Weka-GDPM – Integrating Classical Data Mining Toolkit to Geographic Information Systems

Vania Bogorny, Andrey Tietbohl Palma, Paulo Martins Engel, Luis Otavio Alvares

Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Av. Bento Gonçalves, 9500, Porto Alegre, RS, Brasil
{vbogorny,andrey,engel,alvares}@inf.ufrgs.br

Abstract. *Geographic data preprocessing is the most effort and time consuming step in spatial data mining. In order to facilitate geographic data preprocessing and increase the practice of spatial data mining, this paper presents Weka-GDPM, an interoperable module that supports automatic geographic data preprocessing for spatial data mining. GDPM is implemented into Weka, which is a free and open source classical data mining toolkit that has been widely used in academic institutions. GDPM follows the Open GIS specifications to support interoperability with Geographic Information Systems. It automatically generates data at two granularity levels without using prior knowledge and provides support for both distance and topological spatial relationships.*

1. Introduction

Large amounts of geographic data have been used more and more in many areas in different application domains such as urban planning, transportation, telecommunication, marketing, etc. These data are stored under Geographic Database Management Systems (GDBMS), and manipulated by Geographic Information Systems (GIS). The latter is the technology which provides a set of operations and functions for geographic data analysis. However, within the large amount of data stored in geographic databases there is implicit, non-trivial, and previously unknown knowledge that cannot be discovered by GIS. Specific techniques are necessary to find such knowledge, which is the objective of Knowledge Discovery in Databases (KDD).

KDD is an interactive process which according to [Fayyad 1996] consists of five main steps: *selection, preprocessing, transformation, data mining* and *evaluation/interpretation. Selection, preprocessing* and *transformation* are data preparation steps in which data are rearranged to the format required by data mining algorithms. For non-spatial databases it is stated that between 60 and 80 percent of the time and effort in the whole KDD process is required for data preparation [Addrians 1996]. For geographic databases this problem increases significantly because of the complexity of geographic data that must be considered. We have addressed this problem in [Bogorny 2005a] proposing an interoperable framework for geographic data preprocessing. In [Bogorny 2006a] we extended this framework using semantic knowledge to improve data preprocessing steps.

Different solutions for general knowledge discovery in geographic databases have been proposed in the literature, but only a few addressed aspects of data preparation. Most approaches propose data mining query languages or new spatial data mining algorithms. Han, for example, proposed a geo-mining query language named GMQL, implemented in the GeoMiner software prototype [Han 1997]. Malerba (2000) proposed an object-oriented data mining query language named SDMOQL, implemented in the INGENS software prototype. Ester (2000) defined a set of new operations to compute geographic neighbors. In these approaches it is expected that the GDBMS will implement the proposed languages and operations. However, most GDBMS follow the Structured Query Language (SQL), which

became the standard language to manipulate databases, and do neither implement data mining languages nor operations to either automate or semi-automate geographic data preprocessing.

Existing spatial data mining software prototypes such as GeoMiner [Han 1997] and INGENS [Malerba, 2000] are no longer available for practical purposes either inside or outside academic institutions. Other prototypes, such as Ares [Appice 2005], implement a spatial feature extractor, but compute *all* spatial relationships among all spatial feature types. This is a problem for mining real databases, since the spatial join computation performs a cartesian product among spatial feature types to compute spatial relationships. Indeed, Ares implements only the Spada algorithm for mining spatial association rules.

Aiming to make a contribution for the spatial data mining field and facilitate the practice of knowledge discovery in geographic databases, we propose GDPM (Geographic Data Preprocessing Module), which integrates geographic databases and the classical data mining toolkit Weka [Witten 2005]. Weka is a free and open source classical data mining toolkit which provides friendly graphical user interfaces to perform the whole KDD process, implements a variety of data mining algorithms, and has been largely used for mining non-spatial databases. The objective of GDPM is to automate geographic data preprocessing.

The main contribution of this work is for the data mining user, since no background knowledge is required to manipulate GDPM. Additional contributions include:

- Interoperability with any GDBMS constructed under OpenGIS Simple Features Implementation Specification [OGC 1999a].
- Different classical data mining algorithms can be applied in the data mining step.
- The user can easily define the relevant spatial feature types, the target feature type, different spatial relationships, as well as different granularity levels.
- The feature's spatial representation is independent of geometric type. While most spatial data mining algorithms are restricted to only points, our framework supports any geometric primitive.

The remaining of this paper is organized as follows: Section 2 presents background knowledge about geographic databases. Section 3 details the main aspects as well as the problems to be solved in geographic data preprocessing. Section 4 presents Weka-GDPM, and Section 5 concludes the paper and suggests directions of future works.

2. Background

Geographic databases (GDB) store real world entities, also called spatial features, located in a specific region [OGC 1999b]. Spatial features (e.g. Canada, France) belong to a feature type (e.g. country), and have both non-spatial attributes (e.g. name, population) and spatial attributes (geographic coordinates *x*,*y*). In GDB every different feature type is usually stored in a different relation, because most geographic databases follow the relational [Shekhar 2003] or object-relational approach. Figure 1 shows an example of geographic data stored in relational databases, where the spatial feature types street, water resource, and gas station are different relations with spatial (shape) and non-spatial attributes.

The spatial attributes of geographic object types, represented by *shape* in Figure 1, have intrinsic spatial relationships (e.g. close, far, contains, intersects). Because of these relationships real world entities can affect the behavior of other features in the neighborhood. This makes spatial relationships be the main characteristic of geographic data to be considered for data mining and knowledge discovery [Ester 2000]. It is also the main characteristic which differs geographic/spatial data mining from non-spatial data mining.

Spatial relationships are usually not explicitly stored in geographic databases, so they have to be computed with spatial operations. There are basically 3 spatial relationships to consider [Gutting 1994]: *distance, order,* and *topological*. *Distance* relationships are based on the Euclidean distance between two spatial features. *Direction* relationships deal with the order

as spatial features are located in space. *Topological* relationships characterize the type of intersection between two spatial features and can be classified in *Equal*, *Disjoint*, *Touches*, *Within*, *Overlaps*, *Crosses*, *Contains*, *Covers*, and *CoveredBy*. Mainly topological and distance relationships have been used in spatial data mining, and they will be the focus in this paper.

(a) Street

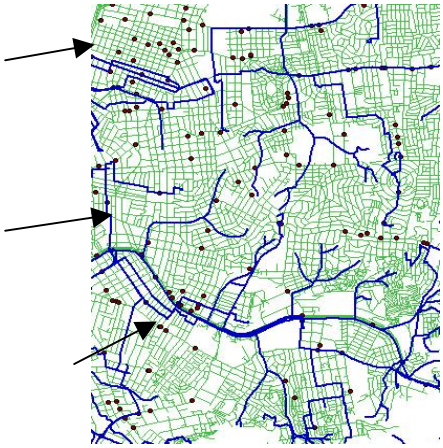
Gid	Name	Shape
1	Azenha	Multiline [$(x_1,y_1),(x_2,y_2)$,..]
2	Lajeado	Multiline [$(x_3,y_3),(x_4,y_4)$,..]

(b) WaterResource

Gid	Name	Shape
1	Jacui	Multiline [$(x_5,y_5),(x_2,y_2)$,..]
2	Guaiba	Multiline [$(x_6,y_6),(x_2,y_2)$,..]
3	Uruguai	Multiline [$(x_5,y_5),(x_7,y_7)$,..]

(c) GasStation

Gid	Name	VolDiesel	VolGas	Shape
1	BR	20000	85000	Point[(x_8,y_8)]
2	IPF	30000	95000	Point[(x_9,y_9)]
3	Elf	25000	120000	Point[(x_3,y_3)]



(d) GEOMETRY_COLUMNS

f_table_schema	f_table_name	f_geometry_column	type	SRID
Public	Street	Shape	Multiline	-1
Public	WaterResource	Shape	Multiline	-1
Public	GasStation	Shape	Point	-1

Figure 1 – Geographic data storage structure in OGC based GDBMS

GIS implement specific functions to manipulate geographic data. The OGC (Open GIS Consortium) is an organization dedicated to develop standards for geographic operations and geographic data integration, aiming to provide interoperability for GIS. Among many specifications established by the OGC, two are of fundamental importance for this work: operations to compute *spatial relationships* and the *database schema* metadata. Topological relationships are computed according to Egenhofer´s (1995) and Clementini´s (1993) approach.

The database schema metadata are stored in a database table named GEOMETRY_COLUMNS, which is automatically created in a GDBMS that follows OGC specifications. An example is illustrated in Figure 1 (d). This table consists of a row for each feature type in the geographic database with spatial attributes. It is instantiated automatically when geographic data are loaded to the database the first time, and stores all database characteristics, including the database schema name, all geographic table names (f_table_name), the name of the geometry column (f_geometry_column), and its type (type).

For automatic geographic data preprocessing the database table GEOMETRY_COLUMNS can be used to retrieve all spatial feature types stored in a GDB, as will be explained in the following section.

3. Geographic Data Preprocessing in GDPM

To prepare geographic databases for data mining the main steps include the definition of a target feature type on which discovery will be performed, a set of relevant feature types that may have an influence over the target feature type because of spatial relationships, the granularity level in which data will be represented and the spatial relationships to be computed. The target feature type and every relevant feature type are different database tables in a geographic database. The resultant single file that the data preparation step must generate contains one row for each instance (e.g. Porto Alegre) of the target feature type (e.g. city), and columns are predicates. The predicates are non-spatial attributes (e.g. population) of the target feature type and spatial relationships with the relevant feature types (e.g. contains_river).

Spatial relationships are computed with SQL queries which spatially join all instances *t* (e.g. Porto Alegre) of the target feature type *T* (e.g. city) and *all* instances *o* (e.g. Guaiba River) of every relevant feature type *O* (e.g. river) in a set of relevant feature types *S* (e.g. street, gasStation, river) that have any spatial relationship (e.g. touches, contains, close) with *T*. Being *T* a set of instances $T=\{t_1, t_2,\ldots,t_n\}$, $S = \{ O_1, O_i,\ldots, O_m\}$, and $O_i = \{ o_1, o_2,\ldots, o_q\}$, the extraction of spatial relationships implies the comparison of every instance of *T* with every instance of *O*, for all *O* in *S*.

According to the objective of the discovery, data can be represented at different granularity levels [Han 1995]. For example, having some regions in a metropolitan area high pollution incidence, it might be interesting to consider spatial predicates of factories in a more generalized level such as *contains(factory)*. In some specific cases, it might be interesting to consider spatial predicates of the different types of factories such as *contains(chemical_factory)*, *contains(metalurgical_factory)*. In very specific cases, it might be interesting to consider the instances of factories, such as *contains(chemical_factory_x)*, *contains(metallurgical_factory_y)*.

In this paper we consider two granularity levels detailed in [Bogorny 2005a]: feature instance and feature type. The former is a very low granularity in which the type of spatial features and their instance identifier is considered (e.g. river_1). The latter is a more generalized granularity level where only the feature type is considered (e.g. river). These two granularities can be automatically generated without using prior knowledge and without requiring background knowledge from the data mining user. In the following we explain how these granularity levels are generated for topological and distance relationships.

3.1 Topological Relationships

Topological relationships are mutually exclusive such that only one topological relationship holds between two spatial feature *instances* (e.g. Porto Alegre city and Canoas city). At the feature instance granularity level every instance of the target feature type may have only one topological relationship with an instance of a relevant feature type. Table 1 (left) illustrates an example of the spatial join computation where city 1 has the relationship *contains* with River_1, *crosses* with River_2, and *contains* with Slum_1. At the feature instance granularity level, when transforming the spatial join output (Table 1 left) into the Weka input format (Table 1 right), the relevant feature type name with the respective instance is transformed in an attribute name. The value of this attribute will be the respective topological relationship. For the relevant feature instances that have no relationship with an instance of the target feature (e.g. River_3 and city_1, River_1 and city_3), the attribute value is filled with "?", which is the symbol used by Weka to represent the absence of an attribute.

Table 1 - *Feature Instance* Granularity Level for topological relationships

TargetF_id (city)	RelevantF Instance	Relationship
1	River_1	Contains
1	River_2	Crosses
2	River_3	Contains
2	River_4	Crosses
3	River_2	Crosses
1	Slum_1	Contains
2	Slum_2	Contains

TargetF_id (city)	River_1	River_2	River_3	River_4	Slum_1	…
1	Contains	Crosses	?	?	Contains	
2	?	?	Contains	Crosses	?	
3	?	Crosses	?	?	?	

At the feature type granularity level, as shown in Table 2 (left), the relevant feature instance is not stored in the spatial join output. For example, city 1 has two topological relationships with the relevant feature type River, *contains* and *crosses* (with different rivers). At this granularity level, to preserve the type (semantic) of the topological relationship, we need to create a different attribute name (e.g. contains_river, crosses_river) for every relevant feature type with a different topological relationship with the target feature. Indeed, it is difficult to specify dominance between topological relationships in order to define the strongest. As this

problem has not been addressed in the literature so far, we propose to preserve the relationship type by concatenating it to the feature type, while the attribute value receives the string "yes" when the relationship holds and " ?" if there is no topological relationship, as shown in Table 2(right).

Table 2 - *Feature Type* Granularity Level for topological relationships

TargetF_id (city)	RelevantF Type	Relationship
1	River	Contains
1	River	Crosses
2	River	Contains
2	River	Crosses
3	River	Crosses
1	Slum	Contains
2	Slum	Contains

TargetF_id (city)	Contains_River	Crosses_River	Contains_Slum	…
1	Yes	Yes	Yes	
2	Yes	Yes	Yes	
3	?	Yes	?	

According to the objective of the discovery, the use of topological relationships can lose interesting information. To understand this problem, let us consider the geographic map shown in Figure 2, where the small polygons are slums, large polygons are districts, and black lines are water bodies of the city of Porto Alegre. Let us suppose that district is the target feature type and slums and water bodies are the relevant feature types.

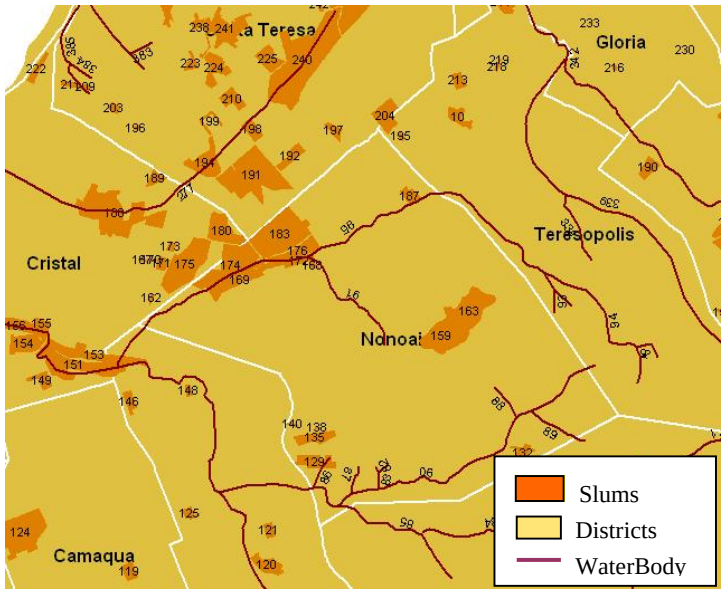


Figure 2 - Partial map of the Porto Alegre city representing districts, slums, and water bodies

In Figure 2 we can observe the different topological relationships that districts may have with both slums and water bodies. The district *Nonoai*, for example, "*contains*" slums (e.g. 159 and 183), "*touches*" slums (e.g. 180), and "*overlaps*" slums (e.g. 174). Different topological relationships may not generate patterns when mining data at the feature *instance* granularity level if thresholds such as minimum support, for example, for mining spatial association rules are not very low. For example, the predicate *touches(slum_183)* for district Santa Tereza is different from the predicate *contains(slum_183)* for the district Nonoai. The same occurs for slum_180 which is *within* district Cristal and *touched* by district Nonoai. When the objective is to investigate high criminal incidence, for example, and either slum 180 or 183 are responsible for high criminal incidence in districts Nonoai and Cristal, this might not be discovered by data mining algorithms.

To solve this kind of problem we suggest to consider general topological relationships *intersects* and *non-intersects*. When different instances of the target feature (e.g. district Cristal and district Nonoai) have topological relationships with the same instance of a relevant feature

type (e.g.*slum_180*) a predicate *intersects(slum_180)* is generated. Because of space limitations we do not show an example of this transformation process, but details can be found in (Bogorny, 2006b).

3.2 Distance Relationships

Distance relationships are computed according to the distance parameters provided by the user. If only one distance parameter (*dist1*), is provided neighborhoods are considered *very close* if their distance from the target feature is less or equal to *dist1*. When two distance measures are informed (*dist1* and *dist2*), than neighborhoods are considered *very close* if their distance from the target feature is less or equal to *dist1,* and *close* if their distance is between *dist1* and *dist2*, as shown in the example in Table 3 (left) for the feature instance (above) and feature type (below) granularity level.

Table 3 – *Feature Instance* and *Feature Type* Granularity for distance relationships

TargetF_id (city)	RelevantF Instance	Relationship
1	River_1	VeryClose
1	River_2	Close
2	River_3	Close
2	River_4	Close
3	River_2	VeryClose
1	Slum_1	Close
2	Slum_2	VeryClose

TargetF_id (city)	River_1	River_2	River_3	River_4	…
1	VeryClose	Close	?	?	
2	?	?	Close	Close	
3	?	VeryClose	?	?	

TargetF_id (city)	RelevantF Type	Relationship
1	River	VeryClose
1	River	Close
2	River	Close
3	River	VeryClose
1	Slum	Close
2	Slum	VeryClose

TargetF_id (city)	VeryClose_River	Close_River	Close_Slum	…
1	Yes	Yes	Yes	
2	?	Yes	?	
3	Yes	?	?	

The relationship *far* is not considered because experiments showed the generation of an enormous amount of non-interesting patterns. For example, the city 1 is very close to river 1 and close to river 2, but far from all other rivers. For distance relationships we can say that *close* is dominant over *far,* because all spatial objects that are *not close,* will be far. In case someone would like to consider things that are *far,* the value of the distance metric *dist2* can be increased in order to cover *far* spatial objects.

4. Weka and GDPM

Weka is a free and open source non-spatial data mining toolkit developed in Java. It has a non-spatial data preprocessing module named weka.Explorer in which it is possible to establish a database connection, open a web site, or an *arff* (input text file in the format required by Weka) file. The module GDPM is fully integrated into Weka in order to automatically access and preprocess geographic databases. (see Bogorny 2006b for details).

In order to support GDPM we extended the Weka database connection interface, shown in Figure 3(left). We added the button *GeographicData,* which calls the GDPM module, shown in Figure 3(right). As can be observed in Figure 3(right), the user provides the database schema and GDPM loads all geographic database tables (from the table geometry_columns) to the box Target Feature and Relevant Features. This allows the user to choose only the spatial feature types of interest, and not the whole geographic database. GDPM automatically generates data at two granularity levels for distance, topological, and high level topological relationships (intersects), without any concept hierarchy, as have been explained in the previous section.

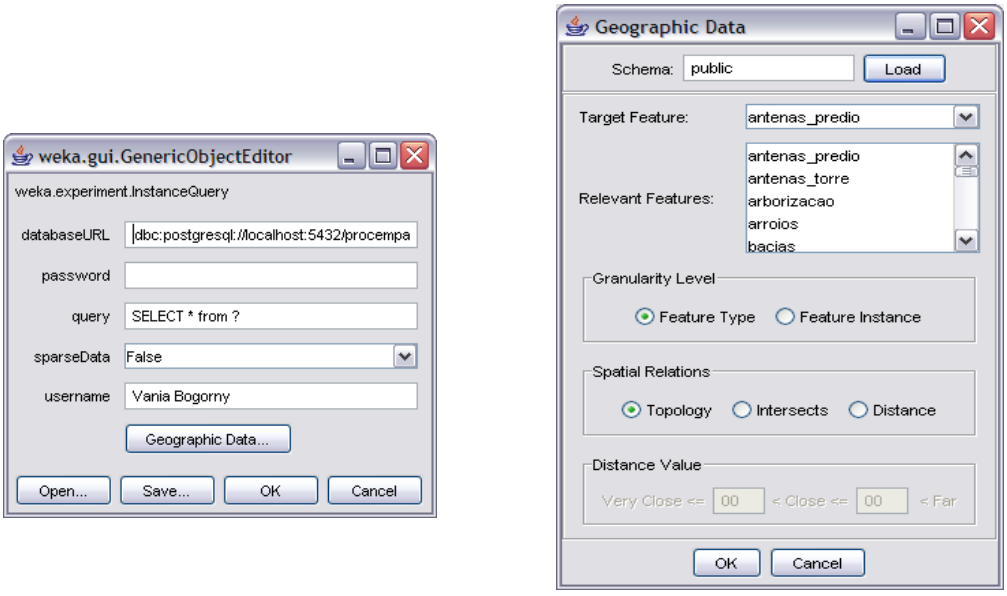


Figure 3 – Geographic data preprocessing interface

The spatial join step is performed among the target feature type and all selected relevant feature types. This is performed into the geographic database with the spatial operations implemented by the GDBMS, which follows the OGC approach. The result is stored in a temporary database table called *<target feature type name>_temp* (e.g. city_temp). This table contains the attributes *gid_target_feature_type*, *relevantF* (which is the name of the relevant feature type), and *relationship*, which were shown in section 3. The attribute *gid* (geographic identifier) is also standardized by the OGC and generated for all database tables.

The transformation retrieves the non-spatial attributes of the target feature type and the spatial predicates generated by the spatial join step. Transformation is performed in memory and generates as an output an *arff* file. As the *arff* file may contain a large number of attributes, mainly when mining data at the feature instance granularity level, the transformation step cannot store the result in a database table, since most GDBMS have a limited maximal number of columns.

GDPM has been tested with real GDB stored in PostGIS, which follows the OGC specifications. Experiments with different data mining techniques including classification and association rules were performed to evaluate and validate the data preprocessing method.

5. Conclusions and Future Works

This paper addressed the problem of geographic data preprocessing for spatial data mining. The main contribution of this work is for the data mining user. A free and open source spatial data mining toolkit that supports automatic geographic data preprocessing will facilitate and increase the practice of spatial data mining, since we are submitting GDPM to the Weka team in order to include it in the next Weka release.

A problem that is well known is the large amount of patterns generated by many data mining techniques. In geographic databases this problem increases because many discovered patterns are well known, because of the natural dependences of geographic data. A large number of such dependences can be automatically eliminated in geographic data preprocessing steps using prior knowledge [Bogorny 2006a]. The future ongoing work is to implement into GDPM the definition and automatic elimination of well known geographic dependences between the target feature type and relevant feature types.

ACKNOWLEDGMENT

Our thanks for both CAPES and CNPQ which partially provided the financial support for this research.

6. References

Adriaans, P. and Zantinge, D. “Data mining”. Addison Wesley Longman, Harlow, England.1996.

Appice, M., Berardi, M., Ceci, M. and Malerba, D. “Mining and Filtering Multi-level Spatial Association Rules with ARES”. In: Foundations Of 15th International Symposium Of Intelligent Systems, ISMIS, 3488., 2005, New York, Proceedings… [S.l], Springer LNCS, 2005,pp.342-353.

Bogorny, V., Engel, P.M. and Alvares, L.O. (2005a) “A Reuse-Based Spatial Data Preparation Framework for Data Mining”. In Proc of the 17 th International Conference on Software Engineering and Knowledge Engineering, (SEKE'05).Taiwan, China, pp. 649-652.

Bogorny, V., Engel, P. M. and Alvares, L.O (2006a) “GeoARM – an interoperable framework to improve geographic data preprocessing and spatial association rule mining”. In: Proc of the 18th International Conference on Software Engineering and Knowledge Engineering (SEKE’2006), San Francisco, California, pp. 79-84.

Bogorny, V., Palma, A. and Alvares, L.O. (2006b) “Extending the Weka Data Mining Toolkit to support Geographic Data Preprocessing”. Instituto de Informatica- UFRGS, Porto Alegre, Technical Report – RP 354.

Clementini, E., Di Felice, P. and Van Ostern, P. (1993). “A small set of formal topological relationships for end-user interaction”. In: ABEL, D; OOI, B.C. (Eds.). Advances in Spatial Databases. Springer-Verlag, 1993. pp. 277-295.

Ester, M., Frommelt, A., Kriegel, H.-P., and Sander, J. “Spatial Data Mining: Database Primitives, Algorithms and Efficient DBMS Support”. Journal of Data Mining And Knowledge Discovery, 4, 2-3(Jul. 2000), 193-216.

Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. (1996). “From data mining to discovery knowledge in databases”. *AI Magazine*, 3(17): 37-54.

Egenhofer, M. and Franzosa, R. (1995). “On the equivalence of topological relations”. International Journal of Geographical Information Systems, 9(2) 133-152.

Gutting, R. H. (1994). “An Introduction to Spatial Database Systems”. The International Journal on Very Large Data Bases, V3 (4), (October), pp. 357 - 399.

Han, J. (1995). “Mining Knowledge at Multiple Concept Level”. In: Proceedings of the 4th International Conference on Information and Knowledge Management (CIKM'95), Baltimore, Maryland, Nov. 1995, pp. 19-24.

Han, J., Koperski, K. and Stefanvic, N. (1997) “GeoMiner: a system prototype for geographic data mining”. In Proceedings of the ACM-SIGMOD international conference on Management Of Data (SIGMOD’97) (May 13-15,1997). ACM Press, Tucson, AR, p. 553-556.

Malerba, D. et al. (2000). “Discovering geographic knowledge: the INGENS system”. In Foundations of Intelligent Systems, 12th International Symposium, (ISMIS), Lecture Notes in Artificial Intelligence, 1932, 40-48, Springer, Berlin, Germany.

OGC (1999a). “OpenGIS simple features specification for SQL”. http://www.opengeogeographic.org/docs/99-054.pdf, August 2005.

OGC (1999b). “Topic 5, the OpenGIS abstract specification – OpenGIS features – Version 4”. http://www.OpenGIS.org/techno/specs.htm. August 2005.

Shekhar, S. and Chawla, S. “Spatial databases: a tour”. Prentice Hall, Upper Saddle River, NJ, 2003.

Witten, I. and Frank, E. “Data Mining: Practical machine learning tools and techniques”, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.