A Framework for Trajectory Data Preprocessing for Data Mining

Luis Otavio Alvares, Gabriel Oliveira, Vania Bogorny

Instituto de Informatica – Universidade Federal do Rio Grande do Sul Porto Alegre – Brazil {alvares,gpaoliveira,vbogorny,}@inf.ufrgs.br

Abstract

Trajectory data play a fundamental role to an increasing number of applications, such as traffic control, transportation management, animal migration, and tourism. These data are normally available as sample points. However, for many applications, meaningful patterns cannot be extracted from sample points without considering the background geographic information. In this paper we present a framework to preprocess trajectories for semantic data analysis and data mining. This framework provides two different methods to add semantic geographic information to the important parts of trajectories from an application point of view.

1. Introduction

The increasing use of GPS devices to capture the position of moving objects demands tools for the efficient analysis of large amounts of data referenced in space and time. Current analysis over trajectories of moving objects have basically to be performed manually. Another problem is that most techniques for the analysis of this kind of data and more sophisticated approaches as data mining algorithms consider only the raw trajectories, that are generated as pure (x,y,t) coordinates. In the last years, some works have been developed for trajectory data analysis, such as [5], in particular for discovering dense regions or similar trajectories. However, these approaches consider only the geometric properties of trajectories, what is very limited for many real applications.

GPS and other electronic devices that capture moving object trajectories do not collect the background geographic information. We claim that for several real applications there is a need to preprocess trajectories to add additional information that gives to trajectories more meaningful characteristics. Indeed, this should be the first step, before any trajectory data analysis. We claim that the first additional information to be considered, is the geographic context where trajectories are captured.

Figure 1 shows an example where we can observe the

necessity of extra information to understand trajectories. Figure 1 (left) shows an example of a geometric trajectory, in which the objects move to the same region at a certain time. Considering a pure geometric approach where only the trajectory points themselves are used for mining it could only be discovered that the four trajectories meet in a certain region, or the trajectories are dense in this region at a certain time. In Figure 1 (right), considering the background geographic knowledge, the moving objects go from different hotels (H) to meet the Eiffel Tower at a certain time. From these trajectories with some semantics, the moving pattern from Hotel to Eiffel Tower could be discovered. In this example, it is clear that the origin of the trajectories is in sparse locations that have the same semantics (it is a hotel). A pure geometric trajectory data mining algorithm would not be able to discover such semantic pattern.

In [2] we presented a *spatial* framework to automatically preprocess geographic data for data mining. In this paper we present an intelligent *spatio-temporal* framework to preprocess trajectories, in order to transform trajectory sample points in a higher level of abstraction, adding geographic semantics to trajectories.

The main contribution of this work is a framework to allow a user to both analyze and mine trajectories in a high level of abstraction, considering the needs of the application. This framework implements two different methods to add semantics to raw trajectories: one is based on the intersection of trajectories with places relevant to the application and the other is based on the speed of the trajectory. Furthermore, different classical data mining algorithms can be applied in the data mining step.

The remaining of this paper is structured as follows: Section 2 introduces some concepts of semantic trajectories, Section 3 presents the proposed framework for trajectory data analysis and mining, Section 4 presents an implementation of the framework and some experiments, and Section 5 concludes the paper and suggests directions of future works.

2. Basic Concepts

Recently Spaccapietra [8] introduced the first conceptual model for trajectory data, with two key concepts: stops and



Figure 1. (left) Geometric (raw) trajectories and (right) semantic trajectories

moves. *Stops* are important places of the trajectory from an application point of view, where the moving object has stayed for a minimal amount of time. *Moves* are subtrajectories between two consecutive stops.

To better understand what stops and moves are, we present one formal model where geographic object types are defined a priori by the user as the important places of the trajectory. This model has been introduced in [1] for querying trajectories, but it is not the only way to formally define stops and moves. It will be briefly presented in the following subsections.

2.1. Trajectory Samples and Candidate Stops

Trajectory data are normally available as sample points.

Definition 1 A sample trajectory is a list of space-time points $\langle p_0, p_1, \ldots, p_N \rangle$, where $p_i = (x_i, y_i, t_i)$ and $x_i, y_i, t_i \in \mathbf{R}$ for $i = 0, \ldots, N$ and $t_0 < t_1 < \cdots < t_N$.

To transform trajectory sample points into *stops* and *moves* it is necessary to provide the important places of the trajectory which are relevant for the application. These places correspond to different spatial feature types (spatial object types). For each relevant spatial feature type that is important for the application, a minimal amount of time is necessary, such that a trajectory should continuously intersect this feature in order to be considered a stop. This pair is called candidate stop.

Definition 2 A candidate stop C is a tuple (R_C, Δ_C) , where R_C is a polygon in \mathbb{R}^2 and Δ_C is a strictly positive real number. The set R_C is called the geometry of the candidate stop and Δ_C is called its minimum time duration. An application \mathcal{A} is a finite set $\{C_1 = (R_{C_1}, \Delta_{C_1}), \ldots, C_N = (R_{C_N}, \Delta_{C_N})\}$ of candidate stops with mutually nonoverlapping geometries R_{C_1}, \ldots, R_{C_N} .

In case that a candidate stop is a point or a polyline, a polygonal buffer is generated around this object, and thus it is represented as a polygon in the application, in order to overcome spatial uncertainty.

2.2. Stops and Moves

Definition 3 Let T be a trajectory and let

$$\mathcal{A} = \{ C_1 = (R_{C_1}, \Delta_{C_1}), \dots, C_N = (R_{C_N}, \Delta_{C_N}) \}$$

be an application. Suppose we have a subtrajectory $\langle (x_i, y_i, t_i), (x_{i+1}, y_{i+1}, t_{i+1}), \ldots, (x_{i+\ell}, y_{i+\ell}, t_{i+\ell}) \rangle$ of T, where there is a (R_{C_k}, Δ_{C_k}) in \mathcal{A} such that $\forall j \in [i, i+\ell] :$ $(x_j, y_j) \in R_{C_k}$ and $|t_{i+\ell} - t_i| \geq \Delta_{C_k}$, and this subtrajectory is maximal (with respect to these two conditions), then we define the tuple $(R_{C_k}, t_i, t_{i+\ell})$ as a stop of T with respect to \mathcal{A} .

A move of T with respect to A is one of the following cases: (i) a maximal contiguous subtrajectory of T in between two temporally consecutive stops of T; (ii) a maximal contiguous subtrajectory of T in between the initial point of T and the first stop of T; (iii) a maximal contiguous subtrajectory of T in between the last stop of T and the last point of T; (iv) the trajectory T itself, if T has no stops.

When a move starts in a stop, it starts in the last point of the subtrajectory that intersects the stop. Analogously, if a move ends in a stop, it ends in the first point of the subtrajectory that intersects the stop.

It is important to notice that the place where a stop occurs is a spatial feature (relevant geographic object) which is intersected by a trajectory for the minimal amount of time. This spatial feature will enrich the trajectory with its spatial and non-spatial information. For instance, if a hotel is a stop, its geometry and the non-spatial attributes (e.g. name, stars, price) is information that can be further used for both querying and mining trajectories.

Definition 4 A Semantic Trajectory S is a finite sequence $\{I_1, I_2, ..., I_n\}$ where I_K is a stop or a move.

3. The proposed framework

Figure 2 shows an interoperable framework with support to the whole discovery process. It is composed of three abstraction levels: data repository, data preparation, and data mining.

At the bottom are the geographic data repositories, stored in GDBMS (geographic database management systems), constructed under OGC [6] specifications. On the top are the data mining toolkits or algorithms. In the center is the trajectory data preparation level which adds semantics to trajectories according to the application domain. In this level the data repositories are accessed through JDBC connections and data are retrieved, preprocessed, and transformed into the format required by the mining tool/algorithm.

There are three main modules to implement the tasks of trajectory data preparation for mining: Clean Trajectories, Add Semantics, and Transformation, which are described in the sequence.



Figure 2. The Semantic Trajectory Mining Framework

3.1. Clean trajectories

The Clean Trajectories module performs many verifications over the trajectory dataset in order to eliminate noise, what is very common in this kind of data, and assure that the trajectory dataset is in the format required by the Add Semantics module.

Some of the verifications include: i) the calculated speed between two consecutive points should not be greater than a specified threshold; ii) the trajectory points should be in a temporal order; iii) the trajectories should not have more than one point with the same timestamp; iv) each trajectory should have a given minimum number of points.

3.2. Add Semantics

To prepare trajectory data to data mining, the main step is to add semantics to these trajectories. We do that by using the concepts of stops and moves. Two algorithms have been developed so far. The first one, introduced in [1], considers the intersection of a trajectory with the user-specified relevant feature types for a minimal time duration (candidate stops), which we call IB-SMoT (Intersection-Based Stops and Moves of Trajectories).

In general words, the algorithm verifies for each point of a trajectory T if it intersects the geometry of a candidate stop R_C . In affirmative case, the algorithm looks if the duration of the intersection is at least equal to a given threshold Δ_C . If this is the case, the intersected candidate stop is considered as a stop, and this stop is recorded. If a point does not belong to a subtrajectory that intersects a candidate stop for Δ_C it will bee part of a move.

Figure 3 illustrates this method. In the example, there are four candidate stops with geometries R_{C_1} , R_{C_2} , R_{C_3} , and R_{C_4} . Let us consider a trajectory T represented by the space-time points sequence $\langle p_0, \ldots, p_{15} \rangle$ and t_0, \ldots, t_{15} are the time points of T. First, T is outside any candidate stop, so we start with a move. Then T enters R_{C_1} at point p_3 . Since the duration of staying inside R_{C_1} is long enough, (R_{C_1}, t_3, t_5) is the first stop of T, and $\langle p_0, \ldots, p_3 \rangle$



Figure 3. Example of a trajectory with four candidate stops and two stops

is its first move. Next, T enters R_{C_2} , but for a time interval shorter than Δ_{C_2} , so this is not a stop. We therefore have a move until T enters R_{C_3} , which fulfills the requests to be a stop, and so $(R_{C_3}, t_{13}, t_{15})$ is the second stop of T and $\langle p_5, \ldots, p_{13} \rangle$ is its second move.

The second algorithm is called CB-SMoT [7], and is a clustering method based on the variation of the speed of the trajectory. The intuition of this method is that the parts of a trajectory in which the speed is lower than in other parts of the same trajectory, correspond to interesting places. CB-SMoT is a two-step algorithm. In the first step, the slower parts of one single trajectory are identified, using a spatiotemporal clustering method that is a variation of the DB-SCAN [3] algorithm considering one-dimensional line (traiectories) and speed. In the second step, the algorithm identifies where these potential stops (clusters) are located, considering the candidate stops. In case that a potential stop does not intersect any of the given candidates, it still can be an interesting place. In order to provide this information to the user, the algorithm labels such places as *unknown stops*. Unknown stops are interesting because although they may not intersect any relevant spatial feature type given by the user, a pattern can be generated for unknown stops if several trajectories stay for a minimal amount of time at the same unknown stop. In this case, the user may investigate what this unknown stop is.

Figure 4 illustrates the method CB-SMoT. Considering the trajectory $T = \langle p_0, p_1, \ldots, p_n \rangle$ represented in Figure 4, the first step is to compute the clusters. Suppose that T has 4 potential stops, the clusters G_1, G_2, G_3 and G_4 , represented by ellipsis. In this example the user has specified 4 candidate stops, identified by the rectangles R_{C1}, R_{C2}, R_{C3} and R_{C4} . The cluster G_1 intersects the candidate stop R_{C1} for a time greater than Δ_{c1} , then the first stop of the trajectory is R_{C1} . The same occurs with the cluster G_2 , considering R_{C3} , which is the second stop of the trajectory. The clusters, G_3 and G_4 do not intersect any candidate stop. Therefore, G_3 and G_4 are unknown stops.

The two methods cover a relevant set of applications. IB-SMoT is interesting in applications where the speed is not important, like tourism and urban planning. In this kind of application, the presence or the absence of the moving object in relevant places is more important. However, in other



Figure 4. Example of a trajectory with 2 stops and 2 unknown stops

applications like traffic management, CB-SMoT, which is based on speed, would be more appropriate.

The output of the *Add Semantics* module are relations of stops and moves in the database. The schema of the stop relation has the following attributes:

```
STOP (Tid integer, Sid integer, SFTname varchar,
SFid integer, startT timestamp,
endT timestamp)
```

where:

where:

- *Tid*: is the trajectory identifier.
- Sid: is the stop identifier. It is an integer value starting from 1, in the same order as the stops occur in the trajectory. This attribute represents the sequence as stops occur in the trajectory.
- *SFTname*: is the name of the relevant spatial feature type (geographic database relation) where the moving object has stayed for the minimal amount of time.
- *SFid*: is the identification of the instance (e.g. Ibis) of the spatial feature type (e.g. Hotel) in which the moving object has stopped.
- *startT*: is the time in which the stop has started, i.e., the time that the object enters in a stop.
- *endT*: is the time in which the moving object leaves the stop.

In a relational model, the attributes *SFTname* and *SFid* are a foreign key to a geographic relation. Therefore, the stop relation significantly facilitates querying trajectories from a semantic point of view. Queries can be performed considering both spatial and non-spatial attributes of any spatial object that represents a stop. The relation of moves has the following schema, with four attributes more than the stop relation:

```
MOVE (Tid integer, Mid integer, SFTlname varchar,
SF1id integer, SFT2name varchar,
SF2id integer, startT timestamp,
endT timestamp, the_move multiline )
```

```
• Mid: is the identifier of the move in the trajectory. It starts with 1, in the same order as the moves occur in the trajectory.
```

- *SFT1name and SFT2name* : are the names of the spatial feature type in which the move respectively starts and finishes.
- *SF1id and SF2id*: are the identifier (feature instance) of the start and end stop of the move.
- *the_move*: is the set of points that corresponds to the spatial properties of a move.

3.3. Transformation

The Transformation module uses as input the tables of stops and moves in the database, generated by the *Add Semantics* module and generates an output file in the format required by a specific mining algorithm or tool. Although each tool can use a specific format, there are two main format types. One, the most used, can be seen as an horizontal type, where each line corresponds to one trajectory and each column corresponds to one stop or move. The other type is a vertical one, where each line corresponds to a stop or move of a trajectory. This second type is mostly used for sequential pattern mining.

Another key issue performed by the Transformation module is to generate the output file in the granularity level specified by the user. In fact, the stop and move table is generated in the lowest granularity level (instances of objects for the spatial dimension and timestamp for the time dimension). However, it is almost impossible to find patterns at this granularity level. It is very difficult to some events occur in the same second, for instance several trajectories arriving at home at exactly the same moment. To overcome this problem, in our framework the user can specify different granularity levels, for instance to consider intervals of one hour. This means that one event that occurs at 18:10PM will be considered at the same case as another occurred at 18:20PM. Depending on the application, the time granularity can be year, month, week, day, hour, etc. Analogously, the space granularity can change, including even the semantics of the object. For instance, in the example of Figure 1 the space granularity was the class of the object (Hotel), what will allow that a pattern from Hotel to Eiffel Tower could be discovered. In this case, Hotel is at the feature type granularity level and the Eiffel Tower at instance granularity level. If both were at feature type granularity level, the discovered pattern could be from Hotel to Touristic Place.

Furthermore, the user can specify what will be considered in the mining step: (i) only the space dimension; (ii) the space and the time of the beginning of the stop or move; (iii) the space and the time of the end of the stop or move; or (iv) the space and the time of beginning and ending of the stop or move.

4. Validation and experiments

The proposed framework was implemented in the java programming language in a module called STPM (Semantic Trajectory Preprocessing Module) and tested using Weka as data mining tool and PostGIS as data repository. Weka[4] is a free and open source non-spatial data mining toolkit with several data mining algorithms.

With the information specified by the user, the STPM module connects to the database through JDBC and can execute different tasks. Usually, the first one is to clean the trajectory dataset and put it in the format required by STPM. After that, the user can generate semantic trajectories. To do this, he should supply some information to the program like the method (IB_SMoT or CB_SMoT), the spatial feature types of interest (candidate stops) with the respective minimum time duration in order to be considered a stop, etc. Before mining, the last step is to generate an .arff file (the native Weka input format), which can be either in the horizontal or vertical type.

So far, we have tested the prototype with data stored in a Postgresql/Postgis database. We have performed some experiments with real trajectory data collected in the city of Rio de Janeiro. A first experiment was performed considering the districts of Rio de Janeiro and the trajectories. We used the IB-SMoT method and frequent pattern mining to find the districts crossed by a large number of trajectories (minsup=10%) considering time intervals (07:00-09:00, 09:01-12:00, 12:01-17:00, 17:01-20:00, other). Some frequent patterns found are:

{Barra[07:00-09:00], Joa[07:00-09:00], SaoConrado[07:00-09:00]} (s= 0.18) {Joa[17:01-20:00], SaoConrado[17:01-20:00]} (s= 0.2)

The first pattern expresses that 18% of the trajectories cross the districts Barra, Joa and SaoConrado between 7AM and 9AM. The second pattern means that 20% of the trajectories cross Joa and SaoConrado in the period between 5PM and 8PM.

In a second experiment we used the set of streets as background geographic information, and the CB-SMoT method to generate stops. We also used more refined time intervals. The objective was to investigate the streets and periods of slow traffic. An example of an association rule found in this experiment is:

$ElevadaDasBandeiras[18:01-18:30] \rightarrow$ AvenidaDasAmericas[18:31-19:00] (s= 0.05) (c=0.58)

This rule expresses that 58% of the trajectories with slow traffic at Elevada das Bandeiras between 6PM and 6:30PM, also have slow traffic at Avenida das Americas between 6:31PM and 7PM. This pattern of slow traffic occurs in 8% of the trajectories.

We can observe by the examples above that this framework facilitates the analysis of the obtained results from a user point of view. The output is in a high abstraction level, what we call semantic patterns, in opposition of pure geometric patterns generated by other approaches.

5. Conclusions and Future Work

Trajectory data are normally available as sample points, what makes their analysis in different application domains expensive from a computational point of view and quite complex from a user's perspective. A higher abstraction level considering semantics is needed.

This paper presented a framework to preprocess sample point trajectories for semantic trajectory data analysis and mining. The framework is application and domain independent.

As future ongoing work we are extending the framework in order to consider other methods to add semantics to trajectories.

Acknowledgements

This work was partially supported by the Brazilian agencies CAPES (Prodoc Program) and CNPq. We would like to thank the Traffic Engineering Company of Rio de Janeiro for the trajectory data.

References

- L. O. Alvares, V. Bogorny, B. Kuijpers, J. A. F. de Macedo, B. Moelans, and A. Vaisman. A model for enriching trajectories with semantic geographical information. In *ACM-GIS*, pages 162–169, New York, NY, USA, 2007. ACM Press.
- [2] V. Bogorny, P. M. Engel, and L. O. Alvares. Geoarm: an interoperable framework to improve geographic data preprocessing and spatial association rule mining. In K. Zhang, G. Spanoudakis, and G. Visaggio, editors, *SEKE*, pages 79– 84, 2006.
- [3] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In E. Simoudis, J. Han, and U. M. Fayyad, editors, *Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231. AAAI Press, 1996.
- [4] E. Frank, M. A. Hall, G. Holmes, R. Kirkby, B. Pfahringer, I. H. Witten, and L. Trigg. Weka - a machine learning workbench for data mining. In O. Maimon and L. Rokach, editors, *The Data Mining and Knowledge Discovery Handbook*, pages 1305–1314. Springer, 2005.
- [5] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory pattern mining. In P. Berkhin, R. Caruana, and X. Wu, editors, *KDD*, pages 330–339. ACM Press, 2007.
- [6] OGC. Opengis standards and specifications. Available at: http://http://www.opengeospatial.org/standards. Accessed in August 2008, 2008.
- [7] A. T. Palma, V. Bogorny, and L. O. Alvares. A clusteringbased approach for discovering interesting places in trajectories. In ACMSAC, pages 863–868, New York, NY, USA, 2008. ACM Press.
- [8] S. Spaccapietra, C. Parent, M. L. Damiani, J. A. de Macedo, F. Porto, and C. Vangenot. A conceptual view on trajectories. *Data and Knowledge Engineering*, 65(1):126–146, 2008.