A methodology for dynamic data mining based on fuzzy clustering

Fernando Crespo[a], Richard Weber[b,*]

[a] *Departamento de Ingeniería Industrial y Sistemas, Universidad Católica de Chile, Chile*
[b] *Departamento de Ingeniería Industrial, Universidad de Chile, República 701, Santiago, Chile*

Abstract

Dynamic data mining is increasingly attracting attention from the respective research community. On the other hand, users of installed data mining systems are also interested in the related techniques and will be even more since most of these installations will need to be updated in the future. For each data mining technique used, we need different methodologies for dynamic data mining. In this paper, we present a methodology for dynamic data mining based on fuzzy clustering. Using the implementation of the proposed system we show its benefits in two application areas: customer segmentation and traffic management.
c

Keywords: Dynamic data mining; Fuzzy clustering; Customer segmentation

1. Introduction

Data mining is part of an iterative process called KDD—knowledge discovery in databases [9]. This process consists basically of steps that are performed before doing data mining (such as: selection, pre-processing, transformation of data [8]), the actual data mining part, and subsequent steps (such as: interpretation, evaluation of results). Clustering techniques are used for data mining if the task is to group similar objects (e.g. bank customers) into the same classes (segments) whereas objects of different classes show different characteristics.

Once a data mining system has been installed and is being used in daily operation, its user has to be concerned about the system's future performance because the extracted knowledge is based on past behavior of the analyzed objects.

[*] Corresponding author. Tel.: +5626784056; fax: +5626897895.
E-mail addresses: facrespo@puc.cl (F. Crespo), rweber@dii.uchile.cl (R. Weber).

If future behavior is very similar to past behavior (e.g. if bank customer do not change their preferences over time) using the initial data mining system could be justified. If, however, behavior changes over time, the continued use of the initial system could lead to non-acceptable results and—as a consequence—to non-acceptable decisions based on these results. Here is where dynamic data mining comes in, a new research area that is concerned about any combination of traditional data mining with dynamic aspects.

It becomes apparent, though, that "something" has to be done if a user wants to keep applying his/her data mining system in a changing environment. Basically, there are three strategies:

1. The user neglects changes in the environment and keeps on applying the initial system without any updating.
2. Every certain period—which depends on the particular application—a new system is developed using all the available data.
3. Based on the initial system and "new data" an update of the classifier is performed.

Strategy 1 has the advantage of being "computationally cheap" because no update of the data mining system is performed. Furthermore, it does not require changes in subsequent processes, such as, e.g. design of marketing campaigns for customer segments. Its disadvantage is that current tendencies could not be detected.

Following strategy 2, the user has always a system "up-to-date" due to the use of current data. Disadvantages of this strategy are the computational costs of creating each time from scratch the respective system.

In this paper, we propose a methodology following strategy 3 where we first identify the need for a system's update by applying it to new data and second perform the update by using efficiently the previous system. This approach has two advantages: it is computationally more efficient than "learning from scratch" and—may be more important—identifying explicitly the changes of the system could provide further insights into the changes of the respective environment. Our methodology is application-independent and based on parameters, which the user can adjust according to the respective application.

Section 2 gives an overview on recent developments of dynamic data mining where different approaches and tendencies are discussed. Section 3 presents a methodology for dynamic data mining based on fuzzy clustering for the particular case of changing classes and class structures. Section 4 shows each step of the new methodology in detail using a simulated data set. Section 5 presents results applying the proposed methodology for dynamic customer segmentation as well as for dynamic traffic state identification. Section 6 concludes the presented work and provides future perspectives.

2. Recent developments of dynamic data mining

Within the area of data mining various methods have been developed in order to find useful information in a set of data. Among the most important ones are decision trees, neural networks, association rules, and clustering methods [9].

For each of the above-mentioned data mining methods, updating has different aspects and some updating approaches have been proposed, as we will see next.

	Static classes	**Dynamic classes**
Static objects	Classical case	**Proposed methodology.**
Dynamic objects	Clustering of trajectories	

Fig. 1. Taxonomy of dynamic data mining for clustering.

- *Decision trees*: Various techniques for incremental learning and tree restructuring (see e.g. [22]) as well as the identification of concept drift (see e.g. [5]), have been proposed in the literature.
- *Neural networks*: Updating is often used in the sense of re-learning or improving the net's performance by learning with new examples presented to the network.
- *Association rules*: Raghavan et al. have developed systems for dynamic data mining for association rules; see e.g. [20].
- *Clustering*: Below, we describe in more detail approaches for dynamic data mining using clustering techniques that can be found in literature.

Recent developments of clustering systems using dynamic elements are concerned about modeling the clustering process dynamically, i.e. adaptations of the algorithm are performed while applying it to a static set of data.

CHAMELEON is such a system, which uses hierarchical clustering where the merging decision on each hierarchy dynamically adapts to the current cluster characteristics [11].

Other than in hierarchical clustering, objective function-based clustering methods, such as, e.g. c-means and fuzzy c-means, need to specify the number of clusters (here: c) before running the respective algorithm. The determination of an appropriate number of clusters has been subject of several investigations (see e.g. [4,24]). "Dynamic partitional clustering using evolution strategies" is an approach where the "cluster number is optimized during run time" [16] using evolutionary algorithms.

The clustering methods mentioned before are using dynamic elements during their application to a static set of data. Next, we will present data mining methods, which can be applied to a dynamic set of data. Fig. 1 provides a taxonomy of various situations where we are faced with "dynamic data" in the case of clustering.

In several situations, the current feature values are not sufficient in order to explain the underlying phenomenon. In such cases, we may be interested in the development of these values such that features become trajectories. This is the case for example in medicine, where patients' conditions depend not only on the current values of, e.g. blood pressure but also on their development during the relevant past. Another example could be machine monitoring, where a machine's condition depends on the development of its temperature [21]. In these cases, we are dealing with dynamic objects, i.e. feature values are trajectories instead of real values.

In order to be able to cluster such dynamic objects, we need a distance measure between two vectors where each component is a trajectory (function) instead of a real number. Functional fuzzy c-means [10] is a fuzzy clustering algorithm that has been developed for such purposes where the similarity between two trajectories is determined using membership functions. Alternatives for determining the similarity between two functions are Wavelets (see e.g. [1]) and Time Warping (see e.g. [13]).

Another approach where trajectories of feature values are clustered is Matryoshka [17], which is based on a hidden Markov model (HMM) for temporal data clustering. Given as input objects described by temporal data (trajectories), it determines the optimal number of classes where each class is characterized as an HMM and an assignment of objects to classes.

This area of data mining research where objects are described by dynamic data (trajectories) is called temporal data mining [2] and belongs to the more general field of dynamic data mining, where also classes of objects can be dynamic.

3. Dynamic data mining using fuzzy clustering

In this section, we present a methodology for dynamic data mining using fuzzy clustering that assigns static objects to "dynamic classes", i.e. classes with changing structure over time.

It starts with a given classifier and a set of new objects, i.e. objects that appeared after the creation of the current classifier. The period between the creation of a classifier and its update is called a *cycle*. The length of such a cycle depends on the particular application, e.g. we may want to update buying behavior of customers in a supermarket once a year whereas a system for dynamic machine monitoring should be updated every 5 minutes.

Section 3.1 contains the terminology used. In Section 3.2, we present a general view of our approach before we provide the details of each of its steps in Section 3.3.

3.1. Basic terminology

From the huge number of clustering methods (see e.g. [6]) we focus in our methodology on fuzzy clustering (see e.g. [25]), since the degree of membership of an object to the classes found provides a strong tool for the identification of changing class structures. We are working with fuzzy c-means in order to build an initial classifier and to update our classifier in each cycle but the presented methodology can be extended to any other technique which determines such degrees of membership (e.g. possibilistic clustering). Details on fuzzy c-means and a general presentation of fuzzy clustering can be found, e.g. in [4].

We use the following notation:

c	number of classes
n	number of objects
p	number of features describing each object
x_j	feature vector of object j, $j = 1, \ldots, n$
v_i	center of class i, $i = 1, \ldots, c$
μ_{ij}	degree of membership of object j to class i, $i = 1, \ldots, c$, $j = 1, \ldots, n$
M^t	$(c \times n)$-matrix in cycle t with μ_{ij} at position (i, j), $i = 1, \ldots, c$, $j = 1, \ldots, n$

3.2. Global view of the proposed methodology for dynamic data mining using fuzzy clustering

Before we present our methodology for dynamic data mining, it is important to mention the difference between identifiable objects and those we cannot identify. In customer segmentation, e.g. where each customer has its ID, we have identifiable objects (customers) and we can trace the development of a particular customer over time. If, on the other hand, we want to study, e.g. the buying behavior of customers of a supermarket we cannot identify our objects and therefore we cannot trace the behavior of a specific customer over time.

In this paper, we develop an approach for the case of non-identifiable objects. For the other case of identifiable objects, we presented updating schemes in [7].

Possible changes of the classifier's structure we want to identify in one cycle are:

- creation of new classes,
- elimination of classes,
- movement of classes in the feature space.

Our methodology applies the following five steps in order to detect these changes.

Step I: Identify objects that represent changes. For each of the new objects we want to know if it can be explained well by the given classifier or not. With other words, we want to identify objects that represent possible changes in the classifier structure because they are not well classified. If there are many objects that represent such possible changes we proceed with Step II, in the other case we go immediately to Step III.

Step II: Determine changes of class structure. In Step I, we have identified the objects that could not be well classified. Now we want to decide if we have to change the classifier structure (i.e. create new classes) or if it is sufficient to just move the existing classes.

If "many new objects" are identified in Step I to represent changes, we have to create a new class, in the other case we just move the existing classes in the feature space.

Step III: Change the class structure. Here we perform the changes according to the results of Steps I and II (move or create classes).

Step III.1: Move classes. We update the position of the existing class centers having the new objects and knowing that they do not ask for a structural change. The respective formulas are given in Section 3.3.

Step III.2: Create classes. If we know that classes have to be created, we first determine an appropriate class number. Then we apply fuzzy c-means with the new class number to the available data set.

Step IV: Identify trajectories of classes. We identify trajectories of the classes from the previous cycles in order to decide if they received new objects. Classes that did not receive new objects during several cycles have to be eliminated.

Step V: Eliminate unchanged classes. Based on the result of Step IV, we eliminate classes that did not receive new objects during an "acceptable period".

Fig. 2 provides a global view of the proposed methodology.

3.3. *Detailed description of a cycle of the proposed methodology*

During a cycle m new objects have appeared. Let $k = n+1, \ldots, n+m$ be the index of the m new objects.

Step I: Identify objects that represent changes. Goal of this step is to identify those objects that are not well classified by the given classifier. For this reason, we need the distances between each pair of the currently existing class centers. This pair wise distance is

$$d(v_i, v_j) \quad \forall i \neq j, \quad i, j \in \{1, \ldots, c\}.$$

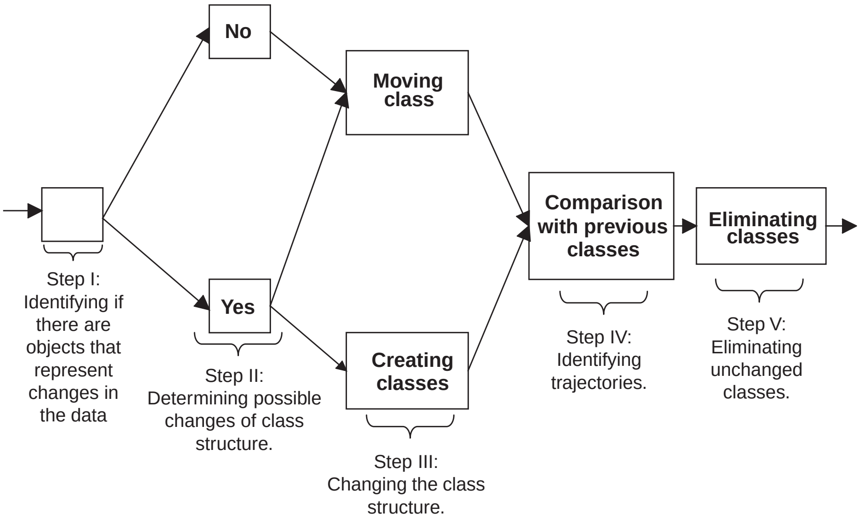
No
Moving class
Comparison with previous classes
Eliminating classes
Yes
Creating classes
Step I: Identifying if there are objects that represent changes in the data
Step II: Determining possible changes of class structure.
Step III: Changing the class structure.
Step IV: Identifying trajectories.
Step V: Eliminating unchanged classes.

Fig. 2. Global view of methodology for dynamic data mining considering changing class structures.

Additionally, we need the distance between the new object k and the center v_i from the current class structure. This distance is

$$\hat{d}_{ik} = \hat{d}(x_k, v_i), \quad i \in \{1, \dots, c\}, \quad k \in \{n+1, \dots, n+m\}.$$

Finally, we apply the given classifier to the m new objects and obtain

$$\hat{\mu}_{ik} = \text{membership of new object } k \text{ to class } i, \quad i \in \{1, \dots, c\} \quad k \in \{n+1, \dots, n+m\}.$$

Based on this preliminary work, we identify objects that represent changes of the given classifier. In this step, we just want to know if it is adequate to create a new class or if class movement is sufficient. Therefore, we want to identify objects that are “not classified well by the existing classifier” and “far away from the current classes”. We apply the following two conditions in order to detect such objects.

Condition 1: $|\hat{\mu}_{ik} - 1/c| \leqslant \alpha \ \forall k \in \{n+1, \dots, n+m\} \ \forall i \in \{1, \dots, c\}$.

Given a value $\alpha \geqslant 0$, Condition 1 determines those objects that have a membership value close to the inverse of the class number. A new object k that has all its membership values close to $1/c$ cannot be classified satisfactorily.

In order to determine the parameter α we propose one of the following strategies:

- It can be determined context-dependent if the respective knowledge exists.
- If we know the correct class of some objects in a given cycle, we can determine α dynamically depending on the rate of correct classification of objects. For example, if the rate of correct classification is high in one cycle, α should be closer to 0 in the following cycle in order to generate less changes of the classifier structure.

Condition 2: $\hat{d}_{ik} > \frac{1}{2} \min\{d(v_i, v_j)\} \ \forall k \in \{n+1, \dots, n+m\} \ \forall i \neq j \in \{1, \dots, c\}$.

Condition 2 determines if a new object k is "far away from the current classes". We assume this to be the case, if its distance to each class i is larger than the minimal distance between two classes i and j.

Based on these two conditions we define

$$1_{IC}(x_k) = \begin{cases} 1 & x_k \text{ fulfills Conditions 1 and 2,} \\ 0 & \text{else.} \end{cases}$$

With other words: $1_{IC}(x_k)$ has value 1 if and only if object k cannot be classified well by the current classifier.

If $\sum_{k=n+1}^{n+m} 1_{IC}(x_k) = 0$, we proceed with Step III.1, else we go to Step II.

Step II: Determine changes of class structure. Given that at least one object represents a possible change of the classifier structure we now want to check if we need a new class or if moving the existing classes is sufficient. To do this we apply the following criterion:

$$\frac{\sum_{k=n+1}^{n+m} 1_{IC}(x_k)}{m} \geqslant \beta \quad \text{with a parameter } \beta, \quad 0 \leqslant \beta \leqslant 1.$$

If the relation between new objects that represent a possible change and the total number of new objects (m) is above a predefined threshold β we create new classes (III.2). In the other case, we just move the existing classes (III.1).

In order to determine the parameter β we propose one of the following strategies:

- It can be determined context-dependent if the respective knowledge exists.
- If we know the correct class of the objects in a given cycle, we can determine β as the rate of correct classification of objects. This way we have an adaptive parameter setting for β in the following cycle, i.e. if many objects are classified correctly in one cycle it needs more new objects that represent a possible change in order to create a new class in the following cycle.

Step III: Change the class structure. Depending on the result of Step II, we want to create new classes or just move the existing ones.

Step III.1: Move classes. There are basically two options for class movement:

- Applying the underlying clustering algorithm (in our case: fuzzy c-means) with previous and new objects (without changing the number of classes).
- Determining "class centers" representing the new objects and combining them with the previous class centers.

The second option of moving classes combines the centers of the existing classes with centers representing the new objects belonging to the same class, respectively. For this reason, we define the indicator function of a new object k for class i

$$1_{C_i}(x_k) = \begin{cases} 1 & \text{object } k \text{ is assigned to class } i, \\ 0 & \text{else.} \end{cases}$$

Throughout this paper we assign an object to a class if it has its highest membership value to this class. For each class i, we determine the class centers representing only the new objects of this class by

$$v_i^* = \frac{\sum_{k=n+1}^{n+m}(1 - 1_{IC}(x_k))(\hat{\mu}_{ik})^m x_k}{\sum_{k=n+1}^{n+m}(1 - 1_{IC}(x_k))(\hat{\mu}_{ik})^m}, \quad 1 \leqslant i \leqslant c.$$

Combining these "centers of the new objects" with the previous centers we determine the new class center by

$$\hat{v}_i = (1 - \lambda_i)v_i + \lambda_i v_i^*,$$

where the weight λ_i indicates the proportion of new objects assigned to class i

$$\lambda_i = \frac{\sum_{k=n+1}^{n+m}(1_{C_i}(x_k) \cdot (1 - 1_{IC}(x_k)) \cdot \hat{\mu}_{ik})}{\sum_{j=1}^{n}(1_{C_i}(x_j) \cdot \mu_{ij}) + \sum_{k=n+1}^{n+m}(1_{C_i}(x_k) \cdot (1 - 1_{IC}(x_k)) \cdot \hat{\mu}_{ik})}.$$

Step III.2: Create classes. If Step II tells us that we have to create one or more new classes since many new objects cannot be assigned properly to the existing classes, we first have to determine an adequate new number of classes (e.g. c^{new}). For this purpose we apply the concept presented by Li and Mukaidono [18], which is called *structure strength* and is based on the idea that "the knowledge of a part allows us to guess easily the rest of the whole". The authors define a loss function $L(c)$ as within-group sum-of-squared-error (WGSS) for a given cluster solution

$$L(c) = \sum_{k=1}^{N} \sum_{i=1}^{c} u_{ik} d_{ik}^2,$$

where c is the number of classes, N the number of objects, u_{ik} the degree of membership of object k to class i and d_{ik} the distance between object k and class center i.

Based on this loss function the authors determine the number c of classes as follows:

$$\begin{aligned} S(c) &= \text{structure strength} \\ &= \alpha(\text{effectiveness of classification}) + (1 - \alpha)(\text{accuracy of classification}) \\ &= \alpha \log(N/c) + (1 - \alpha) \log(L(1)/L(c)). \end{aligned}$$

The authors suggest to measure the effectiveness by $\log(N/c)$, i.e. a classification with less classes is more effective. They propose to measure the accuracy by the term $\log(L(1)/L(c))$, i.e. a classification with more classes is more accurate. $L(1)$ is the variance of the entire data set and α is the weight between effectiveness and accuracy. The authors suggest $\alpha = 0.5$ in the case of an unbiased estimation, i.e. effectiveness and accuracy should have the same weight [18].

The value c, which maximizes $S(c)$, is supposed to be an adequate class number.

Having an adequate new number of classes (e.g. c^{new}) for all objects we continue with iterations by our basic clustering algorithm (here: fuzzy c-means) in order to determine the best c^{new} classes representing all objects.

Step IV: Identify trajectories. Having performed necessary movements and/or creations of classes we have to check if there are classes that should be eliminated. As preparation for the elimination step (Step V), we identify the development of classes during previous cycles. To reach this we have to identify trajectories of classes, a non-trivial task since we are dealing with non-identifiable objects (see Section 3.2), and we need a counter c_i^t for each class i in cycle t.

We have the following two cases for each class i in cycle t:

- Class i has been created in cycle $t-1$. In this case we set its counter $c_i^t = 1$.
- Class i is the result of moving a certain class j in cycle $t-1$.
  In this case we set: $c_i^t = c_j^{t-1} + 1$.

Step V: Eliminate unchanged classes. The idea of *eliminating* a class can be stated in the following way: “A class has to be eliminated if it did not receive new objects for a *long* period.” It has to be defined what “*long* period” means.

In Step IV, we identified for each class i in cycle t its counter (c_i^t), i.e. the number of cycles it has been active. We define a maximum number of cycles a class could stay active without receiving new objects (here: T cycles). If a class does not receive new objects for T cycles, it will be eliminated. In the proposed methodology, we work with the same threshold value of T cycles for all classes.

In order to determine the parameter T we propose one of the following strategies:

- It can be determined context-dependent if the respective knowledge exists.
- We can determine T applying in each cycle the concept of *structure strength* presented in Step III.2 and introduced by Li and Mukaidono [18], in order to determine an adequate number of classes.

It should be mentioned that eliminating a class does not mean to forget completely this class. If we eliminate a class from the current class structure, the respective information will be kept in a separate memory. If in a later cycle in Step III.2 (create classes) a new class is generated that is very similar to a previously eliminated one, we get additional information from the periodically entering data. This way we can detect “cycles” in our data structure, if e.g. a class is eliminated and newly created with certain periodicity (e.g. seasonal customer behavior).

4. Application of the proposed methodology to simulated data

We have implemented the proposed methodology in Matlab® in order to demonstrate its effectiveness in different applications. In this section, we apply it to a simulated data set, so that we can show each of its steps (movement of class centers, eliminating classes, creating classes).

4.1. Description of data set used and initial solution

We generated 500 objects for each of the four classes we used as initial solution. Each object is described by two features whose values are normally distributed with the following mean values: $(0,15)$, $(8,35)$, $(15,0)$, and $(15,20)$, respectively. Fig. 3 shows the initial data set graphically.

We applied fuzzy c-means with $c = 4$ classes and $m = 2$ as fuzzifier. Table 1 presents the respective cluster solution.

4.2. Detailed application of proposed methodology

We chose the following parameters:

- *Step* I: in order to identify objects that represent changes we use $\alpha = 0.05$;

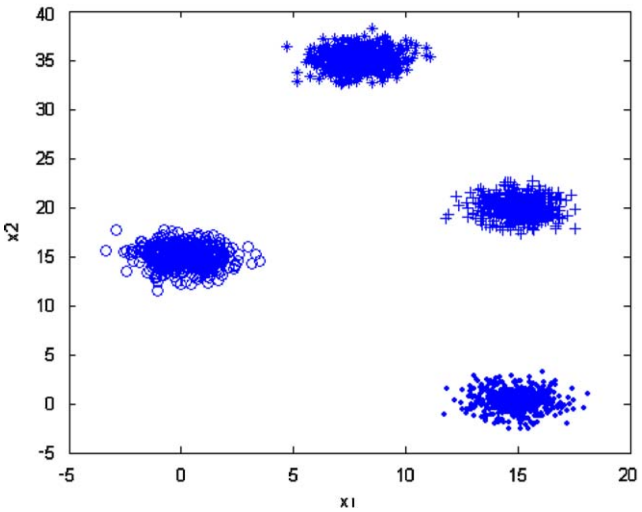

Fig. 3. Initial data set of simulated data.

Table 1
Initial cluster solution of simulated data set

Class	Variables		Number of objects
	$X1$	$X2$	
1	14.98	0.23	500
2	0.19	15.03	500
3	8.01	35.01	500
4	15.05	20.08	500
Total			2000

- *Step* II: with $\beta = 0.2$ we create a new class if the number of objects representing changes (result from Step I) is larger than 20% of the total number of new objects;
- in Step V, we eliminate a class if it did not receive new objects for $T = 2$ periods.

In the first cycle, 600 new objects arrive as shown in Fig. 4.

According to conditions 1 and 2 of Step I, no new object represents changes of the classifier structure. We go immediately to Step III and move the centers of classes 1, 3, and 4 who received 200 new objects each. Results are shown in Table 2. The last three columns of this table contain binary variables indicating if the respective change has been performed.

In the second cycle 500 new objects arrive. The entire set of objects is shown in Fig. 5.

Applying our methodology in cycle 2 we get Table 3.

Since 200 out of 500 new objects (40%) represent changes, we created a new class (class 5). Additionally, centers of classes 1, 3, and 4 have been moved. In Step V, we eliminated class 2 in cycle 2 because it did not receive new objects during $T = 2$ cycles.

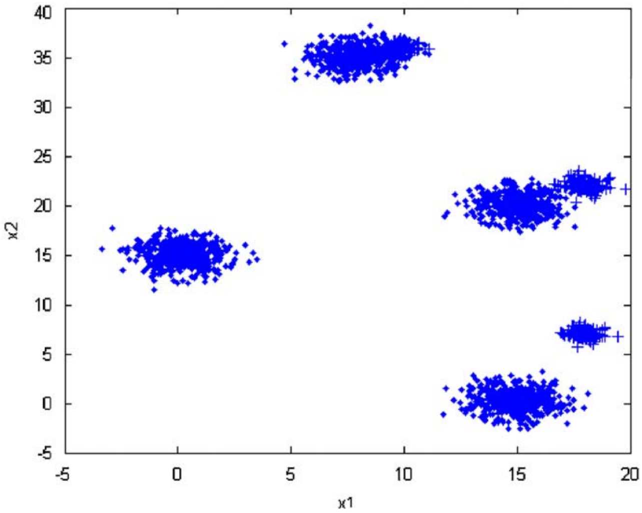

Fig. 4. Complete data set including new objects from cycle 1.

Table 2
Result after first cycle

Class	Variables		Number of objects	Class moved	Class created	Class eliminated
	$X1$	$X2$				
1	15.55	1.65	700	1	0	0
2	0.19	15.03	500	0	0	0
3	8.70	35.54	700	1	0	0
4	15.72	20.36	700	1	0	0
Total			2600			

In the third cycle, 600 new objects arrive leading to the data set as shown in Fig. 6 (objects belonging to the previously eliminated class 2 are not shown).

Applying our methodology in the third cycle we get the result as shown in Table 4.

Analyzing explicitly the performed changes, e.g. elimination of class 2 and creation of class 5 could provide further insights into the behavior of the observed system.

5. Application of the proposed methodology to real data

In this section, we show the versatility of the proposed methodology, presenting two applications: one for dynamic customer segmentation and one for traffic management.

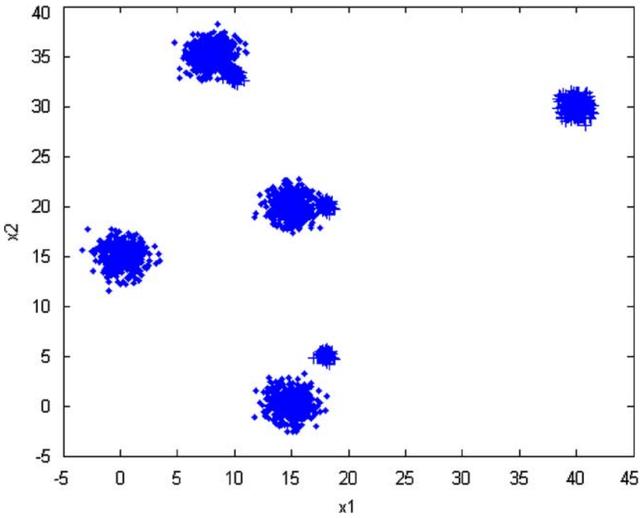

Fig. 5. Complete data set including new objects from cycle 2.

Table 3
Result after second cycle

Class	Variables		Number of objects	Class moved	Class created	Class eliminated
	$X1$	$X2$				
1	15.83	2.03	800	1	0	0
2	—	—	0	0	0	1
3	8.86	35.24	800	1	0	0
4	16.05	20.39	800	1	0	0
5	39.98	29.96	200	0	1	0
Total			2600			

5.1. Dynamic customer segmentation

Customer segmentation is an important requirement for improved customer relationship management (CRM). Data mining systems have been developed successfully for customer segmentation, e.g. [23]. These systems are, however, static in the sense that they do not detect changes of customer behavior. Here we show the potential of dynamic data mining for advanced customer segmentation using the data set from the CoIL challenge 2000.

The task in CoIL challenge 2000 required to determine profiles of potential customers for a certain product. For more details on the CoIL challenge, the reader is referred to www.dcs.napier.ac.uk/coil/.

Description of the data set and initial segmentation: The original data set contains 5822 customers, each one described by 86 features. Previous studies have revealed the following five features

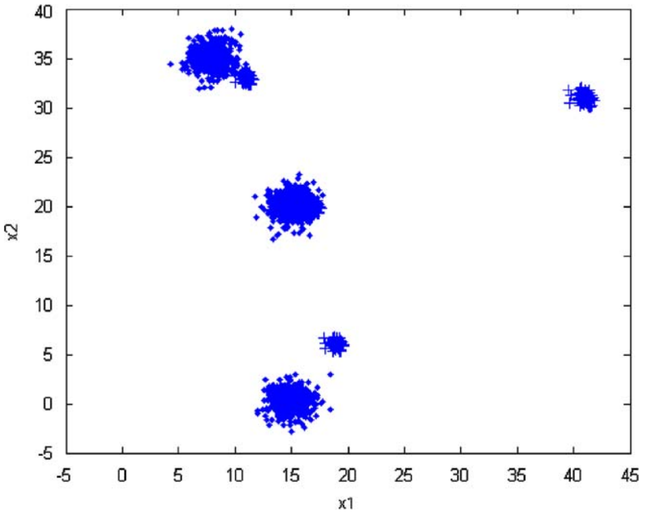

Fig. 6. Data set including new objects from cycle 3.

Table 4
Cluster solution in cycle 3

Class	Variables		Number of	Class	Class	Class
	$X1$	$X2$	objects	moved	created	eliminated
1	16.16	2.44	900	1	0	0
2	—	—	0	0	0	0
3	9.09	35.00	900	1	0	0
4	16.15	20.33	900	1	0	0
5	40.16	30.16	500	1	0	0
Total			3200			

providing a good identification of customers [12]:

F1: PPERSAUT: Contribution car policies.
F2: PBRAND: Contribution fire policies.
F3: APERSAUT: Number of car policies.
F4: MINKGEM: Average income.
F5: MOSHOOFD: Customer main type.

Applying fuzzy c-means with $c = 4$ classes we obtained a solution represented in Table 5.

Application of our methodology: In order to apply our methodology to the given static data set, we simulated dynamic behavior in the following way. Based on the 4-class solution, shown above we selected data from 2 of the four classes in order to determine an initial solution (Table 6). Then we add in each cycle a subset of the remaining data and apply our methodology using the following

Table 5
Class centers for solution with $c = 4$ classes

Class	F1	F2	F3	F4	F5	Number of objects
1	0.00	1.72	0.00	3.47	7.98	1608
2	5.80	2.04	1.10	4.30	2.62	1288
3	5.82	1.94	1.10	3.48	7.98	1689
4	0.00	1.62	0.00	4.18	2.77	1237
Total						5822

Table 6
Class centers of initial solution

Class	F1	F2	F3	F4	F5	Number of objects
1	0.02	1.69	0.00	3.41	7.88	524
2	0.05	1.68	0.01	4.22	2.66	476
Total						1000

Table 7
Class centers of solution in cycle 1

Class	F1	F2	F3	F4	F5	Number of objects	Changes
Situation: *creation of a new class and movement of classes*							
1	0.01	1.66	0.00	3.45	7.89	1018	Movement
2	0.03	1.62	0.00	4.18	2.62	832	Movement
3	5.76	2.01	1.10	4.24	2.66	1000	Creation
Total						2850	

parameters:

- *Initial solution*: $c = 2$ classes, fuzzifier $m = 2$.
- *Step* I: in order to identify objects that represent changes we use $\alpha = 0.05$.
- With $\beta = 0.2$, we create a new class if the number of objects representing changes is larger than 20% of the total number of new objects.
- In Step V, we eliminate a class if it did not receive new objects for $T = 2$ periods.

In the first cycle a class has been created and in cycle 3 the same class has been eliminated while the other classes have been moved in the feature space (Tables 7–9).

Each cycle of the proposed methodology provides information on the changes that occurred in the respective universe (here: customer base). This enables an analyst to draw conclusions on changing customer behavior, which could be important for appropriate marketing decisions.

Table 8
Class centers of solution in cycle 2

Class	F1	F2	F3	F4	F5	Number of objects	Changes
Situation: *movement of classes*							
1	0.01	1.66	0.00	3.45	7.92	1216	Movement
2	0.03	1.62	0.00	4.18	2.62	832	—
3	5.76	2.01	1.10	4.24	2.66	1000	—
Total						3048	

Table 9
Class centers of solution in cycle 3

Class	F1	F2	F3	F4	F5	Number of objects	Changes
Situation: *movement of classes and elimination of class 3*							
1	0.01	1.66	0.00	3.44	7.90	1633	Movement
2	0.03	1.60	0.01	4.19	2.62	1057	Movement
3						0	Elimination
Total						2690	

5.2. *Traffic state identification by dynamic data mining*

Due to increased traffic in many cities and on major highways it has become necessary to apply transport telematics techniques in order to improve the management of traffic systems as well as to introduce new services for car drivers into the market.

Traffic control centers (TCC) are potential end-users for such improved traffic management systems. Several TCCs already apply successfully knowledge-based systems [19] and data mining for traffic state determination [14] as well as traffic data analysis [15].

The systems mentioned above are, however, static in the sense that they do not identify changes in traffic states. This is where we propose the use of dynamic data mining.

We used traffic data, which has been collected between June 21, 1999 and October 30, 2000 from a German highway [3]. We analyzed the traffic behavior of the 66 Mondays in this period. According to experts in traffic management, Mondays typically show a different pattern compared to other days. The following figure illustrates the number of vehicles (car density) passing one particular sensor. The x-axis represents time (0–24 h) and the y-axis represents number of vehicles per hour.

Raw data are collected in intervals of 5 min. Applying Fourier-transform to the original time series and representing it using the first seven coefficients of the Fourier-transform has led to the approximation shown as smooth line in Fig. 7 [3].

We also had sensor data regarding the speed of the vehicles in the same format, i.e. each 5 min a measurement. These time series have also been processed using Fourier-transform. We used the

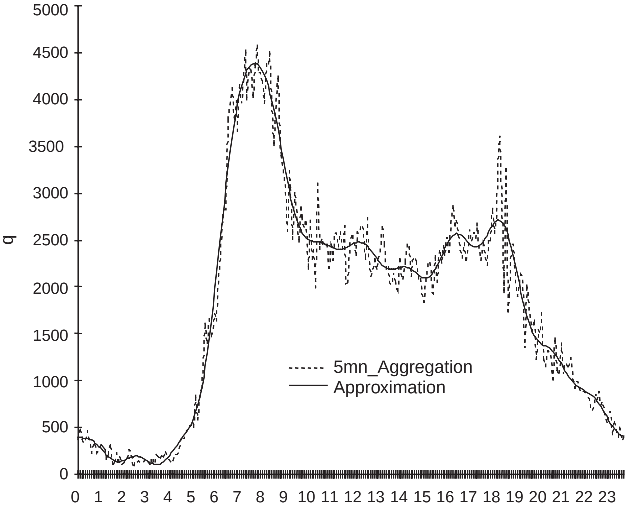

Fig. 7. Raw data and approximation (smooth line) for car density.

resulting 14 features (Fourier coefficients) in order to describe the objects (traffic state for a particular day at a particular sensor) we worked with.

From the initial data set, we used the first 33 Mondays in order to determine the initial solution with three traffic states. Then we updated this solution each month, i.e. adding four new objects (four new Mondays) in each cycle leading to eight cycles. We used the following parameters in the proposed methodology as described in Section 3:

Step I: (detect changes): $\alpha = 0.05$.

Step II: (create classes): $\beta = 0.25$.

Step V: (eliminate classes): $T = 4$ cycles.

Fig. 8 shows the approximations corresponding to the class centers of the initial structure and after eight cycles (8 months). During the first seven cycles the number of classes did not change, class centers just moved slightly. The unity of the y-axis is 100 cars/h.

As can be seen, one class has been eliminated in cycle 8. Revising the objects available and consulting the experts this change could be confirmed for the last period analyzed (October 2000).

6. Conclusions and future perspectives

We presented a methodology for dynamic data mining based on fuzzy clustering, which allows updates of the underlying classifier. We used fuzzy c-means but any other fuzzy or possibilistic clustering technique can be used as well. Changes that we have studied are movement, creation, and elimination of classes and any of their combination.

We applied our methodology for dynamic customer segmentation and dynamic traffic state identification using real-world data. The results provide updated class structures and—even more important—potential insights a user gets by analyzing changes in his/her application domain.

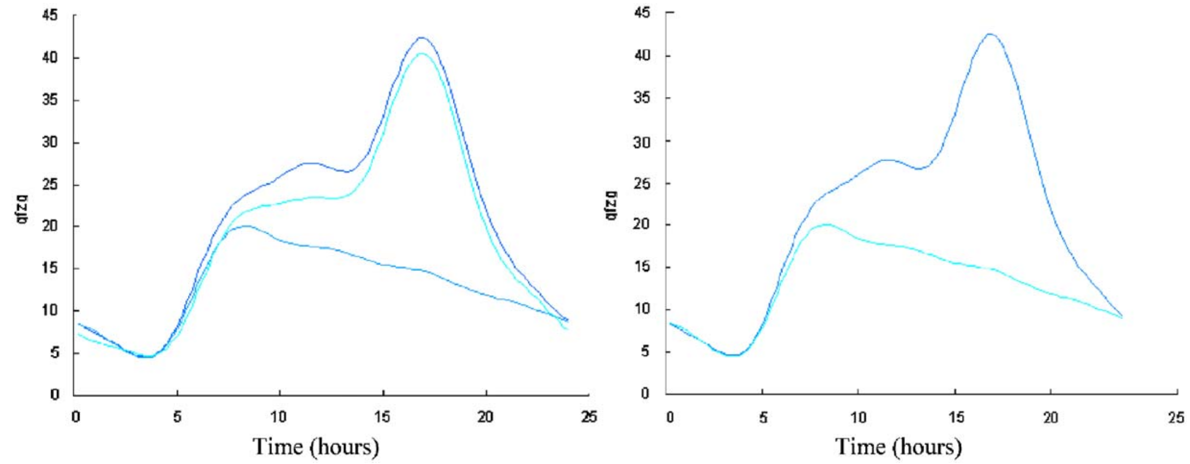

Fig. 8. Initial representation of class centers (left) and representation of class centers after eight cycles (right).

More work is necessary in order to understand better the potentials and limitations of dynamic data mining using clustering techniques.

The parameters of our methodology need further investigation. For the moment we assume to have these parameters as input, which can most probably be determined by an experienced user in a given application domain. An automatic adaptation of these parameters by employing an appropriate learning technique would be an interesting additional feature. Applying the proposed methodology in order to solve different problems would also give further information regarding future research.

Acknowledgements

This project was supported by Conicyt (FONDECYT Project No. 1000865) and by the Nucleus Millennium Science on Complex Engineering Systems.

References

[1] J.-F. Angers, Curves comparison using wavelet, Working Paper CRM-2833, Départment de mathématiques et de statistique, Université de Montréal, Montreal, Quebec, February 2002.

[2] C.M. Antunes, A.L. Oliveira, Temporal data mining: an overview, Workshop on Temporal Data Mining (KDD2001), San Francisco, September 2001.

[3] M. Bastian, H. Kirschfink, R. Weber, TRIP: automatic traffic state identification and prediction as basis for improved traffic management services, Proc. of the Second Workshop on Information Technology, Cooperative Research between Chile and Germany, 15–17 January 2001, Berlin, Germany.

[4] J.C. Bezdek, J. Keller, R. Krishnapuram, N.R. Pal, Fuzzy Models and Algorithms for Pattern Recognition and Image Processing, Kluwer, Boston, London, Dordrecht, 1999.

[5] M. Black, R.J. Hickey, Maintaining the performance of a learned classifier under concept drift, Intell. Data Anal. 3 (6) (1999) 453–474.

[6] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, 2nd Edition, Wiley, New York, Chichester, 2001.

[7] J. Espinoza, R. Weber, Desarrollo de una metodología para la actualización de una estructura de clases basada en métodos de Agrupamiento, XI CLAIO Concepción, Chile, 2002 (in Spanish).

[8] A. Famili, W.-M. Shen, R. Weber, E. Simoudis, Data preprocessing and intelligent data analysis, Intell. Data Anal. 1 (1) (1997) 3–23.

[9] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, San Francisco, 2001.

[10] A. Joentgen, L. Mikenina, R. Weber, H.-J. Zimmermann, Dynamic fuzzy data analysis based on similarity between functions, Fuzzy Sets and Systems 105 (1) (1999) 81–90.

[11] G. Karypis, E.-H. Han, V. Kumar, Chameleon: a hierarchical clustering algorithms using dynamic modeling, IEEE Comput. 32 (8) (1999) 68–75 (special issue on data analysis and mining).

[12] U. Kaymak, M. Setnes, Target selection based on fuzzy clustering: a volume prototype approach to CoIL Challenge, 2000, http://www.liacs.nl/∼putten/library/cc2000/report2.html.

[13] E.J. Keogh, M.J. Pazzani, Dynamic time warping with higher order features, in: First SIAM Internat. Conf. on Data Mining (SDM), Chicago, USA, 2001.

[14] H. Kirschfink, M. Boero, J. Hernandez, Intelligent traffic management models, Seventh World Congress on Intelligent Transport Systems, Turin, 6–9 November 2000.

[15] H. Kirschfink, R. Weber, Using fuzzy tools in traffic data analysis, Proc. of the Second Meeting of the EURO Working Group on Urban Traffic and Transportation, Paris, France, 15–17 September 1993, pp. 349–356.

[16] C.-Y. Lee, E.K. Antonsson, Dynamic partitional clustering using evolution strategies, Proc. of the Third Asia Pacific Conf. on Simulated Evolution and Learning, Nagoya, Japan, 25–27 October 2000.

[17] C. Li, G. Biswas, M. Dale, P. Dale, Matryoshka: a HMM based temporal data clustering methodology for modeling system dynamics, Intell. Data Anal. Internat. J. 6 (3) (2002) 281–308.

[18] R.-P. Li, M. Mukaidono, A maximum-entropy approach to fuzzy clustering, in: Proc. of the Internat. Joint Conf. 4th IEEE Internat. Conf. Fuzzy/2nd Internat. Fuzzy Engineering Symp. (FUZZ/IEEE-IFES), Yokohama, Japan, March 1995, pp. 2227–2232.

[19] M. Molina, J. Hernández, J. Cuena, A structure of problem—solving methods for real—time decision support in traffic control, Internat. J. Hum. Comput. Stud. (49) (1998) 577–600 (special issue on problem solving methods).

[20] V. Raghavan, A. Hafez, Dynamic data mining, in: R. Loganantharaj, G. Palm, M. Ali (Eds.), Intelligent Problem Solving—Methodologies and Approaches: Proc. of Thirteenth International Conference on Industrial Engineering Applications of AI & Expert Systems, Springer, New York, 2000, pp. 220–229.

[21] J. Strackeljan, R. Weber, Quality control and maintenance, in: H.-J. Zimmermann (Ed.), D. Dubois, H. Prade (Series Eds.), Practical Applications of Fuzzy Technologies, The Handbooks of Fuzzy Sets Series, vol. 7, Kluwer Academic Publishers, Boston, London, Dordrecht, 1999, pp. 161–184.

[22] P.E. Utgoff, Decision tree induction based on efficient tree restructuring, Mach. Learning 29 (1) (1997) 5–44.

[23] R. Weber, Customer segmentation for banks and insurance groups with fuzzy clustering techniques, in: J.F. Baldwin (Ed.), Fuzzy Logic, Wiley, Chichester, 1996, pp. 187–196.

[24] M.P. Windham, Cluster validity for fuzzy clustering algorithms, Fuzzy Sets and Systems 5 (1981) 177–185.

[25] H.-J. Zimmermann, Fuzzy Set Theory—and Its Applications, 4th Edition, Kluwer Academic Publishers, Boston, Dordrecht, London, 2001.