Discovering Trend-Based Clusters in Spatially Distributed Data Streams

Anna Ciampi, Annalisa Appice, and Donato Malerba

Dipartimento di Informatica, Università degli Studi di Bari
via Orabona, 4 - 70126 Bari - Italy
{aciampi,appice,malerba}@di.uniba.it

Abstract. Many emerging applications are characterized by real-time stream data acquisition through sensors which have geographical locations and/or spatial extents. Streaming prevents from storing all data from the stream and performing multiple scans of the entire data sets as normally done in traditional applications. The drift of data distribution poses additional challenges to the spatio-temporal data mining techniques. We address these challenges for a class of spatio-temporal patterns, called trend-clusters, which combine the semantics of both clusters and trends in spatio-temporal environments. We propose an algorithm to interleave spatial clustering and trend discovery in order to continuously cluster geo-referenced data which vary according to a similar trajectory (trend) in the recent past (window time). An experimental study demonstrates the effectiveness of our algorithm.

1 Introduction

Spatio-temporal data mining has recently gained considerable attention from both the research and practitioner communities. The main reason for this interest is that datasets containing prominent spatial and temporal data elements are growing rapidly due to the daily collection of geo-referenced data through sensor networks.

Despite the significant advances made recently in spatio-temporal data mining, many existing spatio-temporal data analysis approaches take only a static view of the geo-spatial phenomena [7]. These approaches extract a finite set of data points based on user-provided criteria (e.g., the space and time of interest) and address data mining tasks for static spatio-temporal data only. However, a static perspective is inadequate as data often arrive dynamically, continuously and with a drift of the underlying data distribution. On the other hand, a dynamic perspective poses challenges such as avoiding multiple scans of the entire data sets, optimizing memory usage, and mining only the most recent patterns.

The sliding window model [3] is a natural choice to efficiently mine a stream of data where the most recent data observations are considered to be more critical and preferable. This model improves the space/time efficiency of learning by capitalizing on the fact that multiple scans due to pattern evaluation are limited to basic slides.

The sliding-window model was originally defined for a single, non-spatially related data stream source. The main contribution of this work is its extension to the case of a several *spatially distributed* data stream sources. This way, the model can be exploited in a wider range of ubiquitous knowledge applications, which are characterized by both temporal and spatial locality [16].

The scenario we consider is that of a stream composed by sets of observations for a numeric attribute (theme) which are transmitted at consecutive time points by a (variable) number of sources. These stream sources are identified by a progressive number and geo-referenced. By taking into account both the spatial arrangement of the transmitting sources and the temporal arrangement of streamed data, we design a framework to mine a kind of spatio-temporal patterns, called *trend clusters*. These patterns are *spatial clusters* of sources which transmit values whose temporal variation, called *trend polyline*, is similar over the recent window time. We introduce an algorithm, named TRUST (TRend based clUstering algorithm for Spatio-Temporal data stream), for mining these trend-clusters from spatially distributed data streams. The algorithm makes no assumption on the number of spatial clusters as well as on their shape.

The paper is organized as follows. The next section formalizes the problem statement. Section 3 revises related works. Section 4 describes the algorithm and Section 5 reports an experimental study.

2 Problem Statement

In this work, spatio-temporal data are modeled according to the *snapshot* data model [2], where the time domain is linear, absolute and discrete, the spatial domain is field-based and the attribute domains are variable. For every time point of the stream, values collected at this time point form a time-stamped thematic layer. This way, a layer is a collection of geo-referenced values measured for a thematic attribute at the time point. Time-stamped values vary over a continuous space surface which is modeled in the field-based data model [19] as a function: $f\colon \mathbb{R}^2 \times T \mapsto \textit{Attribute domain}$, where $\mathbb{R}^2$ is an Euclidean representation of the space surface and T is the time line. For the kind of applications considered in this work, it is reasonable to assume that the locations of the transmitting sources are fixed. Therefore, field functions are actually defined over only a finite set of positions, those of active sources. Though finite, field domains can vary over consecutive layers, since the number of transmitting sources can change in time (sources may become inactive and transmit no data for some time interval).

In the stream perspective, thematic layers arrive continuously at consecutive time-points. According to the sliding window schema [9], the stream is broken down into *slides* of p layers arriving in series. A sliding window of size w is composed by w consecutive slides (see Figure 1). Both window and slide sizes are manually defined. In particular, window size depends on the time horizon we are interested in processing. Differently, the slide size should be tuned in order to find the best trade-off between learning time and accuracy. Each time a slide flows in, patterns are generated locally to the slide. Then these local patterns

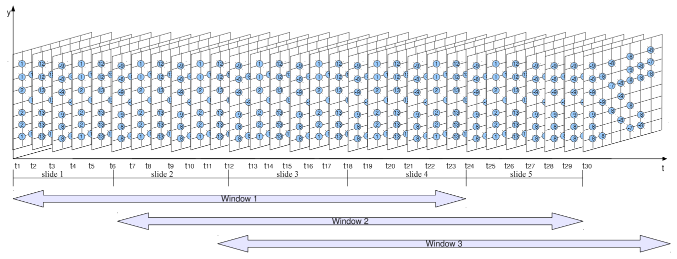

Fig. 1. Sliding windows with $p = 6$ and $w = 4$.

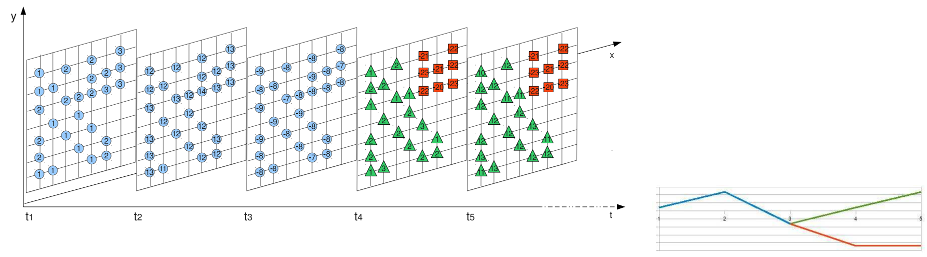

Fig. 2. Trend clusters: the blue cluster groups circle sources which vary according to the blue trajectory over t1, t2 and t3. The red (green) cluster groups squared (triangular) sources which vary according to the red (green) trajectory over t4 and t5. Numbers are the values transmitted by the sources.

are used to approximate the set of patterns in the recent sliding window. This way, multiple scans due to pattern evaluation are limited to slides, thus making the discovery process more efficient both in time and space.

Patterns considered in this work, called *trend clusters*, are the result of *spatial clustering* and *trend discovery* with the sliding window model. Spatial clusters are obtained by processing data in a single time-stamped layer, so that sources in the same cluster model the continuity in space of the measured attribute, while sources in separate clusters model the variation over space [15]. Spatial continuity expresses the similarity between observed values over the spatial organization, which is defined by spatial relations (e.g., distance) between distinct sources. Sources grouped together in spatial clusters of consecutive layers model a spatially local trend for the measured attribute. Trend clusters for an attribute can be represented by colored trajectories over time (see Figure 2).

3 Related Works

In order to clarify the background of this work, related researches on clustering in data stream mining and spatio-temporal data mining are reported below.

Data Stream Mining. Guha et al. [10] have presented a constant factor of approximation for the k-Medians algorithm which performs a single pass clustering in data stream. Babcock et al. [4] have extended this algorithm by framing k-Medians algorithm in sliding window model. The limitation of both algorithms is that their results are often spherical clusters. They do not consider that clusters in data streams could be of arbitrary shape and arbitrary number. Aggarwal et al. [1] have proposed a two-phases clustering algorithm, called CluStream, which separates out the clustering process into an online micro-clustering phase and an offine macro-clustering phase. The problem with CluStream is the predefined constant number of micro-clusters. Additionally, since a variant of k-means is adopted to get the final macro-clusters, a "natural" macro-cluster may be split into two parts. To discover arbitrary clusters in data stream and to handle outliers, Cao et al [6] have proposed a density-based clustering algorithm, called DenStream. Finally, unsupervised naive-Bayesian network algorithms which are designed to discover clusters in distributed data streams [17, 5] may be relevant for this work. In fact, a network is here employed to model spatial arrangement of data. Anyway, the kind of pattern we discover, that is, patterns with a polyline trajectory to describe how data vary in time within the cluster, requires a time series based processing of streamed data which is not performed by these naive Bayesian network algorithms.

Spatio-temporal Data Mining. Most of research in spatio-temporal data clustering is focused on the discovery of either trajectory clusters or moving clusters. Trajectory clusters group trajectories of similar shape. Vlachos et al. [20] have firstly defined a Least Common Subsequence distance which is used as a base to apply traditional (partitioning and hierarchical) clustering algorithms to object trajectories. Gaffney and Smyth [8] have proposed a clustering algorithm which models a trajectory as individual sequence of points generated from a regression model. Unsupervised learning is carried out using EM algorithm to cope with the cluster memberships. Nanni and Pedreschi [18] have adapted density-based algorithm to trajectory data by using a distance measure between trajectories. Differently, Lee et al. [13] have proposed a partition-and-group framework for clustering trajectories which partitions a trajectory into a set of segments, and then, groups similar segments into a cluster. Differently, the moving clusters group moving objects in clusters whose identity remains unchanged while cluster location and content may change over time. The key difference is that a trajectory cluster has a constant set of objects throughout its lifetime, while the content of a moving cluster may change over time (i.e., one or more object may leave the group or new objects may enter it). A seminal method to discover moving clusters from an history of recorded trajectories has been proposed in Kalnis et al. [12]. Spatial clusters are discovered at each snapshot by resorting to a static density-based clustering algorithm and results are then combined into a set of moving clusters. Li et al. [14] have proposed to exploit the micro-clustering originally proposed by Zhang et al. in [21] and extend it to moving micro-clustering. In this work, a moving micro-cluster denotes a group of objects that are not only close to each other at current time, but also likely to move together for a while.

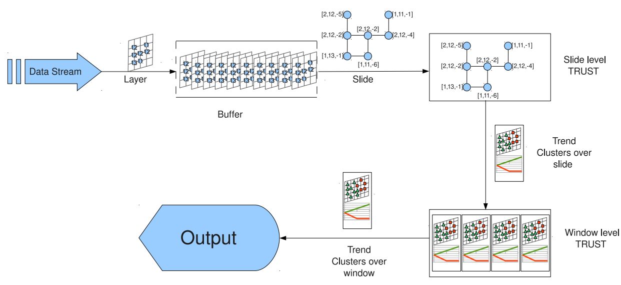

Fig. 3. The framework of TRUST.

4 The Algorithm

The framework of TRUST is reported in Figure 3. A buffer continuously consumes thematic layers and pours them slide-by-slide into the TRUST system. TRUST operations consist in: (1) buffering slide layers in a graph-based synopsis data structure, (2) discovering trend-clusters over the slide time (slide-level clustering), and then (3) approximating trend clusters by combining the slide-level trend cluster sets (window-level clustering). After a slide goes through TRUST, slide layers are discarded, while the set of trend cluster discovered over the slide is maintained in main memory for the window time. For each slide, the number of discovered trend clusters ranges between 1 (i.e., sources are all grouped in a single cluster) and the number of sources (one cluster for each source). Due to the sliding window model, once the set of trend clusters is locally discovered over the last income slide, the oldest set is discarded. The number p of layers in a slide, the number w of the slides composing the sliding window, the value-similarity threshold δ, the slide-level trend continuity threshold θ and the window-level trend continuity threshold ϵ are given before TRUST starts.

4.1 Buffer Synopsis Structure

Since TRUST is in charge of discovering trend clusters over the sliding windows of a data stream, it needs to maintain on-line a representation of the spatial organization which arises in the layers of the current sliding window.

As reported in [15], the spatial organization of a layer can be modeled by means of a graph $G(N, E)$. In this work, N is the set of nodes (sources) which transmit signals over at least one slide of the sliding window. E is a binary (spatial) relation between nodes, $E \subseteq \{\langle u, v\rangle | u, v \in N\}$ (e.g., an edge $\langle u, v\rangle \in E$ connects the nodes u and v iff the Euclidean distance between the corresponding sources is less than a threshold). Each node in N is assigned with w binary labels which describe the active/inactive state of the node at each slide of the sliding window. A node is active at a slide if it is active in at least one layer in the slide, inactive otherwise. The nodes of G which are active at the current slide defines the structure of the data synopsis where the layers are buffered for the time of the slide-level clustering phase. Similarly, the edges of E which connect nodes which are active on the slide identify the spatially close sources which are the only candidates to be grouped in the same cluster during the slide-level clustering phase. The entire edge set E is used during the window-level clustering phase to approximate the set of trend clusters for the entire window.

The graph G is constructed from scratch with the first slide which flows in the stream. After that, the structure of G is updated each time a new slide flows in the stream. The update operations consist in: (1) adding a new node (and the edges) to N (and E) in order to map a transmitting source which is firstly monitored in the sliding window, the state of this node is active over the current slide, inactive over the past $w-1$ slides of the sliding window; (2) updating an existing node f N by discarding the oldest state label and assigning an active label if the source transmits signals in the current slide (inactive otherwise); and (3) removing a node (and the edges) from N (and E) if the node was labeled as inactive over each slide of the sliding window.

Let us denote: N_A as the set of nodes which belong to N and are active in the current slide, and E_A as the set of edges which belong to E and connect active nodes in the current slide. $G_A(N_A, E_A)$ represents the graph structure of the synopsis where the lastly income slide layers are temporally stored for the slide-level clustering phase. For each active node $u \in N_A$, a p-sized bucket is stored in the synopsis data structure. This bucket is denoted as B_u and it stores the p values streamed by the corresponding source in the slide time (see Figure 4). This way, the slot $B_u[l]$ is the value buffered in u by the l^{th} layer which arrives in the slide. It is noteworthy that there are two cases in which missing values may be stored in a bucket. In the former case, a source, which has already been active in the past, does not transmit the signal for one or more layers in the current slide (e.g., the sensor is gone down or the transmission is not in time), the missing value is replaced by the lastly observed value for the source. In the latter case, a source transmits the first signal at a layer of the current slide, missing values which precede the first transmission are replaced with this firstly transmitted value.

4.2 Slide-level trend cluster Discovery

Before presenting how the slide-level trend cluster discovery is performed, we introduce some preliminary definitions.

Definition 1 (δ-close measure ψ_δ). *Let X be a continuous theme with domain $[\alpha, \beta]$ and δ be a real value in $[0,1]$. The δ-close measure is a function $\psi_\delta \colon X \times X \mapsto \{0,1\}$ such that* $\psi_\delta(x_1, x_2) = \begin{cases} 1 & \frac{\|x_1 - x_2\|_1}{\beta - \alpha} \leq \delta \\ 0 & \textit{otherwise} \end{cases}$.

Based on Definition 1, we define the E_δ^θ close relation between the edged nodes of G.

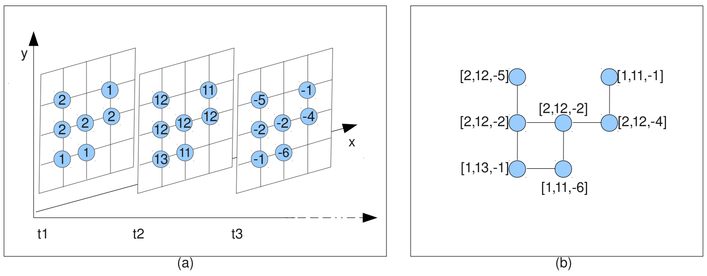

Fig. 4. The graph synopsis data structure (b) where layers (a) are buffered with $p = 3$.

Definition 2 (E^{θ}_{δ} close relation). *Let δ be a real value in $[0,1]$ and θ be a real value in $[0,1]$; E^{θ}_{δ} is the binary relation between the edged nodes of G_A ($E^{\theta}_{\delta} \subseteq E_A$) which is defined as $E^{\theta}_{\delta} = \{\langle u,v\rangle \in E | \sum_{i=1}^{p} \psi_{\delta}(B_u[i], B_v[i]) \geq \theta \times p\}$, where $B_u[i]$ ($B_v[i]$) is the i^{th} slot value in the bucket B_u (B_v).*

We use the E^{θ}_{δ} close relation to define the E^{θ}_{δ}-connectivity in G.

Definition 3 (E^{θ}_{δ}-connected). *A node u is E^{θ}_{δ}-connected to a node v ($u, v \in N_A$), with respect to E^{θ}_{δ}, iff: (i) $\langle u,v\rangle \in E^{\theta}_{\delta}$,* (direct connectivity) *(ii) or $\exists w \in N_A$ such that $\langle u,w\rangle \in E^{\theta}_{\delta}$ and w is E^{θ}_{δ}-connected from v* (undirect connectivity).

Finally, we define the function of *δ-homogeneity* over G_A.

Definition 4 (δ-homogeneity). *Let δ be a real value in $[0,1]$. The function δ-homogeneity : $2^N \mapsto \{true, false\}$ is defined as:*

$$\delta - homogeneity(N_i) = \begin{cases} true & \forall u,v \in N_i,\ \frac{\|\eta(B_u)-\eta(B_v)\|_1}{\beta-\alpha} \leq \delta \\ false & otherwise \end{cases}.$$

In Definition 4, the function η: $2^X \mapsto X$ returns the middle point (median) over a sequence of values in X. The choice of the median is motivated by the fact that it is robust, while the mean would be influenced by outliers. Finally, a trend cluster is defined as follows.

Definition 5 (trend cluster). *A trend cluster is the pair $C \Leftrightarrow P$ such that:*

1. *$C \subseteq N$ satisfies the following properties: (a) $\forall u,v \in C$: u E^{θ}_{δ}-connected to v, and (b) C is a δ-homogeneous node set , i.e., δ-homogeneity(C)=true;*
2. *$P = \langle \eta(C|_1), \eta(C|_2) \ldots, \eta(C|_p)\rangle$ where $C|_l$ is the set $\{B_u[l] | u \in C\}$ and $\eta(\cdot)$ is the median function.*

Note that, in Definition 5, the E^{θ}_{δ}-connectivity guarantees the (spatial) continuity of polylines which are associated to the nodes grouped in a cluster at

Algorithm 1 Slide-level discovery: $G_A(N_A, E_A) \mapsto \Gamma$

```
– Main routine
1: $\Gamma = \oslash$
2: for all $u \in N_A$ do
3:    if $u$ is $UNCLASSIFIED$ then
4:       $C \leftarrow \{u\}$
5:       $C \leftarrow expandCluster(C, G_A, u)$
6:       $P \leftarrow \langle \eta(C,1), \eta(C,2), \ldots, \eta(C,p) \rangle$
7:       $\Gamma \leftarrow \Gamma \cup \{C \Leftrightarrow P\}$
8:    end if
9: end for
– expandCluster $(C, G_A, u)$
1: $N_\theta^\delta(u) \leftarrow E_\delta^\theta neighborhood(u, G_A)$
2: if $\delta$-homogeneity$(C \cup N_\delta^\theta(u))$ then
3:    $C \leftarrow C \cup N_\delta^\theta(u)$
4:    for all $n \in N_\delta^\theta(u)$ do
5:       $C \leftarrow expandCluster(C, G_A, n)$
6:    end for
7: else
8:    for all $n \in N_\delta^\theta(u)$ do
9:       if $\delta$-homogeneity$(C \cup \{n\})$ then
10:         $C \leftarrow C \cup \{n\}$
11:         $C \leftarrow expandCluster(C, G_A, n)$
12:      end if
13:   end for
14: end if
```

each layer. The condition on $\delta-$homogeneity is added to guarantee that the entire cluster evolves with a *single* close polyline that is reasonably approximated with P. Closeness depends on δ and θ. Since a cluster is the node set of a sub-graph extracted from G by satisfying the conditions of E_δ^θ-connectivity and δ-homogeneity, TRUST resorts to a neighborhood-based graph partitioning algorithm-like in order to discover the trend clusters over slide.

The top-level description of the algorithm is in Algorithm 1. The key idea is to exploit the construction of node neighborhood based on the E_δ^θ close relation (namely E_δ^θ-neighborhood) and to grow clusters by merging partially overlapping E_δ^θ neighborhoods only if the resulting cluster is a δ-homogeneous node set. The definition of the E_δ^θ-neighborhood of a node u in G is reported below.

Definition 6 (E_δ^θ-neighborhood). *Let u be a node, the neighborhood $N_\delta^\theta(u)$ of u is defined as $N_\delta^\theta(u) = \{v | \langle u, v \rangle \in E_\delta^\theta \quad \wedge UNCLASSIFIED(v)\}$, where $UNCLUSSIFIED(v)$ means that v is not assigned to any cluster.*

In the *main routine* of Algorithm 1, a novel empty cluster C is created for a node $u \in N$ which is currently $UNCLASSIFIED$. C is firstly grown with u, then $expandCluster(C, u)$ grows C by using u as seed. In particular, $expandCluster(C, u)$ evaluates the property of $\delta - homogeneity$ for the node set $C \cup N_\delta^\theta(u)$ (see the call of $\delta - homogeneity(\cdot)$ function in Algorithm 1). If $C \cup N_\delta^\theta(u)$ is a δ-homogeneous node set, C is grown with $N_\delta^\theta(u)$. Otherwise, the addition of each neighbor $n \in N_\delta^\theta(u)$ is evaluated node-by-node. A neighbor is individually added to C only if the output cluster remains a $\delta-$ homogeneous node set. Each time C changes (by either adding an entire neighborhood or only few neighbors), $expandCluster(\cdot, \cdot)$ is recursively called to further grow C by considering the new clustered nodes as candidate seed of the expansion (see the recursive call of $expandCluster(\cdot, \cdot)$ in Algorithm 1).

When a cluster C is completely constructed (no further node can be added to), the polyline P, which describes how the clustered nodes evolve over the slide time, is built. P is the sequence of p points, that is defined as $P = \langle 1, \eta(C, 1)\rangle, \langle 2, \eta(C, 2)\rangle, \ldots, \langle p, \eta(C, p)\rangle$, where each point $\langle l, \eta(C, l)\rangle$ is the representative of the behavior of the cluster C over the l^{th} layer in the buffered slide. In Algorithm 1, the function $\eta(C, l)$ (that is called with argument $l = 1, \ldots, p$) returns the median of the values stored in the l^{th} slots for the nodes grouped in C.

Finally, each trend cluster $C \Leftrightarrow P$ is added to the cluster set Γ. Once all the nodes have been classified in a cluster, values in the buckets are discarded and the algorithm stops by returning Γ as output of the slide-level discovery process. Note that Γ can be intended as a reasonable summarization of the slide layers.

4.3 Window-level trend cluster Discovery

The discovery of trend clusters over a window capitalizes on the trend cluster sets, denoted as $\Gamma_1, \ldots, \Gamma_w$, which have been locally discovered for the slides of W. Indeed, while past slide layers are discarded, the discovered trend cluster sets are maintained for the window time. This way, the window-based discovery analyzes a reduced set of data. The window level clustering bases on the consideration that nodes, which are repeatedly grouped together in a cluster for each slide of the window, map sources which are close in space and which evolve with a close polyline over the entire window time. Based upon this consideration, the window-level component of TRUST determines the trend clusters over the window by clustering the spatially close sources which are continuously clustered together over the slides of the window. Clustering is performed based on the binary cluster relation denoted as $E_\epsilon^{\Gamma_1,\ldots,\Gamma_w}$ which is defined below.

Definition 7 ($E_\epsilon^{\Gamma_1,\ldots,\Gamma_w}$ window cluster relation). *Let ϵ be a real value in $[0, 1]$, $E_\epsilon^{\Gamma_1,\ldots,\Gamma_w}$ is defined over E as* $E_\epsilon^{\Gamma_1,\ldots,\Gamma_w} = \{\langle u, v\rangle \in E | \sum_{i=1}^{w} clustered(u, v, \Gamma_i) \geq \epsilon \times w\}$ *with* $clustered(u, v, \Gamma_i) = \begin{cases} 1 & \exists (C \Leftrightarrow P) \in \Gamma_i \colon u, v \in C \\ 0 & otherwise \end{cases}$.

The binary window cluster relation $E_\epsilon^{\Gamma_1,\ldots,\Gamma_w}$ identifies each pair of edged nodes of G which are clustered together over at least $\epsilon \times w$ slides of W. Hence, clusters over window are obtained by partitioning the graph $G(N, E)$ in completely connected sub-graphs according to the binary window cluster relation

Algorithm 2 Window-level discovery: $\mathit{\Gamma}_1, \mathit{\Gamma}_2, \ldots, \mathit{\Gamma}_w, G(E, N) \mapsto \mathit{\Gamma}$

```
– Main routine
 1: $E_\epsilon^{\Gamma_1,\ldots,\Gamma_w} \leftarrow determineWindowClusterRelation(\Gamma_1, \Gamma_2, \ldots, \Gamma_w, E)$
 2: $\Gamma \leftarrow \oslash$
 3: for all $u \in N$ do
 4:     if $u$ is $UNCLASSIFIED$ then
 5:         $C \leftarrow \{u\}$
 6:         $C \leftarrow expandCluster(C, E_\epsilon^{\Gamma_1,\ldots,\Gamma_w}, u)$
 7:         $P \leftarrow polyline(C, \{\Gamma_1, \ldots, \Gamma_w\})$
 8:         $\Gamma \leftarrow \Gamma \cup \{C \Leftrightarrow P\}$
 9:     end if
10: end for
– $expandCluster(C, E_\epsilon^{\Gamma_1,\ldots,\Gamma_w}, u)$
 1: for all $\langle u, v\rangle \in E_\epsilon^{\Gamma_1,\ldots,\Gamma_w}$ and $v$ is $UNCLUSSIFIED$ do
 2:     $C \leftarrow C \cup \{v\}$
 3:     $C \leftarrow expandCluster(C, E_\epsilon^{\Gamma_1,\ldots,\Gamma_w}, v)$
 4: end for
```

$E_\epsilon^{\mathit{\Gamma}_1,\ldots,\mathit{\Gamma}_w}$ (see Algorithm 2). Let C be a cluster over window W, C is assigned with the trend polyline P and $C \Leftrightarrow P$ is output as a trend cluster over W. P is built by sequencing the points of the polylines which are associated, slide by slide, to the clusters which include C (see the call of function *polyline*$(\cdot,\cdot)$ in Algorithm 2). Formally, $P = P^1, P^2, \ldots, P^w$ where:

$$P^j = \begin{cases} P_i^j & \exists (C_i^j \Leftrightarrow P_i^j) \in \mathit{\Gamma}_j \colon C \subseteq C_i^j \\ \text{null} & \text{otherwise} \end{cases} . \qquad (1)$$

5 Experiments

We evaluate TRUST with a synthetic data stream and two real data streams. Experiments evaluate accuracy and number of discovered patterns and the efficiency of learning. As a measure of the accuracy of the trend cluster set $\mathit{\Gamma}$, we use the average absolute percentage error ($MAPE$), a statistics that is widely used in time series [11]. MAPE computes the absolute error that is performed when the polylines of $\mathit{\Gamma}$ are used to fit data windowed in W. Formally,

$$MAPE(\mathit{\Gamma}, W) = \sum_{u \in N} mape(\mathit{\Gamma}, u|_W)/|N|,$$

where N denotes the source set over which the stream is transmitted over the window W and $u|_W$ denotes the series of $p \times w$ values which are streamed from the source $u \in N$ over the window W. $mape(\mathit{\Gamma}, u|_W)$ is the mean absolute percentage error which measures how the polyline P of the trend cluster $C \Leftrightarrow P$ fits real values streamed in the $u|_W$ (with $u \in C$). It expresses accuracy as a percentage, and is defined by $mape(\mathit{\Gamma}, u|_W) = \frac{1}{p \times w} \sum_{i=1}^{p \times w} \left\| \frac{(u[i|_W]) - P[i]}{(u[i|_W])} \right\|_1$ with $(C \Leftrightarrow P) \in \mathit{\Gamma}$ and $u \in C$.

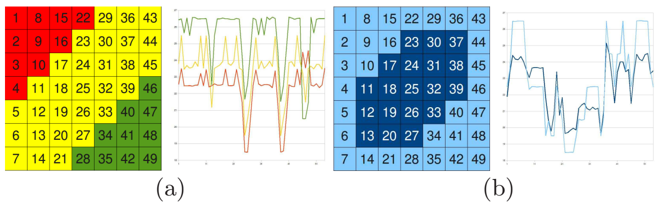

Fig. 5. The trend cluster configurations according to the synthetic data are generated.

Synthetic data stream. As a synthetic data stream we have considered the stream transmitted by 49 sources which are distributed over a squared 7×7 grid. A source is close to the neighbor sources which are located in the grid cells around the source (see Figure 5). Let X be the thematic attribute of the stream, X ranges in $[18, 27]$. The stream is obtained by sequencing 54 layers of measurements of X which are generated according to the trend based cluster configuration reported in Figure 5.a. and 54 layers which are generated according to the trend based cluster configuration reported in Figure 5.b. Each source measures values which evolve according to the polyline representing the trend of the corresponding cluster. For each layer, for each source the polyline value is noised with an error generated according to the Normal distribution N(0,1). Experiments are run with $p = 9$, $w = 6$, $\delta = 10\%[27 - 18] = 0.9$, $\theta = 0.8$ and $\epsilon = 1$. θ is set to a value different from 1 in order to take into account variation in data due to the noise. This way TRUST processes twelve slides and discovers trend clusters over seven consecutive sliding windows. The cluster configurations discovered for each sliding window are reported in Figure 6. As expected, the clustering results show that TRUST is able to exactly detect the cluster configuration that is defined in (a) over the first window (slides 1-6) and the cluster configuration that is defined in (b) over the last window (slides 6-12) of the processed stream. The results of clustering over intermediate sliding windows correctly reveal the concept drift from the configuration (a) toward the configuration (b). Indeed, to adapt the discovered cluster configuration to the effective evolution trend of the windowed data, TRUST moves the sources 5, 6, 7, 14, 21, 29, 36, 43, 44 and 45 in two new spatial clusters. This way, TRUST still identifies groups of spatially contiguous sources whose trend is approximately the same over the window under consideration. It is noteworthy that 5, 6, 7, 14 and 21 are grouped in one spatial cluster, while 29, 36, 43, 44 and 45 are grouped in a separate cluster over the intermediated sliding windows (slides 2-8, slides 3-9, slides 4-10, slides 5-11) due to the spatial discontinuity of sources, while the plot of the trend associated to these spatial clusters show two polylines which are overlapping. In Table 1, the mean absolute percentage error ($MAPE$) of trend cluster sets discovered over the sliding windows of data stream is reported. $MAPE$ provides an estimate of the accuracy of cluster polylines in fitting windowed data. The observed value of $MAPE$ is always low (it is less than 0.03 for each window), thus confirming that the trend discovery is accurate.

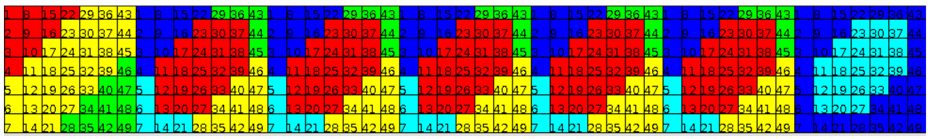

Fig. 6. The clusters discovered by TRUST over each sliding window of the synthetic data stream ($p = 9$, $w = 6$, $\delta = 10\%(27 - 18)$, $\theta = 0.8$ and $\epsilon = 1$).

Table 1. *MAPE* of trend cluster sets over sliding windows of processed data stream.

error	**Sliding windows**						
	[1-6]	[2-7]	[3-8]	[4-9]	[5-10]	[6-11]	[7-12]
MAPE	0.01885	0.01921	0.01975	0.02083	0.02229	0.02267	0.02290

Intel Berkley Lab data stream. Intel Lab data stream[1] contains real information collected from 54 sensors deployed in the Intel Berkeley Research lab between February 28th and April 5th. A sensor is considered to be close to each other sensor into the six meters range. The sensors which we consider in this experiment have collected timestamped temperature values once every 31 seconds (epoch). There are several epochs where no temperature value is transmitted by one or more sensors (missing values). We have processed the entire stream that is 2.3 about million readings (about 65000 layers) collected from these sensors in 73541 millisecs by setting $p = 20$ and $w = 5$, $\theta = 1$ and $\epsilon = 1$. Due to the space limitation, we describe only the trend cluster configurations which have been discovered by processing a subset of 5000 layers. Experiments are run with $\delta = 10\%[\beta - \alpha]$. Due to the presence of several outlier values, we use a box plot to derive a reasonable representation of the data range $[\alpha, \beta]$. The groups of streamed values collected over an initial calibration time are depicted through their five-number summaries (box plot): the smallest observation (sample minimum), lower quartile (Q_1), median (Q_2), upper quartile (Q_3), and largest observation (sample maximum). This way we derive $\alpha = Q_1 - 1.5*(Q_3 - Q_1) = 10.405$ and $\beta = Q_3 + 1.5 * (Q_3 - Q_1) = 37.085$. Hence, $\delta = 2.668$. To evaluate the effectiveness of sliding window model in the stream environment, we compare each trend cluster set obtained with $p = 20$ and $w = 5$ with the cluster results obtained by learning the full widowed data at once ($p = 100$ and $w = 1$). Elapsed time of the learning phase with sliding windows is reported in Figure 7.a. The comparison between $p = 20$ $w = 5$ and $p = 100$ $w = 1$ is plotted in Figure 7.b. We consider only the windows which cover the same portion of data stream (e.g., slides 1-5, 6-10, and so on). Since $\theta = 1$ and $\epsilon = 1$ the same trend clusters are discovered in both configurations, while the elapsed time is different due to the use of sliding window model. We can observe that TRUST capitalizes on the slide-based discovery to analyze a reduced set of data (slide), thus decreasing the time of learning in the online discovery process without affecting number and trend of discovered clusters. This is an empirical confirmation of effectiveness

[1] http://db.csail.mit.edu/ labdata/labdata.html

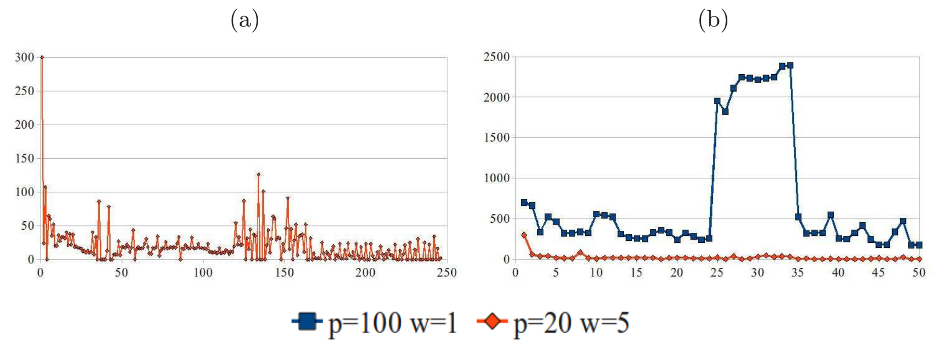

Fig. 7. Elapsed time (ms) with (a) $p = 20$ $w = 5$ over the sliding windows, (b) $p = 20$ $w = 5$ vs. $p = 100$ $w = 1$ over windows which cover same data portion. $\theta = 1$ $\epsilon = 1$.

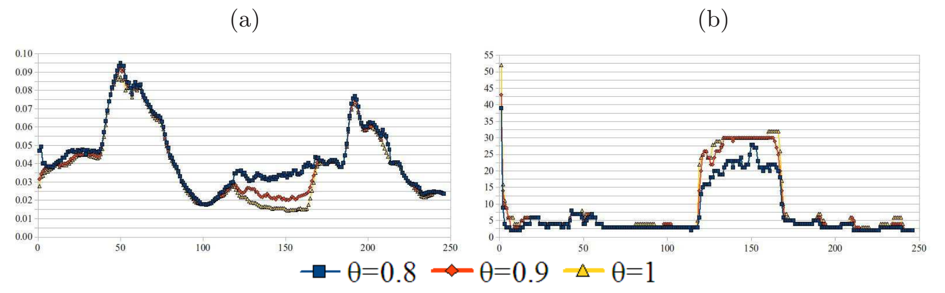

Fig. 8. (a) MAPE and (b) number of clusters with $p = 20$ and $w = 5$.

of sliding window model to improve efficiency of mining trend-based patterns in a stream. Further experiments have been performed to evaluate dependence of TRUST from θ. The number of clusters as well as the MAPE for consecutive sliding-windows are plotted in Figure 8.a-b by varying θ between 0.8, 0.9 and 1. The number of clusters shows that TRUST is able to group sources over windows thus summarizing windowed data by means of the polylines which are associated to the clusters (the number of discovered clusters is generally less than 10 with a pick to 32 in the central plotted windows). As expected, we observe that by relaxing the threshold of the trend continuity over slide (θ), the number of clusters decreases and, consequently, also the accuracy of fitting decreases. The fitting capability is always good enough with an error that is at worst 0.1. The issue is to find the trade-off between summarization degree (number of clusters) and fitting accuracy (MAPE). In this work, we do not consider $\epsilon < 1$ since we intend to detect window-time continuous trend clusters.

South American Climate data stream. South American climate data stream[2] contains monthly-mean air temperature values recorded between 1960 and 1990 over a 0.5 degree by 0.5 degree of latitude/longitude grid of South America, where

[2] http://climate.geog.udel.edu/~climate/html_pages/archive.html

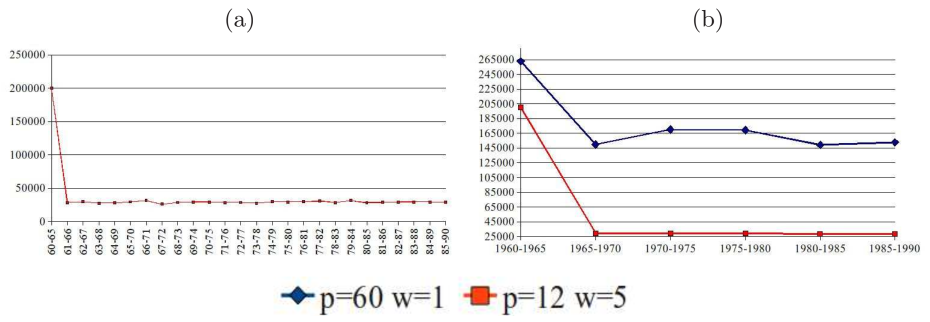

Fig. 9. Elapsed time (ms) with (a) $p = 12$ $w = 5$ over the sliding windows, (b) $p = 12$ $w = 5$ vs. $p = 60$ $w = 1$ over windows which cover same data portion. $\theta = 1$ $\epsilon = 1$.

the grid nodes are centered on 0.25 degree for a total of 6447 node sources. A source is close to the neighbor sources which are located in the grid cells around the source. Experiments are run with $\delta = 10\%[\beta - \alpha]$. We use the box plot to group streamed temperature values and to determine $\alpha = 6.925$, $\beta = 37.125$ and, hence, $\delta = 3.02$.

The elapsed time of the learning process is reported in Figure 9 for two parameter settings: a) $p = 12$ (one year) and $w = 5$; b) $p = 60$ and $w = 1$. Windows cover the same portion of data stream. Also in this stream, we observe that TRUST capitalizes on the slide-level discovery to improve efficiency of learning. Although the number of sources is significantly high, learning is efficiently enough to run on-line. Indeed, after the first window, where five slides have to be processed before being able to output clusters over the window, the elapsed time to learn each single slide and derive the clusters over the window is less than 30 secs.

We also evaluate the effectiveness of trend clusters discovered by varying θ between 0.8, 0.9 and 1. The number of clusters as well as the MAPE values for consecutive sliding-windows are plotted in Figure 10.a-b. We observe that the number of clusters is greatly lower than 6447 (at worst 4 clusters over each window), thus TRUST is effective in summarizing data. Additionally, MAPE value is low (less than 0.06 independently on θ), this confirms that also in this stream the summarization by means of the discovered trend clusters is accurate in fitting the real windowed data. A final consideration concerns θ, by relaxing which, the number of clusters decreases only over few windows and in these cases also the accuracy of fitting slightly decreases.

Some conclusions can be drawn from this empirical study. TRUST is effective in discovering accurate trend-based clusters as we proved with the analysis of MAPE. TRUST is scalable as we proved with the analysis of data streams generated by both a low number of sources (synthetic data and Berkley Lab data) and an high number of sources (South American Climate data). Additionally, TRUST is able to operate in real world situations and cope with possible variation in the sources configurations over consecutive layers (Berkley Lab data).

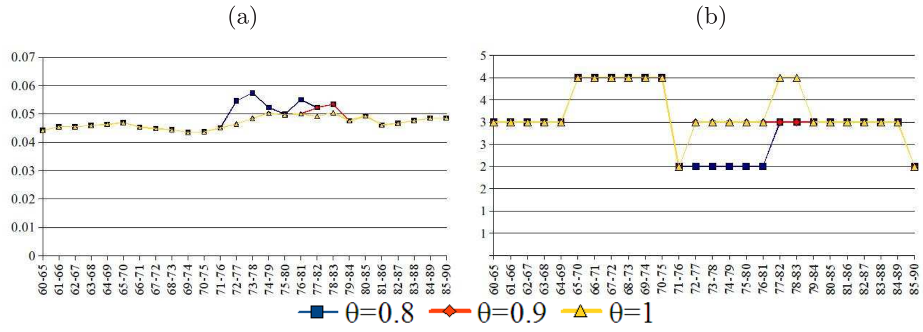

Fig. 10. (a) MAPE and (b) number of clusters ($p = 12$ $w = 5$).

Final considerations concern the utility of patterns that TRUST discovers. Our basic consideration is that trends provide useful information in spatially distributed data stream, where knowing how spatially clustered values vary in time can help drawing useful conclusions. For instance, in weather monitoring, it is essential to know how temperatures evolve (increasing or decreasing) over regions of the Earth and how the shape of these regions changes in time. Additionally, trend-based representation of spatially distributed data stream is considered close to the human intuition. Finally, trend clusters can be used for summarizing spatio-temporal data and subsequent storing in data warehouses.

6 Conclusions

The paper presents TRUST an algorithm to retrieve groups of spatially continuous geo-referenced data which vary according to a similar trend polyline in the recent window past. As future work we plan to investigate how the algorithm depends on the order of analyzing the geo-referenced streamed values and we intend to study criteria to suggest the best order of evaluation.

Acknowledgments

This work is supported by the Strategic Project PS121: "Telecommunication Facilities and Wireless Sensor Networks in Emergency Management".

References

1. C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for clustering evolving data streams. In *VLDB 2003*, pages 81–92, 2003.
2. C. Armenakis. Estimation and organization of spatio-temporal data. In *GIS*, 1992.
3. B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *PODS 2002*, pages 1–16. ACM, 2002.

4. B. Babcock, M. Datar, R. Motwani, and L. O'Callaghan. Maintaining variance and k-medians over data stream windows. In *PODS 2003*, pages 234–243. ACM, 2003.
5. K. Bhaduri, K. D. K. Sivakumar, H. Kargupta, R. Wolff, and R. Chen. Algorithms for distributed data mining. In *Data Streams: Models and Algorithms (book chapter)*, volume 31, pages 309–334. Springer-Verlag, 2007.
6. F. Cao, M. Ester, W. Qian, and A. Zhou. Density-based clustering over an evolving data stream with noise. In J. Ghosh, D. Lambert, D. B. Skillicorn, and J. Srivastava, editors, *SIAM SDM 2006*, 2006.
7. W. Chang, D. Zeng, and H. Chen. A stack-based prospective spatio-temporal data analysis approach. *Decis. Support Syst.*, 45(4):697–713, 2008.
8. S. Gaffney and P. Smyth. Trajectory clustering with mixtures of regression models. In *KDD 1999*, pages 63–72. ACM, 1999.
9. V. Ganti, J. Gehrke, and R. Ramakrishnan. Mining data streams under block evolution. *SIGKDD Explorations*, 3(2):1–10, 2002.
10. S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams. In *FOCS*, pages 359–366, 2000.
11. R. Hyndman and A. B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688, 2006.
12. P. Kalnis, N. Mamoulis, and S. Bakiras. On discovering moving clusters in spatio-temporal data. In C. Bauzer Medeiros et al., editor, *SSTD 2005*, volume 3633 of *Lecture Notes in Computer Science*, pages 364–381. Springer-Verlag, 2005.
13. J.-G. Lee, J. Han, and K.-Y. Whang. Trajectory clustering: a partition-and-group framework. In *SIGMOD 2007*, pages 593–604. ACM, 2007.
14. Y. Li, J. Han, and J. Yang. Clustering moving objects. In *KDD 2004*, pages 617–622, New York, NY, USA, 2004. ACM.
15. D. Malerba, A. Appice, A. Varlaro, and A. Lanza. Spatial clustering of structured objects. In S. Kramer and B. Pfahringer, editors, *ILP 2005*, volume 3625 of *Lecture Notes in Computer Science*, pages 227–245. Springer-Verlag, 2005.
16. M. May, B. Berendt, A. Cornujols, J. Gama, F. Giannotti, A. Hotho, D. Malerba, E. Menesalvas, K. Morik, R. Pedersen, L. Saitta, Y. Saygin, A. Schuster, and K. Vanhoof. Research challenges in ubiquitous knowledge discovery. In *Next Generation of Data Mining*. Chapman and Hall/CRC, 1 edition, 2008.
17. R. Munro and S. Chawla. An integrated approach to mining data streams. In *Technical Report, University of Sydney. School of Information Technologies*, 2004.
18. M. Nanni and D. Pedreschi. Time-focused clustering of trajectories of moving objects. *J. Intell. Inf. Syst.*, 27(3):267–289, 2006.
19. S. Shekhar and S. Chawla. *Spatial databases: A tour*. Prentice Hall, 2003.
20. M. Vlachos, D. Gunopoulos, and G. Kollios. Discovering similar multidimensional trajectories. In *ICDE 2002*, page 673. IEEE Computer Society, 2002.
21. T. Zhang, R. Ramakrishnan, and M. Livny. Birch: an efficient data clustering method for very large databases. *SIGMOD Rec.*, 25(2):103–114, 1996.