An Efficient Two-Step Method for Classification of Spatial Data

Krzysztof Koperski Jiawei Han Nebojsa Stefanovic

{koperski, han, nstefano}@cs.sfu.ca
School of Computing Science
Simon Fraser University
Burnaby, B.C., Canada V5A 1S6
ph. (604) 291-4411 fax (604) 291-3045

Abstract

Spatial data mining, i.e., discovery of interesting, implicit knowledge in spatial databases, is a highly demanding field because very large amounts of spatial data have been collected in various applications, ranging from remote sensing, to geographical information systems (GIS), computer cartography, environmental assessment and planning, etc. In this paper, an efficient method for building decision trees for the classification of objects stored in geographic information databases is proposed and studied. Our approach to spatial classification is based on both (1) non-spatial properties of the classified objects and (2) attributes, predicates and functions describing spatial relations between classified objects and other features located in the spatial proximity of the classified objects. Several optimization techniques are explored, including a two-step spatial computation technique, use of spatial-join indices, etc. We implemented the algorithm and conducted experiments that showed the effectiveness of the proposed method.

Keywords: Spatial Data Mining, Spatial Decision Support Systems, Decision Trees.

1 Introduction

The study and the development of data mining algorithms for spatial databases [14] is motivated by the large amount of data collected through remote sensing, medical equipment, and other methods. Moreover, the geocoding of consumer addresses in combination with large amount of recorded sales transactions creates very large spatially related databases. This increasing amount of data and demand for decision support systems has created a new area of research, spatial data mining, which is a subfield of data mining that deals with the *extraction of implicit knowledge, spatial relationships, or other interesting patterns not explicitly stored in spatial databases* [13]. It combines research in statistics, machine learning, spatial reasoning and spatial databases. In the last few years algorithms for spatial associations [13], clustering [4], generalized spatial descriptions [15], characteristics of spatial clusters [18] and others were analyzed. One of the areas which has not brought much attention is the classification of spatial objects.

Classification is a data mining technique where the data stored in a database is analyzed in order to find rules that describe the partition of the database into a given set of classes. Each object in a database (in relational databases tuples are treated as objects) is assumed to belong to a predefined class, as it is determined by one of the attributes, called the *class label attribute*. A number of classification methods were proposed by statistics and machine learning researchers [7, 20, 21]. The most common classification method constructs *decision trees*. Such method employs a top-down, divide-and-conquer strategy that partitions the set of given objects into smaller subsets where the *leaf* nodes are associated mostly with a single class. Most of the classification methods consider only relational data. Geographic data consists of spatial objects and non-spatial descriptions of these objects. Non-spatial descriptions of spatial objects can be stored in a relational database where one attribute is a pointer to the spatial description of the object [1]. In the process of spatial classification one wants to find rules that partition a set of classified objects into a number of classes using not only non-spatial properties of the classified objects, but also spatial relations of the classified objects to other objects in the database.

In this paper we address issues regarding classification of spatial data and concentrate on building decision trees for the classification of such data. Furthermore, we analyze the problem of classification of spatial objects in relevance to thematic maps and spatial relations to other objects in the database. Finally, we propose an algorithm that handles large amount of irrelevant spatial relations and we present experimental results of spatial classification process for both synthetic and real datasets.

The rest of the paper is organized as follows. In Section 2 we describe relevant work, the next section presents the in-depth analysis of the problem of spatial classification. Algorithm for building decision trees from spatial data is shown in Section 4. Experimental analysis is presented in Section 5 and the paper ends with the discussion of future work in Section 6.

2 Relevant Work

Classification of spatial data has been analyzed by some researchers. Fayyad *et. al.* [6] used decision tree methods to classify images of stellar objects to detect stars and galaxies. About 3 TB of sky images were analyzed. Data images were preprocessed by low-level image processing system FOCAS, which selected objects and produced basic attributes like: magnitudes, areas, intensity, image moments, ellipticity, orientation, etc. Objects in the training dataset were classified by astronomers. Based on this classification, about ten training sets for decision tree algorithm were constructed. From the decision trees obtained by the learning algorithm, a minimal set of robust, general and correct rules was found. The proposed method deals with image databases and is tailored for the astronomical application. Thus, it is not suitable for the analysis of vector data format, often used in Geographic Information Systems.

A method for classification of spatial objects was proposed by Ester *et. al.* [5]. The proposed algorithm is based on ID3 algorithm [20] and it uses the concept of neighborhood graphs. It considers not only non-spatial properties of the classified objects, but also non-spatial properties of neighboring objects. Objects are treated as neighbors if they satisfy neighborhood relation which may be one of the following:

- Topological-relations, like *meet, intersect.*
- Metric-relations, like *close_to.*
- Direction-relations, like *south, west.*

For example, cities O_c may be classified in relevance to the economic power of the regions that are neighbors of cities O_c. Since the influence of neighboring objects decreases with increasing distance the authors limit the length of the neighborhood paths. This method does not analyze aggregate values of non-spatial attributes for the neighboring objects. For example, if a city is close to three regions with medium population, such a city may have similar properties as a city close to a single region with large population. The algorithm also does not perform relevance analysis and thus, it may produce overspecialized, poor quality trees. Finally, the algorithm does not take into account concept hierarchies that may exist for the non-spatial and spatial attribute values.

Ng and Yu [18] described a method for the extraction of strong, common and discriminating characteristics of clusters based on thematic maps. The authors proposed measures for interest/utility values of the characteristics of clusters. In the search for common characteristics of the clusters the algorithm selects themes whose values are the most similar for a number of clusters. In the process of finding discriminating characteristics of the clusters the algorithm selects themes that discriminate between two sets of clusters in the best way. Unfortunately, only the properties of thematic maps are analyzed and this approach is not extended toward the construction of decision trees.

3 Problem Description

The goal of spatial classification process is to find rules that divide set of classified objects O_c into a number of groups, where objects in each group belong mostly to a single class. Spatial objects O_c may be characterized by different types of information. We classify such information into the following categories:

- Non-spatial attributes of objects (including both classified objects O_c and other objects used for description), like *the number of salespersons in a store.*
- Spatially related attributes with non-spatial values, like *population living within 1km from a store.*
- Spatial predicates, like *distance_less_than_10km(X, sea).*
- Spatial functions, like *driving_distance(X, beach).*

Each of these categories may be used to extract values both for *class label attribute* (e.g., attribute dividing data into classes) and *predicting attributes* (e.g., attributes on whose values decision tree is branched). We can use aggregate values for some of these attributes. For example, calculations of the size of the population living within 1 kilometer from the store may involve finding the sum of populations of all census blocks intersecting one kilometer buffer from the store. Moreover, there may exist concept hierarchies for these attributes which can result in building simpler decision trees.

A number of questions can be associated with spatial classification:

- Which attributes, predicates, or functions are relevant to the classification process?

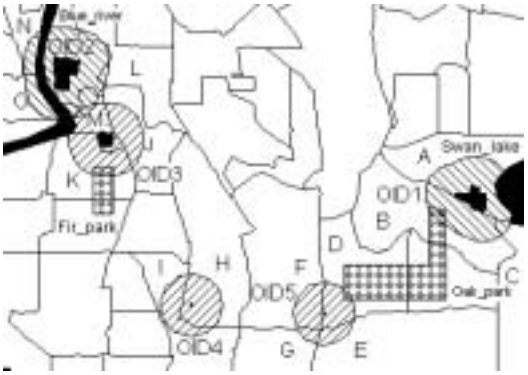

Figure 1: Example map

- How should one determine the size of the buffers that produce classes with high purity?
  Should the size of the buffers be defined by the experts or computed using an algorithm?
- What should be taken into account when aggregates are used:
  - aggregate values for all objects intersecting buffers,
  - aggregate values only for parts of objects intersecting buffers?
- Can one accelerate the process of finding relevant attributes using:
  - sampling,
  - statistical or machine learning methods,
  - progressive refinement methods where rough computations are followed by detailed computations performed only on promising patterns,
  - combination of the above methods?
- How should one handle multiple levels of concepts?

Example: The task is to build a decision tree for the classification of five objects O_i, like shopping malls or stores, that belong to two different classes Y and N which are selected based on attribute *high_profit* with two values Y for "yes" and N for "no". In our example objects OID1 and OID2 belong to class Y and objects OID3, OID4 and OID5 belong to class N as presented in Figure 1. We want to build a decision tree classifying objects O_i based on two types of information: (1) descriptions of the objects in the proximity of objects O_i and (2) non-spatial attributes of the thematic map. The thematic map in our example consists of groups of census blocks. Table 1 shows non-spatial descriptions of some of the block groups. We assume that objects are characterized by areas that are close to them. Thus, one should build buffers around objects O_i and construct decision tree based on census block groups intersecting buffers and other objects located within the buffers. □

BlockID	population	avg_income	crimes
A	5000	$25,500	50
B	4500	$27,500	40
C	5500	$28,300	45
H	6000	$31,500	50
I	5000	$25,500	30
...	...	...	...

Table 1: Description of census block groups

OID	high_profit	others
1	Y	close_to(x,Oak_park), close_to(x,Swan_lake)
2	Y	close_to(x,Blue_river)
3	N	close_to(x,Fir_park), close_to(x,Blue_river)
4	N	
5	N	close_to(x,Oak_park)

Table 2: Descriptions of classified objects

3.1 Handling of Spatial Predicate and Functions

Spatial functions like *driving_distance* or spatial predicates like *close_to* can be used to describe classified objects as it is presented in Table 2. The description should be generalized and then the decision tree algorithm applied. An example of generalized data is presented in Table 3.

The process of finding spatial predicates and functions may be very time consuming. To accelerate this process we use two-step approach in which some rough computations are performed first and then fine computations are done only for the promising patterns. Such process is similar to the two-step approach for mining spatial association rules [13], or to multi-step spatial join approach [2].

OID	high_profit	others
1	Y	close_to(x,park), close_to(x,water)
2	Y	close_to(x,water)
3	N	close_to(x,park), close_to(x,water)
4	N	
5	N	close_to(x,park)

Table 3: Generalized descriptions of classified objects

```
Procedure RELIEF(Predicate_weight, k)
 FOR j := 1 TO max_predicate DO
    Predicate_weight[j] := 0;
 FOR sample_i := 1 TO min(200, k) DO /* size of the sample is set to min(200, k) */
    nearest_hit := FIND_NEAREST_HIT(sample_i); /* find nearest sample from the same class */
    nearest_miss := FIND_NEAREST_MISS(sample_i); /* find nearest sample from different class */
    FOR j := 1 TO max_predicate DO
        Predicate_weight[j] := Predicate_weight[j] − diff(sample_i, nearest_hit, j)
                                                   + diff(sample_i, nearest_miss, j);
 FOR j := 1 TO max_predicate DO
    IF Predicate_weight[j] > min(200, k) * threshold  /√min(200, k)
        THEN Predicate_relevant[j] = TRUE
        ELSE Predicate_relevant[j] = FALSE;

Function diff(sample_i, nearest_hit_or_miss, j)
 /* for symbolic attributes */
    IF sample_i[j] = nearest_hit_or_miss[j] THEN return(0)
        ELSE return(1);
```

Table 4: RELIEF algorithm.

In the first step we can find coarse descriptions for only a sample of objects. For example, one can use Minimum Bounding Rectangles (MBRs) to find *coarse_g_close_to* predicates which imply that MBRs of two objects are within specific distance threshold. Then, some machine learning methods may be used for the extraction of the relevant predicates or functions [12, 23]. In our experiments we used RELIEF algorithm [12] whose algorithmic description is presented in Table 4. The RELIEF algorithm uses nearest neighbor approach to find relevant predicates. For every object s in the sample two nearest neighbors are found where one neighbor belongs to the same class as object s (*nearest hit*) and the other neighbor belongs to a class different than s (*nearest miss*). Based on the descriptions of the nearest neighbors weights for the predicates are modified. If the neighbor belongs to the same class as object s and has the same predicate value, then the weight for this predicate increases. If the neighbor belongs to the same class as object s and does not have the same predicate value, then the weight for this predicate decreases. If the neighbor belongs to different class than object s and has the same predicate value, then the weight for this predicate decreases. If the neighbor belongs to different class than object s and does not have the same predicate value, then the weight for this predicate increases. One can observe that weights for relevant predicates have positive values, while expected weights for non-relevant predicates are zero. Finally, only predicates with weights larger than the predefined threshold are used for classification. The value for the threshold can be set based on statistical theory [12].

In our case only the relevant predicates are computed in detail for all classified objects. Besides improving the efficiency of the algorithm accuracy of the classification and size of the decision tree may improve as well because there will be no branching on irrelevant predicates.

In the construction of the decision tree we used information gain utilized in the ID3 algorithm [20] to find attributes/predicates that partition datasets into classes. Information gain is computed in the following way. If there are n objects from class N and y objects from class Y one can calculate information as:

$$I(y,n) = -\frac{y}{y+n} log_2 \frac{y}{y+n} - \frac{n}{y+n} log_2 \frac{n}{y+n}$$

If an attribute/predicate A with m values is used to partition the data into m classes one can calculate the expected information for this attribute/predicate as:

$$E(A) = \sum_{i=1}^{m} \frac{y_i+n_i}{y+n} I(y_i, n_i)$$

where y_i and n_i are the number of objects from classes Y and N respectively for each of the m values of the attribute/predicate.

Finally, one can calculate information gain if the attribute/predicate A is used to partition the dataset as:

$$info\_gain(A) = I(y,n) - E(A)$$

The attribute/predicate with the highest information gain is used to partition the dataset. Other measures for building decision trees may also be used [8].

Recently some decision tree algorithms that take into account a large number of features describing objects have been proposed [3]. The algorithm [3] can be modified to analyze spatial descriptions of the data. For every classified object a set of generalized predicates that are satisfied by this object is stored. An example of such description is presented in Table 3. For every predicate P from the table we have to find p_1, e.g., the number of objects which belong to class Y and satisfy the pred-

```
Build_tree(Decision_tree, set_of_object's_decriptions)
    Find_Predicate_Counts(set_of_object's_decriptions, Predicate_true, Predicate_false);
    Best_Predicate := Find_Predicate_with_Best_Info_Gain(Decision_tree, Predicate_true, Predicate_false);
    set_of_object's_decriptions_where_Best_Predicate_is_TRUE
                := select_sets(set_of_object's_decriptions, Best_Predicate, TRUE);
    set_of_object's_decriptions_where_Best_Predicate_is_FALSE
                := select_sets(set_of_object's_decriptions, Best_Predicate, FALSE);
    Build_tree(Decision_tree.TRUE_node, set_of_object's_decriptions_where_Best_Predicate_is_TRUE);
    Build_tree(Decision_tree.FALSE_node, set_of_object's_decriptions_where_Best_Predicate_is_FALSE);

Find_Predicate_Counts(set_of_object's_decriptions, Predicate_true, Predicate_false)
    FOR i := 1 TO max_class DO
        { num_objects[i] := 0;
        FOR j := 1 TO max_relevant_predicate DO
            Predicate_true[j, i] := 0; }
    FOR i := 1 TO classified_object_number
        { num_objects[class(object[i])]++;
        FOR EACH predicate_j IN object_descriptions[i]
            Predicate_true[j, class(object[i])]++; }
    FOR i := 1 TO max_relevant_predicate DO
        FOR j := 1 TO max_class DO
            Predicate_false[i, j] := num_objects[j] - Predicate_true[i, j];
```

Table 5: Decision tree building.

icate P and n_1, e.g., the number of objects which belong to class N and satisfy the predicate P. The same calculations have to be done for the negation of the predicate P by finding p_2, e.g., the number of objects which belong to class Y and do not satisfy the predicate P and n_2, e.g., the number of objects which belong to class N and do not satisfy the predicate P. For example, for the predicate $close\_to(x, park)$ one may find $p_1 = 1$, $n_1 = 2$, $p_2 = 1$, and $n_2 = 1$. These values are used to compute the information gain for each of the predicates. In the similar manner for other attributes instead of branching on all attribute values the algorithm branches on one attribute value. For example, we may have separate branches for $sum(population) = MEDIUM$ and for $sum(population) <> MEDIUM$. As claimed in [8] such binary decision trees are usually more compact and more accurate than trees produced by regular ID3 algorithm which branch on all values of the predicates. The pseudo-code of the decision tree algorithm is presented in Table 5.

3.2 Buffer Size and Shape

It is important to determine the proper size and shape for the buffers. It includes buffers which are used for finding aggregate information from thematic maps and buffers used for determining spatial predicates. Buffers represents areas that have an impact on class label attribute of classified objects. For example, in the case of stores and shopping malls a buffer may represent the area where the customers live or work. In business geographics the spatial extent of a store's customers is called *trade area* [19]. The analysis of trade areas helps to answer questions like:

- Which customer segments are driving my business?
- How far will the customers travel to shop?
- Is my site strategically placed to defend or dominate market?

We can analyze different sizes of buffers and choose the best size for the discrimination between classes. It can be done by computing the information gain for all predicting attributes which are aggregates using different sizes of buffers. The size of the buffer with the largest value of information gain is chosen and this size is applied to compute all aggregates for all attributes. The reason to do so is that the decision tree will be first branched on the attribute with the largest information gain value producing the best split of the dataset. Such buffer presents the best trade area for the classification task and trade areas should not change between attributes as they represent the areas with the most influential population of customers.

In addition to that, we may use different criteria to determine shape of the buffers. The buffers may be based on rings, customer penetration polygons, Voronoi diagrams, drive-time polygons, etc. Please see [19] for a detailed discussion. The rings have some advantages like: (1) ease of use, (2) no need to determine trade area based on customer data, (3) easy comparison between sites. One study showed high degree of similarity between rings and customer penetration polygons [19]. In our experiments we used rings (or equidistance buffers for objects that are not points) because no other information was available to

OID	high_profit	Predicates
1	Y	sum_population(x, MEDIUM), avg_income(x, SMALL), close_to(x,park), close_to(x,water)
2	Y	sum_population(x, LARGE), avg_income(x, MEDIUM), close_to(x,water)
3	N	sum_population(x, MEDIUM), avg_income(x, LARGE), close_to(x,park), close_to(x,water)
4	N	sum_population(x, SMALL), avg_income(x, MEDIUM)
5	N	sum_population(x, LARGE), avg_income(x, LARGE), close_to(x,park)

Table 6: Generalized descriptions of classified objects presented as sets of predicates

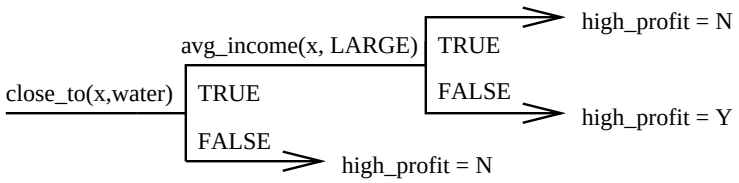

Figure 2: Decision tree for data from Table 6.

determine better neighborhood for the classification task.

3.3 Aggregate Information

Aggregate values for areas close to spatial objects play an important role in the analysis of many business objects like stores, restaurants, gas stations, etc. [19]. To handle aggregate values of non-spatial attributes in thematic maps one can calculate aggregate values for the block groups intersecting buffers. For example, one can notice that only a small part of block C on the map presented in Figure 1 is intersected by buffer for OID1. Thus, one can take into account a part of the particular block group that is intersected by the buffer surrounding classified object and use population living only in this part with the assumption of even distribution. Such calculations, however, would be quite computationally expensive and in reality census blocks are much smaller that the ones shown in Figure 1. Therefore, more blocks are intersected by a buffer and average influence of a single block on aggregate's value is small making a detailed computations less important. The aggregate values for our example are shown in Table 7. This approach transforms descriptions of the classified objects into a flat table which can be easily mined using any classifying algorithm.

The aggregate data can be also generalized and merged with the predicate data so finally each object can be classified using a set of predicates describing properties of both thematic map and other object intersecting trade areas for each object. An example is presented

OID	high_profit	sum(pop)	avg_income
1	Y	15000	$27,127
2	Y	20000	$28,512
3	N	14500	$29,193
4	N	12000	$28,500
5	N	19000	$30,500

Table 7: Descriptions of classified objects

in Table 6. The resulting decision tree is presented in Figure 2.

4 Classification Algorithm

The algorithm for building decision trees for the classification of spatial objects using spatial predicates, spatial functions and thematic maps based on principles discussed in the previous section may be summarized as follows.

Input:

1. Spatial database containing:
   (a) classified objects O_c,
   (b) other spatial objects with non-spatial attributes.
2. Geo-mining query specifying:
   (a) objects to be used in the classification process,
   (b) predictive attributes, predicates, and functions,
   (c) attribute, predicate, or function used as a class label.
3. Set of non-spatial concept hierarchies.

Output: Binary decision tree.

Method:

1. Collect a set S of data specified in the query which consists of a set of classified objects and objects that are used for the description.
2. For the sample of spatial objects O_c from S:
   (a) Build sets of predicates describing all objects using coarse predicates, functions, and attributes.
   (b) Perform generalization of the sets of predicates based on concept hierarchies.
   (c) Find coarse predicates, functions, and attributes which are relevant to the classification task using RELIEF algorithm.
3. Find the best size for the buffer for aggregates of thematic map polygons. It is done by finding for all relevant non-spatial aggregate attributes the size of the buffer X_{max} where the information gain for the aggregated attribute is maximum. Such buffer would be used to compute aggregates for all relevant attributes of thematic maps. Distance-based join index [16] may be used to accelerate the computation.
4. Build sets of predicates describing all objects using relevant fine predicates, functions, and attributes.
5. Perform generalization of the sets of predicates based on concept hierarchies.
6. Generate binary decision tree.

Rationale of the Algorithm. Step 1 is done by spatial query processing. Step 2 is an elimination of irrelevant predicates based on statistical methods. Using data distribution and user defined confidence level one can compute threshold used to eliminate irrelevant attributes. Step 3 finds the best size for the buffer for the most relevant aggregate attribute and uses this size for all other attributes because trade areas that, for example, represent clients who deal with a store, remain the same for all attributes. Step 4 is spatial query processing. Step 5 has been verified in [9]. Step 6 generates binary decision tree using ID3 algorithm [20] modified to allow processing of description in the form of sets of predicates as described in the previous section.

The algorithm presented above introduces a number of optimization techniques in the comparison with algorithm suggested in [5], namely:

1. It uses progressive refinement approach (sampling and coarse predicates followed by fine computations for promising patterns) to accelerate the computation.
2. It may use aggregate information.
3. It may use distance-based join index [16] to accelerate query processing. Neighborhood index [5] may not be the best for that purpose.
4. It may use concept hierarchies which result in simpler decision trees and faster computations [11].
5. It uses relevance analysis process to eliminate predicates and attributes that do not contribute to the quality of classification.

Complexity analysis. Execution time of the algorithm presented above can be estimated using equations presented below. Time to classify objects without filtering relevant attributes is presented by Equation (1). Component $k \times d \times t_f$ presents time of getting fine predicates and component $k \times d_{total} \times t_{build}$ presents time of building decision tree from table of predicate sets. Time of classifying objects with filtering relevant attributes using first only coarse predicates as it was done in our experiments is presented by Equation (2). Component $min(200, k) \times d \times t_c$ presents time to calculate coarse predicates for the sample of objects that are classified and component $(min(200, k))^2 \times d \times t_{filter}$ presents time to filter relevant predicates. Size of the sample is set to $min(200, k)$. Component F_{ratio} $\times k \times d \times t_f$ presents time to calculate relevant fine predicates and component $F_{ratio} \times$ $k \times d_{total} \times t_{build}$ presents time to build decision tree from table of relevant predicate sets.

Table 8 lists some parameters used in the cost analysis. The values of the parameters are based on experiments performed on Pentium 166MHz computer with 64MB of memory. Estimated times are presented in Figure 3.

$$t_1 = k \times d \times t_f + k \times d_{total} \times t_{build} \quad (1)$$

$$\begin{aligned} t_2 = {} & min(200, k) \times d \times t_c \\ & + (min(200, k))^2 \times d \times t_{filter} \\ & + F_{ratio} \times k \times d \times t_f \\ & + F_{ratio} \times k \times d_{total} \times t_{build} \end{aligned} \quad (2)$$

5 Performance Evaluation

We evaluated the quality and the efficiency of our algorithm for the classification of data

Name	Value	Meaning
k		number of objects being classified
d	3	average number of predicates describing an object
d_{total}	60	number of different predicates describing objects
F_{ratio}	0.5	ratio of relevant predicates
t_c	5ms	cost of finding one coarse predicate
t_f	0.13s	cost of computing one fine predicate
t_{build}	0.05ms	constant for building decision tree
t_{filter}	2μs	constant for filtering out non-relevant attributes

Table 8: Database parameters.

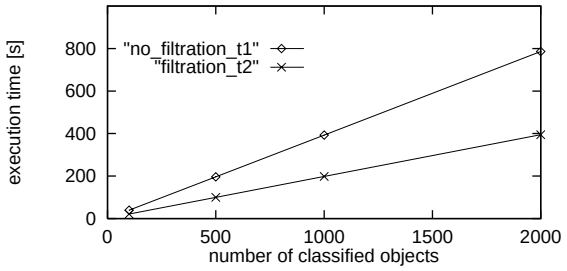

Figure 3: Execution times of the algorithm

threshold	number of relevant predicates	time [s]	quality
$-\infty$	58	318	82%
0.0	13	105	99%
0.1	4	85	100%
0.2	3	82	100%
0.3	2	76	100%

Table 9: Results of the algorithm for artificial data

based on predicates *g_close_to* which represent spatial relations between objects that are classified and other objects located within distance threshold. Relations of the data to thematic maps were not taken into account. In order to evaluate classification quality and execution time of the algorithm we created a synthetic dataset which was merged with the TIGER[1] data [22] for Washington state. The synthetic data contained 1800 polygons with 5 to 55 edges. The polygons belonged to 4 sets: two class label sets (T00 and T01) and two predictive sets (S00 and S01). If object x satisfies predicate *g_close_to(x, S00, 1km)* $\oplus$ *g_close_to(x, S01, 1km)* then this object belongs to the set T00, otherwise it belongs to the set T01. TIGER data contains over 87,000 polygons, lines and points presenting roads, parks, rivers, lakes and other objects in Washington state. MapInfo Professional 4.1[2] Geographic Information System [17] was used as spatial database for the experiments. We have tested quality of classification and time of the execution of the algorithm. Relevance analysis was done based on 100 objects from each class label set. The coarse predicate used for the relevance analysis was *coarse_g_close_to(x, y, 1km)*. This predicate implies that MBRs of two objects are located within one kilometer distance. After relevance analysis was performed fine relevant predicates were found for all objects. 400 objects from each class label set were used for training set and the remaining 50 objects were used for testing. The results are summarized in Table 9.

As one can observe the quality of the classification process increases drastically when relevance analysis is performed. If we choose to do classification based on the algorithm proposed in [5] we get a poor quality because this algorithm performs no relevance analysis. After applying relevance analysis execution time decreased 3 to 4 times depending on the value of the threshold. The algorithm [5] is based on neighborhood indices that avoid process of spatial join which is necessary to extract spatial predicates. When no neighborhood indices exist the algorithm [5] has to produce them which results in the classification time similar to the time in our experiment for threshold equal to $-\infty$ when all predicates are used for decision tree construction. The process of relevance analysis took about 17 seconds where 16 seconds were spent on spatial operations. Process of building decision tree from 800 sets of predicates took about 1 second for every experiment.

We also performed a number of experiments with real data from the TIGER dataset. In most cases the accuracy of the classification increased drastically when relevance analysis was used. The best results were for threshold values between 0.0 and 0.2. The time to perform relevance analysis and build decision tree was between 10% and 50% shorter than the time necessary to build decision tree based on all predicates.

We compared the quality of classification when decision trees were constructed based

[1] TIGER is the registered trademark of U.S. Census Bureau.

[2] MapInfo Professional is the registered trademark of MapInfo Corporation.

on fine predicates with quality of classification when decision trees were constructed based on coarse predicates. In all cases except one, when coarse classification yielded 1% better quality, the classification with fine predicates produced better quality of up to 8%.

Finally, we compared the algorithm where coarse predicates are used for relevance analysis with the algorithm where fine predicates are used for that purpose. Performed experiments showed that the quality of classification was fairly similar for both algorithms, while relevance analysis using fine predicates usually yielded slightly smaller set of relevant predicates. In addition to fine predicates which were judged as relevant ones some *coarse_g_close_to* predicates representing large objects were chosen as relevant ones. This happens if objects from the same class are clustered together as in the case of some of our training data. In this case all objects from the same cluster may satisfy coarse predicate for a large object even though the object related to this predicate is relatively distant and thus, the fine predicates are not satisfied.

6 Discussion and Conclusion

Classification of geographical objects enables researchers to explore interesting relations between spatial and non-spatial data. We have proposed and partially implemented an interesting and efficient method for the classification of spatial objects. The proposed method enables classification of spatial objects based on aggregate values of non-spatial attributes for neighboring regions. It also takes into account spatial relations between objects on the map which may be represented in the form of predicates. The algorithm first performs less costly, approximate spatial computations to obtain a sample of approximate spatial predicates. Relevance analysis is performed only for such sample. Then, refined computations are done only for the set of promising patterns producing smaller and more accurate decision trees.

Conducted experiments showed the importance of relevance analysis which resulted in better quality of decision trees in comparison to the trees produced by algorithm proposed in [5]. We also observed that the time to build decision trees was shorter when relevance analysis is performed on a sample of the objects and only relevant attributes are computed for all objects in comparison to the time when all predicates were used for the construction of the decision tree.

We plan to integrate the classification algorithm with spatial query engine. For example, when sets of predicates are computed for a mall there may be no need to compute intersections of the buffer for this mall with all segments of a highway. If a predicate *g_close_to (mall, highway, 1km)* is true for one segment of the highway then there is no need to compute intersections with other sections of that highway.

We plan to perform experiments using aggregate values for thematic maps and varied distances for *close_to* spatial predicates. The experiments analyzing the impact of different levels of concepts used for the classification on classification accuracy should be constructed. Finally, we would like to integrate this method with our spatial data mining prototype *GeoMiner* [10].

References

[1] W. G. Aref and H. Samet. Extending DBMS with Spatial Operations. *Proc. 2nd Symp. SSD'91*, pp. 299–318, Zurich, Switzerland, Aug. 1991.

[2] T. Brinkhoff, H. P. Kriegel, R. Schneider, B. Seege. Multistep Processing of Spatial Joins. In *Proc. 1994 ACM-SIGMOD Conf. Management of Data*, Minneapolis, Minnesota, pp. 197–208, May 1994.

[3] W. W. Cohen. Learning Trees and Rules with Set-valued Features. *Proc. of the 13th National Conference on Artificial Intelligence (AAAI)*, Portland, OR, 1996.

[4] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases. In *Proc. of the Second International Conference on Data Mining KDD-96*, pp. 226–231, Portland, Oregon, August 1996.

[5] M. Ester, H.-P. Kriegel, and J. Sander. Spatial data mining: A database approach. *Proc. Int. Symp. on Large Spatial Databases (SSD'97)*, Berlin, Germany, pp. 47–66, August 1997.

[6] U. M. Fayyad, S. G. Djorgovski, and N. Weir. Automating the Analysis and Cataloging of Sky Surveys. In [7]

[7] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, Menlo Park, CA, 1996.

[8] U. M. Fayyad, K. B. Irani. The Attribute Selection Problem in Decision Tree Generation. *Proc. of the Tenth National Conference on Artificial Intelligence AAAI-92*, pp. 104–110, Cambridge, MA, MIT Press.

[9] J. Han, and Y. Fu. Exploration of the Power of Attribute-Oriented Induction in Data Mining. In [7].

[10] J. Han, K. Koperski, and N. Stefanovic. GeoMiner: A system prototype for spatial data mining. In *Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data*, pp. 553–556, Tucson, Arizona, May 1997.

[11] M. Kamber, L. Winstone, W. Gong, S. Cheng, and J. Han. Generalization and Decision Tree Induction: Efficient Classification in Data Mining. *Proc. of 1997 Int. Workshop Research Issues on Data Engineering (RIDE'97)*(April, Birmingham, England), 1997, pp. 111-120.

[12] K. Kira, and L. A. Rendell. The Feature Selection Problem: Traditional Methods and a New Algorithm. *Proc. of the Tenth National Conference on Artificial Intelligence AAAI-92* pp. 129–134, Cambridge, MA, MIT Press.

[13] K. Koperski, and J. Han. Discovery of Spatial Association Rules in Geographic Information Databases. In *Advances in Spatial Databases, Proceedings of 4th Symposium, SSD'95.* (Aug. 6-9, Portland, Maine). Springer-Verlag, Berlin, 1995, pp. 47–66.

[14] K. Koperski, J. Han, and J. Adhikary. Mining Knowledge in Geographical Data. In *Communications of ACM*, 1998. (to appear).

[15] W. Lu, J. Han, and B. C. Ooi. Discovery of General Knowledge in Large Spatial Databases. In *Proceedings of Far East Workshop on Geographic Information Systems* (June 21-22, Singapore). World Scientific, Singapore, 1993, pp. 275-289.

[16] W. Lu and J. Han. Distance-Associated Join Indeces for Spatial Range Search. *Proceedings of the Interantional Conference on Data Engineering* pp. 284–292, 1992.

[17] MapInfo Corporation. MapInfo Professional Home Page. www.mapinfo.com/events/mipro/mipro.html

[18] R. T. Ng and Y. Yu. Discovering Strong, Common and Discriminating Characteristics of Clusters from Thematic Maps. *Proceedings of the 11th Annual Symposium on Geographic Information Systems* pp. 392-394, 1997.

[19] K. Peterson. A Trade Area Primer. *Business Geographics*, Vol. 5, No. 9, 1997, pp. 18–21.

[20] J. R. Quinlan. Induction of Decision Trees. *Machine Learning*, No. 1, 1986, pp. 81–106.

[21] S. R. Safavian and D. Landgrebe. A Survey of Decision Tree Classifier Technology. *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 21, No. 3, 1991, pp. 660–674.

[22] U.S. Census Bureau. Tiger/Line Documentation. www.census.gov/geo/www/tiger/tl95doc.html

[23] D. Wetteschereck, D. W. Aha, and T. Mohri. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, No. 10, 1997, pp. 1–37.