm, a new cluster statistics structure, cluster profile (short for *CP*), is presented in which the semantics of text data streams are captured. Different from the cluster droplet [4], cluster profile is build by the semantic smoothing model but the TF. We also present the experimental results illustrating the effectiveness of the technique.

2 The Basic Concepts for the Clustering of Text Data Streams

In text data streams environment, text document data comes as a continuous stream. In order to account for the evolution of the data stream, we assign a time-sensitive weight to each document data point. It is assumed that each data point has a time-dependent weight defined by the function $f(t)$. The function $f(t)$ is also referred to as the *fading function*. The fading function $f(t)$ is a non-monotonic decreasing function which decays uniformly with time t. In order to formalize this concept, we will define the *half-life* of a point in the data stream.

Definition 1 (Half life). The half life t_0 of a point is defined as the time at which $f(t_0) = (1/2) f(0)$.

The aim of defining a half life is to define the rate of decay of the weight associated with each data point in the stream. The decay-rate is defined as the inverse of the half life of the data stream. Similar to [4], we denote the decay rate by $\zeta = 1/\mathrm{t}_0$ and the fading function is defined as follow.

Definition 2 (Fading Function). Consider the time t, the fading function value is defined as $f(t) = 2^{-\zeta t}$, here $\zeta = 1/t_0$ and t_0 is the half life of the stream.

3 The Semantic Smoothing Model

Many previous approaches use word extraction method and single word vector as the document features. However, they suffer from the context-insensitivity problem. The terms in these models may have ambiguous meanings. In contrast, the semantic smoothing model uses the multiword phrases as topic signatures (document features). For example, the multiword phrase "fixed star" (denotes planet) has clearer meaning than the single word "star" (denotes either a celestial body or a pop star).

After phrase extraction, the training process determines the probability of translating the given multiword phrase to terms in the vocabulary. For example, if the word "planet" frequently appears in the documents whose topic contains "fixed star", then "fixed star" and "planet" must have some specific relationship. The translation model finds out such relationship and assigns a degree to describe it. In the following process (e.g. clustering), if we encounter a document contains the topic signature "fixed star" (but not "planet"), we can also assign a rational probability count to the word "planet" for the document.

For each phrase t_k, it would have a set of documents (D_k) containing that phrase. Since not all words in D_k center on the topic signature t_k, we assume D_k is generated by a mixture language model (i.e. all terms in the document set are either translated by the given topic signature model $p(w|t_k)$ or generated by the background collection model $p(w|C)$). The formulas (1) (2) (3) are used to iteratively compute the translation probabilities [7, 8].

$$p(w \mid D_k) = (1 - \beta)\, p(w \mid t_\mathrm{k}) + \beta p(w \mid C) \tag{1}$$

$$\hat{p}^{(n)}(w) = \frac{(1-\beta)^{(n)}\, p(w \mid t_\mathrm{k})}{(1-\beta)p^{(n)}(w \mid t_\mathrm{k}) + \beta p(w \mid C)} \tag{2}$$

$$p^{(n+1)}(w \mid t_\mathrm{k}) = \frac{c(w, D_k)\hat{p}^{(n)}(w)}{\sum_i c(w_i, D_k)p^{(n)}(w_i)} \tag{3}$$

Here, β is a coefficient accounting for the background noise, t_k denotes the translation model for topic signature t_k and $c(w, D_k)$ is the frequency count of term w in document set D_k (means the appearance times of w in D_k) , and C denotes the background collection, which is the set of all word occurrences in the corpus. In practice, the EM algorithm is used to estimate the translation model in the formulas (2) (3).

The cluster model with semantic smoothing (or referred to as semantic smoothing model) is estimated using a composite model $p_\mathrm{bt}(w|c_\mathrm{j})$, which means the likelihood of each vocabulary word w generated by a given document cluster c_j after smoothing. It has two components: a simple language model $p_\mathrm{b}(w|c_\mathrm{j})$ and a topic signature (multiword phrase) translation model $p_t(w|c_j)$. The influence of two components is controlled by the translation coefficient (λ) in the mixture model.

$$p_{bt}(w|c_j) = (1-\lambda)p_b(w|c_j) + \lambda p_t(w|c_j) \quad (4)$$

$$p_b(w|c_j) = (1-\alpha)p_{ml}(w|c_j) + \alpha p(w|C) \quad (5)$$

$$p_t(w|c_j) = \sum_k p(w|t_k)p_{ml}(t_k|c_j) \quad (6)$$

In simple language model (5), α is a coefficient controlling the influence of the background collection model $p(w|C)$ and $p_{ml}(w|c_j)$ is a maximum likelihood estimator cluster model. They can be computed using formulas (7) and (8). In translation model (6), t_k denotes the topic signatures (multiword phrases) extracted from documents in cluster c_j. The probability of translating t_k to individual term (word) is estimated using formulas (1) (2) (3). The maximum likelihood estimator of t_k in cluster c_j can be estimated with (9), where $c(w, c_j)$ denotes the frequency count of word w in cluster c_j, and $c(w, C)$ is the frequency count of word w in the background corpus. $c(t_k, c_j)$ is the frequency count of topic signature t_k (multiword phrase) in cluster c_j. The function of the translation model is to assign reasonable probability to core words in the cluster.

$$p_{ml}(w|c_j) = \frac{c(w,c_j)}{\sum_{w_i \in c_j} c(w_i,c_j)} \quad (7)$$

$$p(w|C) = \frac{c(w,C)}{\sum_{w_i \in C} c(w_i,C)} \quad (8)$$

$$p_{ml}(t_k|c_j) = \frac{c(t_k,c_j)}{\sum_{t_i \in c_j} c(t_i,c_j)} \quad (9)$$

Due to the likelihood of word w generated by a given cluster $p(w|c_j)$ can be obtained by the cluster model with semantic smoothing, the remaining problem for the clustering of text document is how to estimate the likelihood of a document d generated by a cluster. The log likelihood of document d generated by the j-th multinomial cluster model is described in formula (10), where $c(w, d)$ denotes the frequency count of word w in document d and V denotes the vocabulary.

$$\log p(d|c_j) = \sum_{w \in V} c(w,d) \log p(w|c_j) \quad (10)$$

Compared to semantic smoothing model, traditional similarity-based approaches just uses the technique of frequency count of words (which is similar to the simple language model $p_b(w|c_j)$) and does not takes into account the translation model $p_t(w|c_j)$). As shown in [7], semantic smoothing model is efficient to improve the clustering quality for traditional static text document clustering. However, it can not be directly used in the dynamical text data streams environment. The key reason is that, in text data stream, the text data comes as a continuous stream and it is hard to get the background collection model of all document data point of text data streams in advance. That is, it is hard to determine $p(w|C)$ in $p_b(w|c_j)$.

In this paper, we present an improved semantic smoothing model in which the background model is not included and set coefficient α as zero. Then we define the semantic smoothing model as follows.

$$p_{bt}(w \mid c_j) = (1 - \lambda) p_{ml}(w \mid c_j) + \lambda p_t(w \mid c_j) \tag{11}$$

From the formula (11), we can see that the improved semantic smoothing model also consists of two components, one is $p_{ml}(w|c_j)$ (which consists of frequency count of words) and the other is $p_t(w|c_j)= \sum_k p(w \mid t_k) p_{ml}(t_k \mid c_j)$.

4 The Proposed Online Clustering Algorithm

4.1 Cluster Statistics Structure

In order to achieve greater accuracy in the clustering process, we also maintain a high level of granularity in the underlying data structures. We refer to such cluster statistic structure as cluster profile in which the semantics are captured by the improved semantic smoothing model. Similar to the formula (11), in cluster profile, we maintain two kinds of weighted sums of the components $p_{ml}(w|c_j)$ and $p_t(w|c_j)$.

Definition 3 (Weighted Sum of Frequency Count). The weighted sum of frequency count for word w_i in cluster c is defined as $w\_c\ (w_i, c) = \sum_{d \in c} f(t - T_d) c(w_i, d)$. Here, $c(w_i, d)$ denotes the frequency count of w_i in document d, T_d is the arrival time of document d and $f(t - T_d)$ is the weight for word w_i in document d. Similarly, the weighted sum of frequency count for topic signature t_k in the cluster c is defined as $w\_c\ (t_k, c) = \sum_{d \in c} f(t - T_d) c(t_k, d)$, where $c(t_k, d)$ denotes the frequency count of t_k in document d.

Definition 4. (Weighted Sum of Translation). The weighted sum of topic signature translation probability in cluster c for word w_i is defined as $w\_t\ (w_i, c)$ = $\sum_k p(w_i \mid t_k) w\_c(t_k, c) = \sum_k p(w_i \mid t_k) (\sum_{d \in c} f(t - T_d) c(t_k, d))$. Here, $c(t_k$, $d)$ denotes the frequency count of topic signature t_k in document d, $p(w_i|t_k)$ denotes the probability of translating topic signature t_k to word w_i , T_d is the arrival time of document d and $f(t-T_d)$ is the weight for topic signature t_k in document d.

Definition 5. (Cluster profile, CP). A cluster profile $D(t, c)$ for a document cluster c at time t is defined to as a tuple ($\overline{DF2}$, $\overline{DF1}$, s, l). Consider wb denotes the number of distinct words in the dictionary V, then each tuple components is defined as follows.

- The vector $\overline{DF\,2}$ contains *wb* entries and the *i* entry $\overline{DF\,2}_i$ is defined as *w_c* (w_i, *c*).
- The vector $\overline{DF\,1}$ contains *wb* entries and the *i* entry $\overline{DF\,1}_i$ is defined as *w_t* (w_i, *c*).
- The entry *s* is defined as $\sum_k w\_c(t_k, c)$, which denotes the summation of *w_c* (t_k, *c*) for all the topic signature t_k in the cluster *c*.
- The entry *l* denotes the last time when cluster *c* is updated.

Then we can estimate the cluster model with semantic smoothing using formula (12) in which $\frac{\overline{DF\ 2}_i}{\sum_i \overline{DF\ 2}_i}$ and $\frac{\overline{DF\ 1}_i}{s}$ denotes the weighted form of the components $p_{ml}(w|c_j)$ and $p_t(w|c_j)$ respectively.

$$p_{bt}^{'}(w_i \mid c_j) = (1-\lambda)\frac{\overline{DF\ 2}_i}{\sum_i \overline{DF\ 2}_i} + \lambda \frac{\overline{DF\ 1}_i}{s} \tag{12}$$

Interestingly, we also find *CP* structure also has some similar properties for clustering process as the cluster droplet [4].

Property 1 (Additivity). Additivity describes the variation of cluster profile after two clusters c_1 and c_2 are merged as $c_1 \cup c_2$. Consider two cluster profiles as $D(t,c_1)$= ($\overline{DF\,2}(c_1)$, $\overline{DF\,1}(c_1)$, s_{c1}, l_{c1}) and $D(t,c_2)$= ($\overline{DF2}(c_2)$, $\overline{DF\,1}(c_2)$, s_{c2}, l_{c2}). Then $D(t,c_1 \cup c_2)$= ($\overline{DF\,2}(c_{12})$, $\overline{DF1}(c_{12})$, s_{c12}, l_{c12}) can be defined by tuple ($\overline{DF\,2}(c_1)+\overline{DF\,2}(c_2)$, $\overline{DF\,1}(c_1)+\overline{DF1}(c_2)$, $s_{c1}+s_{c2}$, $max(l_{c1}, l_{c2})$).

Proof

(1) For the *i* entry of $\overline{DF2}(c_{12})$, $\overline{DF2}(c_{12})_i$ = *w_c* (w_i, $c_1 \cup c_2$) = $\sum_{d \in c1 \cup c2} f(t-T_d)c(w_i,d) = \sum_{d \in c1} f(t-T_d)c(w_i,d) + \sum_{d \in c2} f(t-T_d)c(w_i,d)$ = *w_c* (w_i, c_1)+ *w_c* (w_i, c_2)= $\overline{DF\ 2}(c_1)_i + \overline{DF2}(c_2)_i$.

(2) For the *i* entry of $\overline{DF1}(c_{12})$, $\overline{DF1}(c_{12})_i$ =*w_t* (w_i,c_1 $\cup c_2$) = $\sum_k p(w \mid t_k) w\_c(t_k, \mathrm{c}_1 \cup c_2)$ = $\sum_k p(w_i \mid t_k)(\sum_{d \in c1 \cup c2} f(t-T_d)c(t_k,d))$ = $\sum_k p(w_i \mid t_k)(\sum_{d \in c1} f(t-T_d)c(t_k,d) + \sum_{d \in c2} f(t-T_d)c(t_k,d))$ = $\sum_k p(w \mid t_k)(w\_c(t_k,\mathrm{c}_1) + w\_c(t_k,\mathrm{c}_2))$ = $\sum_k p(w \mid t_k) w\_c(t_k,\mathrm{c}_1)$ + $\sum_k p(w \mid t_k) w\_c(t_k,\mathrm{c}_2)$ = *w_t* (w_i, c_1)+ *w_t* (w_i, c_2) = $\overline{DF1}(c_1)_i + \overline{DF1}(c_2)_i$.

(3) For s_{c12}, s_{c12}= $\sum_k w\_c(t_k, c_1 \cup c_2) = \sum_k \sum_{d \in c1 \cup c2} f(t-T_d)c(t_k,d)$ = $\sum_k(\sum_{d \in c1} f(t-T_d)c(t_k,d) + \sum_{d \in c2} f(t-T_d)c(t_k,d)) = \sum_k w\_c(t_k, c_1) + \sum_k w\_c(t_k, c_2)$ = $s_{c1}+ s_{c2}$.

(4) For l_{c12}, the proof is trivial since the last updated time of the merged cluster is the later of the original two ones.

Property 2 (Updatability). Updatability describes the variation of cluster profile after a new document is added into the clusters. Consider a new document *d* is merged into a cluster *c*, the current cluster profile $D_b(t,c_b)$= ($\overline{DF2}\ (c_b)$, $\overline{DF1}(c_b)$, s_{cb}, l_{cb}). Then the updated cluster profile is denoted as $D_a(t,c_a)$= ($\overline{DF2}(c_a)$, $\overline{DF1}(c_a)$, s_{ca}, l_{ca}). Here $\overline{DF2}(c_a)_i = \overline{DF2}(c_b)_i + c(w_i, d)$, $\overline{DF1}(c_a)_i = \overline{DF1}(c_b)_i + \sum_k p(w_i \mid t_k) c(t_k, d)$, s_{ca}=s_{cb}+ $\sum_k c(t_k, d)$ and $l_{ca} = t$.

Actually, property2 can also be viewed as the special case of property 1 in which one of the two clusters to be merged only consists of one document.

Property 3 (Fading Property). Fading Property describes the variation of cluster profile with time. Consider the cluster profile at the time t_1 is $D(t_1, c_{t_1})$= ($\overline{DF\,2}(c_{t1})$, $\overline{DF\,1}(c_{t1})$, s_{ct_1}, l_{ct_1}) and no document is added to the cluster c_{t_1} during [t_1, t_2]. Then the cluster profile at the time t_2 is defined as $D(t_2, c_{t_2})$= ($\overline{DF\,2}(c_{t2})$, $\overline{DF1}(c_{t2})$, s_{ct_2}, l_{ct_2}), where c_{t_2}=c_{t_1}, $\overline{DF2}(c_{t2})$ = $\overline{DF\,2}(\mathrm{c}_{t1})$ $*2^{-\zeta(t2-t1)}$, $\overline{DF\,1}(c_{t2})$ = $\overline{DF1}(c_{t1})$ $*2^{-\zeta(t2-t1)}$, s_{ct_2} = $s_{ct_1}$$*2^{-\zeta(t2-t1)}$ and $l_{ct_2} = l_{ct_1}$.

Proof

(1) For the *i* entry of $\overline{DF2}(c_{t2})$, $\overline{DF\,2}(c_{t2})_i$ = w_c (w_i, c_{t2}) = $\sum_{d \in ct2} f(t2 - T_d) c(w_i, d)$ = $\sum_{d \in ct2} f(t2 - t1 + t1 - T_d) c(w_i, d)$ = $\sum_{d \in ct1} 2^{-\zeta(t2-t1+t1-T_d)} * c(w_i, d)$ = $\sum_{d \in ct1} 2^{-\zeta(t2-t1)} * 2^{-\zeta(t1-T_d)} * c(w_i, d) = 2^{-\zeta(t2-t1)} * \sum_{d \in ct1} 2^{-\zeta(t1-T_d)} * c(w_i, d) = \overline{DF\,2}(c_{tl})_i$ $*2^{-\zeta(t2-t1)}$.

(2) For the *i* entry of $\overline{DF1}(c_{t2})$, $\overline{DF1}(c_{t2})_i$ =w_t (w_i, c_{t2}) = $\sum_k p(w \mid t_k) w\_c(t_k, \mathrm{c}_{t2})$ = $\sum_k p(w_i \mid t_k)(\sum_{d \in ct2} f(t2 - T_d) c(t_k, d))$ = $\sum_k p(w_i \mid t_k)(\sum_{d \in ct2} f(t2 - t1 + t1 - T_d) c(t_k, d))$ = $\sum_k p(w_i \mid t_k) \sum_{d \in ct1} 2^{-\zeta(t2-t1)} * 2^{-\zeta(t1-T_d)} * c(w_i, d)$ = $\overline{DF\,1}(c_{t1})_i$ $*2^{-\zeta(t2-t1)}$.

(3) For s_{ct2}, s_{ct2}= $\sum_k w\_c(t_k, c_{t2})$ = $\sum_k \sum_{d \in ct2} f(t2 - T_d) c(t_k, d)$ = $\sum_k \sum_{d \in ct1} f(t2 - t1 + t1 - T_d) c(t_k, d)$ = $\sum_k \sum_{d \in ct1} 2^{-\zeta(t2-t1)} * 2^{-\zeta(t1-T_d)} * c(t_k, d)$ = $s_{ct1}$$*2^{-\zeta(t2-t1)}$.

(4) For l_{ct2}, the proof is trivial since no document is added to cluster c_{t1} during [t_1, t_2].

4.2 The Online Clustering Algorithm

Our online clustering algorithm *OCTS* includes two phases: (1) offline initialization process, (2) online clustering process. The detailed description of *OCTS* algorithm framework is given in Fig.1.

The offline initialization process corresponds to line 1-2. In detail, *OCTS* first reads in the retrospective documents stored in disk as the training text data set. From the training document set, *OCTS* generates the topic signature translation model. In our algorithm implementation, the topic signature translation model is estimated by matrix M(wb*tb), where *wb* denotes the number of vocabulary words and *tb* denotes the number of topic signatures, and the data element M_{ik} of matrix represents $p(w_i|t_k)$. Then *OCTS* reads in the first *k* documents from the text data stream and build the initial cluster profiles *CP*s (each for one cluster) using definition 5.

The online clustering process corresponds to line 3-20. In this process, as a new text document arrives, firstly all the *CP*s would be updated using property 3. Next, the probability $p'_{bt}(w_i | c_j)$ is computed by formula (12). Then the similarity between document *d* and each cluster is estimated using formula (10). Then if the similarity between document *d* and its most similar cluster is less than specified threshold *MinFactor*, the most inactive cluster is deleted, and a new cluster is created. We associate the new cluster with the ID same to the deleted cluster's ID. Then the document *d* is assigned to the new cluster using property 2. Otherwise, document *d* would be assigned to the most similar cluster using property 2.

Algorithm: *OCTS*
Inputs: A stream of text data, a training data set *D*, *k* is the number of clusters, *MinFactor* is the threshold.
Output: A set of *k* cluster profiles (D(t, c_1),…., D(t, c_k))
Method:

```
1.  Extract words and multiword phrases, and build the translation model for the training data set $D$. /* Notice that the extraction process of words and phrase are the same as in [7].*/
2.  Read in the first $k$ documents from the text data stream and generate $k$ cluster profiles (D($t$,$c_1$),….,D($t$,$c_k$));
    /*The while loop is the online clustering process*/
3.  While (the stream is not empty) do
          /*Step4-5 is to update all CP using the fading property */
4.        $t$= GetCurrentTimestamp();
5.        Update all cluster profiles using property 3;
          /*Step 6-8 is the online building of the cluster model */
6.        For each cluster $j$
7.           For each word $w_i$
8.              The cluster model with semantic smoothing $p'_{bt}(w_i | c_j)$ is estimated using formula (12);
          /*Step 9-12 is to find the most similar cluster to document $d$ */
```

Fig. 1. The description of OCTS algorithm framework

```
9.      Read in the next document $d$ of text data stream and extract words and
    multiword phrases from $d$;
10.     For each cluster $j$
11.         The similarity $p(d \mid c_j)$ between document $d$ and $c_j$ is estimated with
    cluster model $p'_{bt}(w_i \mid c_j)$ using formula (10);
12.      $AssignID = \text{argmax}_j\, p(d \mid c_j)$  //Get the ID of the most similar cluster to $d$
        /* Step 13-17 is to delete the most inactive cluster and create a new cluster*/
13.     If ( $p(d \mid c_{AssignID}) < MinFactor$)     //$MinFactor$ is the threshold
14.        $NID = \text{argmin}_j\, D(t,c_j).l$;       /*Get the most inactive cluster's ID,
    if there are more than one inactive clusters then we randomly choose one)*/
15.        Delete the most inactive cluster $c_{NID}$;
16.        Create a new empty cluster and associate it with ID=$NID$ and build the
    cluster profile of the new cluster using definition 5;
17.        Assign document $d$ to the new cluster profile using property 2;
18.      Else
           /*Step 19 is to assign document $d$ to its most similar cluster */
19.        Assign document $d$ to cluster profile with ID = $AssignID$ using property 2;
20. End while
```

Fig. 1. (*continued*)

5 The Experimental Results

In the experimental studies, we compare our text streams clustering method with the framework proposed in [4] (denoted as *Ostc*) and the single pass clustering method in [5] (denoted as *SPstc*) in terms of clustering accuracy. All clustering algorithms are implemented by java 1.4. Our experiment is performed on AMD 1.60G, 240M memory, 40G hard disk and Window XP OS. The normalized mutual information (*NMI*) evaluation function is used to evaluate the clustering quality, and 20ng-newsgroups (*20NG*) (20000 documents) corpus [7] is used as the test data set. To simulate the real stream environment, we randomly give every document a time stamp to stand for the arrival time and create three different text data streams with different document sequence (denoted as Streaml, Stream2 and Stream3). By default, in our experiments, we randomly choose 500 documents as the training set and set the translation coefficient λ=0.6, cluster number k=20, stream speed is 100doc/s, half life=1s and *MinFactor*=0.05.

The first set of experiments is about the clustering quality comparison with different streams and the results are given in Fig.2-Fig.4. In this set of experiments, there are two kinds of results with different test granularities. In the left figures, the NMI values are compared by every 1 second interval (fine granularity). In the right figures, the NMI value is estimated by every 50 seconds interval (coarser granularity).

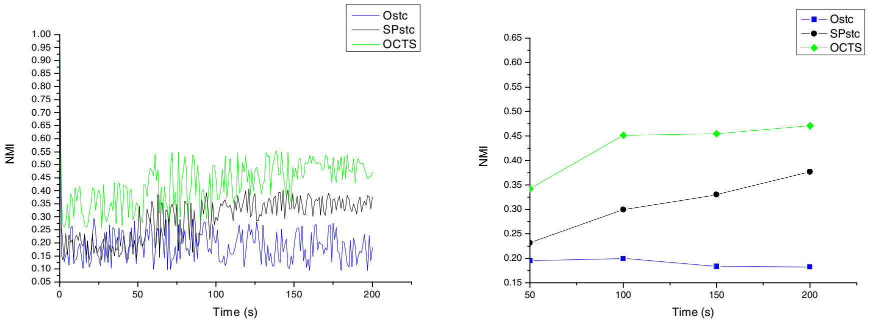

Fig. 2. Clustering Quality Comparison of Stream 1

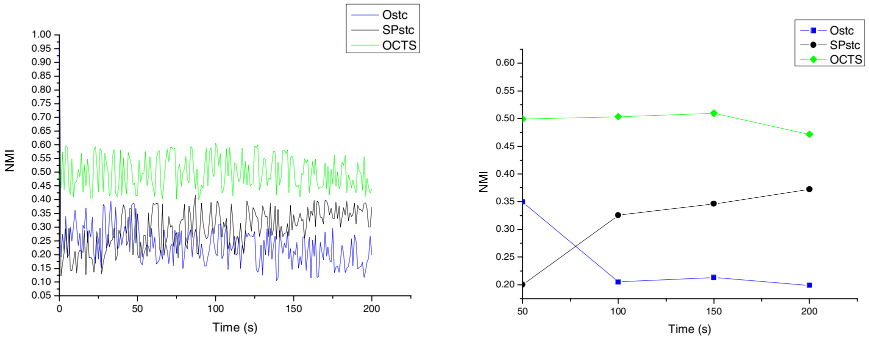

Fig. 3. Clustering Quality Comparison of Stream 2

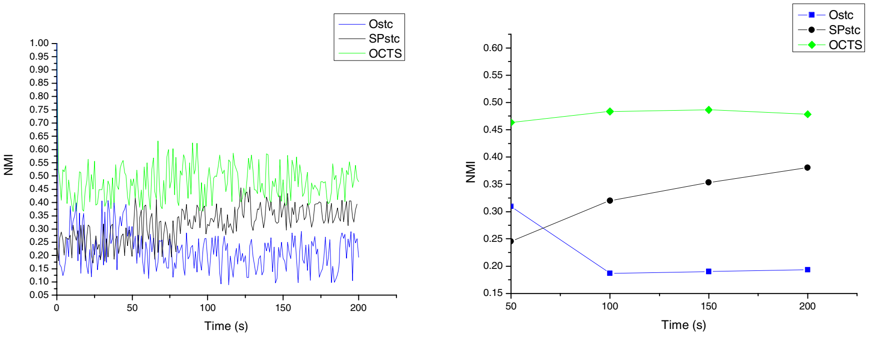

Fig. 4. Clustering Quality Comparison of Stream 3

From the above results, it is easy to know our proposed method *OCTS* obviously outperform *Ostc* and *SPstc* in which both TF [4] and incremental IDF [5] schemas are used. The results also show the effectiveness of semantic smoothing model in the aspect of improving the clustering quality.

The second set of experiments about the clustering quality of *OCTS* with different translation coefficient (λ) and different text data streams is also studied. The results are given in Fig.5-Fig.8. From the results, it is easy to know that the NMI values of *OCTS* with different λ are also better than the NMI values of *Ostc* and *SPstc* (referred to Fig.2, Fig.3 and Fig.4). From these results, we can also know that NMI values of *OCTS* will increase with the increase of the translation coefficient till the peak point (between 0.2 and 0.6) and then go downward. That is, our method is most effective as λ is between 0.2 and 0.6.

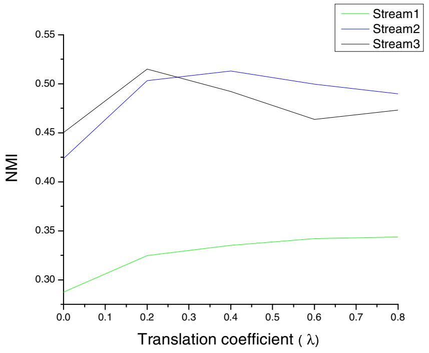

Fig. 5. The variance of the clustering quality with λ (test time: 50 seconds)

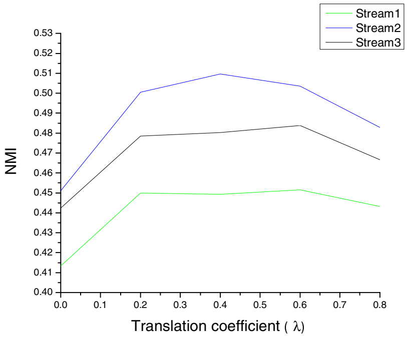

Fig. 6. The variance of the clustering quality with λ (test time: 100 seconds)

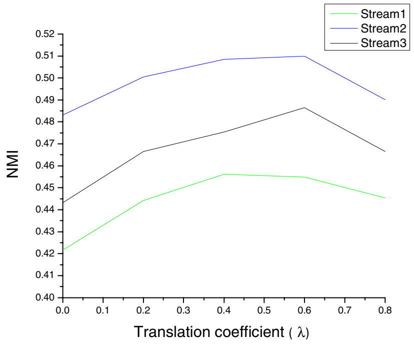

Fig. 7. The variance of the clustering quality with λ (test time: 150 seconds)

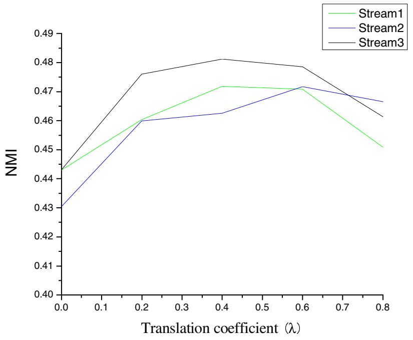

Fig. 8. The variance of the clustering quality with λ (test time: 200 seconds)

6 Conclusions

In this paper, we give an improved semantic smoothing model which is suitable for text data stream environment. Then we use the improved semantic model to improve the clustering quality and present an online clustering algorithm for clustering massive text data streams. In our algorithm, a new cluster statistics structure, cluster profile, is presented in which the semantics of text data streams are captured by the semantic smoothing model. We also present the experimental results illustrating the effectiveness of our technique.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (60573097), Natural Science Foundation of Guangdong Province (05200302, 06104916), Research Foundation of Science and Technology Plan Project in Guangdong Province (2005B10101032), Specialized Research Fund for the Doctoral Program of Higher Education (20050558017), and Program for New Century Excellent Talents in University of China (NCET-06-0727).

References

1. Aggarwal, C.C.: A Framework for Diagnosing Changes in Evolving Data Streams. In: Proc. ACM SIGMOD 2003, pp. 575–586 (2003)
2. Agrawal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: Aberer, K., Koubarakis, M., Kalogeraki, V. (eds.) Databases, Information Systems, and Peer-to-Peer Computing. LNCS, vol. 2944, pp. 81–92. Springer, Heidelberg (2004)
3. Agrawal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for projected clustering of high dimensional data streams. In: Jonker, W., Petković, M. (eds.) SDM 2004. LNCS, vol. 3178, pp. 852–863. Springer, Heidelberg (2004)
4. Aggarwal, C.C., Yu, P.S.: A Framework for Clustering Massive Text and Categorical Data Streams. In: Proc. SIAM conference on Data Mining, pp. 477–481 (2006)
5. Yang, Y., Pierce, T., Carbonell, J.: A study of retrospective and on-line event detection. In: Proc. ACM SIGIR 1998, pp. 28–36. ACM Press, New York (1998)
6. Zhong, S.: Efficient Streaming Text Clustering. Neural Networks 18(5-6), 790–798 (2005)
7. Xiaodan, Z., Zhou, X., Hu, X.: Semantic Smoothing for Model-based Document Clustering. In: Perner, P. (ed.) ICDM 2006. LNCS (LNAI), vol. 4065, pp. 1193–1198. Springer, Heidelberg (2006)
8. Zhou, X., Hu, X., Zhang, X., Lin, X., Song, I.-Y.: Context-Sensitive Semantic Smoothing for the Language Modeling Approach to Genomic IR. In: Proc. ACM SIGIR 2006, pp. 170–177 (2006)
9. Fung, G.P.C., Yu, J.X., Yu, P.S., Lu, H.: Parameter Free Bursty Events Detection in Text Streams. In: Draheim, D., Weber, G. (eds.) TEAA 2005. LNCS, vol. 3888, pp. 181–192. Springer, Heidelberg (2006)