*Programming and Computer Software, Vol. 29, No. 4, 2003, pp. 228–237. Translated from Programmirovanie, Vol. 29, No. 4, 2003.*


# Outlier Detection Algorithms in Data Mining Systems

**M. I. Petrovskiy**

*Department of Computational Mathematics and Cybernetics, Moscow State University, Vorob'evy gory, Moscow, 119992 Russia*
*e-mail: michael@cs.msu.su*

Received February 19, 2003

**Abstract**—The paper discusses outlier detection algorithms used in data mining systems. Basic approaches currently used for solving this problem are considered, and their advantages and disadvantages are discussed. A new outlier detection algorithm is suggested. It is based on methods of fuzzy set theory and the use of kernel functions and possesses a number of advantages compared to the existing methods. The performance of the algorithm suggested is studied by the example of the applied problem of anomaly detection arising in computer protection systems, the so-called intrusion detection systems.

## 1. INTRODUCTION

With the development of information technologies, the number of databases, as well as their dimension and complexity, grow rapidly, resulting in the necessity of automated analysis of great amount of heterogeneous structured information. For this purposes, data mining systems are used. The goal of these systems is to reveal hidden dependences in databases [1]. The analysis results are then used for making a decision by a human or program, such that the quality of the decision made evidently depends on the quality of the data mining.

One of the basic problems of data mining (along with classification, prediction, clustering, and association rules mining problems) is that of the outlier detection [1–3]. The outlier detection is searching for objects in the database that do not obey laws valid for the major part of the data. The identification of an object as an outlier is affected by various factors, many of which are of interest for practical applications. For example, an unusual flow of network packages, revealed by analyzing the system log, may be classified as an outlier, because it may be a virus attack or an attempt of an intrusion. Another example is automatic systems for preventing fraudulent use of credit cards. These systems detect unusual transactions and may block such transactions on earlier stages, preventing, thus, large losses. The detection of an object–outlier may be an evidence that there appeared new tendencies in data. For example, a data mining system can detect changes in the market situation earlier than a human expert.

The outlier detection problem is similar to the classification problem. A specific feature of the former, however, is that the great majority of the database objects being analyzed are not outliers. Moreover, in many cases, it is not a priori known what objects are outliers.

In this work, we consider basic approaches used currently in data mining systems for solving the outlier detection problem. Methods based on kernel functions are considered in more detail, and their basic advantages and disadvantages are discussed. A new algorithm for detecting outliers is suggested, which possesses a number of advantages compared to the existing methods. It makes use of kernel functions and relies on methods of fuzzy set theory. The performance of the suggested algorithm is examined by the example of the applied problem of anomaly detection, which arises in computer protection systems, the so-called intrusion detection systems [4, 5].

## 2. STATISTICAL OUTLIER DETECTION METHODS

A traditional approach to solving the outlier detection problem is based on the construction of a probabilistic data model and the use of mathematical methods of applied statistics and probability theory. A probabilistic model can be either a priori given or automatically constructed by given data. Having constructed the probabilistic model, one sets the problem of determining whether a particular object of the data belongs to the probabilistic model or it was generated in accordance with some other distribution law. If the object does not suit the probabilistic model, it is considered to be an outlier. Probabilistic models are constructed with the use of standard probability distributions and their combinations. Sometimes, models include unknown parameters, which are determined in the course of data mining. Along with a priori given probability distributions, there exist algorithms for estimating probability distributions by empirical data.

### *2.1. Tests*

The probabilistic model is not sufficient for detecting outliers. A procedure that determines whether a particular object is an outlier is required. Such a procedure

0361-7688/03/2904-0228$25.00 

is referred to as a test. A standard test consists in the verification of the basic hypothesis (null hypothesis). The basis hypothesis is a statement that an object fits the probabilistic model of the system, i.e., has been generated by a given distribution law. If there is an alternative hypothesis, such that either the basic or alternative hypothesis is true, the problem of the verification of the null hypothesis is solved by standard methods of probability theory and mathematical statistics. If there is no alternative hypothesis, the verification is more complicated [1].

### *2.2. SmartSifter*

Another interesting approach to detecting outliers by statistical methods is implemented in the SmartSifter algorithm [3]. The basic idea of this algorithm is to construct a probabilistic data model based on observations. In this case, only the model, rather than the entire dataset, is stored. The objects are processed successively, and the model learns while processing each data object. A data object is considered to be an outlier if the model changes considerably after processing it. For these purposes, a special metrics, the outlier factor, is introduced to measure changes in the probabilistic model after adding a new element.

### *2.3. Regression Analysis*

Methods for detecting outliers based on the regression analysis are also classified among statistical methods. The regression analysis problem consists in finding a dependence of one random variable (or a group of variables) $Y$ on another variable (or a group of variables) $X$. Specifically, the problem is formulated as that of examining the conditional probability distribution $Y|X$. In the regression methods for the outlier analysis, two approaches are distinguished. In the framework of the first approach, the regression model is constructed with the use of all data; then, the objects with the greatest error are successively, or simultaneously, excluded from the model. This approach is called a reverse search. The second approach consists in constructing a model based on a part of data and, then, adding new objects followed by the reconstruction of the model. Such a method is referred to as a direct search [6]. Then, the model is extended through addition of most appropriate objects, which are the objects with the least deviations from the model constructed. The objects added to the model in the last turn are considered to be outliers. Basic disadvantages of the regression methods are that they greatly depend on the assumption about the error distribution and need a priori partition of variables into independent and dependent ones.

### *2.4. Replicator Neural Networks*

In the work [7], the outlier detection problem is viewed from a standpoint of regression analysis and replicator neural networks. The direct-propagation neural network suggested in [7] has three inner layers, with the number of nodes in the middle layer being limited. The number of the network inputs coincides with the number of its outputs and is equal to the number of attributes of the problem being examined. Thus, such a network approximates the function $f(x) = x$.

The neural network learns from the entire set of data and, then, is applied to this dataset once more. For each data object, the total error of recovering the object by the network is calculated. The objects with maximum errors are considered to be outliers.

### *2.5. Advantages and Disadvantages of the Statistical Approach*

Statistical methods possess a number of undoubted advantages. Firstly, they are mathematically justified. For example, the verification of competing hypotheses is a conventional problem of mathematical statistics, which can be applied to statistical models used and, in particular, to the detection of outliers. Secondly, if a probabilistic model is given, statistical methods are very efficient and make it possible to reveal the meaning of the outliers found. Thirdly, after constructing the model, the data on which the model is based are not required. It is sufficient to store the minimal amount of information that describes the model.

However, statistical models have a number of serious disadvantages, which make their use in data mining systems inconvenient. First, they require either construction of a probabilistic data model based on empirical data, which is a rather complicated computational task, or a priori knowledge of the distribution laws. Even if the model is parametrized, complex computational procedures for finding these parameters are needed. Moreover, it is not guaranteed that the data being examined match the assumed distribution law if there is no estimate of the distribution density based on the empirical data. Note also that the construction of tests for hypothesis verification in the case of complex combinations of distributions is a nontrivial task.

## 3. DISTANCE-BASED APPROACHES

Currently, so-called distance-based methods for outlier detection, which are based on the calculation of distances between objects of the database and have a clear geometric interpretation, are most popular. These methods, as a rule, calculate a function $F : x \longrightarrow R$, an outlier factor, which quantitatively characterizes an outlier. A specific feature of the distance-based approaches is that the function $F$ depends on the distance between the given object and other objects from the dataset being analyzed.

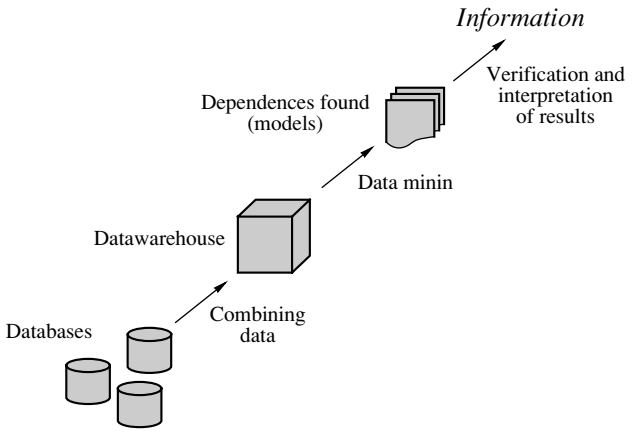


**Fig. 1.** Data mining process.

### *3.1. Method DB(p, D)*

The basic distance-based approach is that implemented in the *DB*(*p*, *D*) method. This method has been suggested in the work [2]. A more detailed discussions of the problem statement, implementation algorithms, and applications can be found in [8, 9].

The *DB*(*p*, *D*) method is based on the following definition of an outlier. An object *o* is an outlier if at least the *p*th fraction of all objects of the database are at a distance greater than *D* from the given object *o*. Various algorithms for this method have been suggested, which are designed for different models of data storage and datasets of different dimensions [1].

The index-based algorithm, for each object, calculates the number of objects belonging to the *D*-neighborhood of the object (i.e., objects located at a distance not exceeding *D*). To find the neighbors, an a priori constructed index is used. The complexity of the algorithm is quadratic.

The nested-loop algorithm is based on partitioning the entire set of objects into blocks, such that, on each step, distances between objects belonging to two blocks only are calculated. This algorithm also has quadratic complexity; however, it does not require preliminary construction of the index, which is a time-consuming operation.

In the cell-based algorithm, the complexity of examining all pairs of objects is reduced through a preliminary partition of the space into cells and construction of estimates for distances between the objects. This algorithm is discussed in detail in the work [3], where it is shown that three passes over the dataset are sufficient for constructing the desired partition. Note that the nested-loop algorithm requires not less than $m - 2$ passes, where $m$ is the number of blocks in the dataset.

### *3.2. The k-Nearest Neighbors Method*

The basic disadvantages of the *DB*(*p*, *D*) method are its globality and the necessity of specifying the values of *p* and *D* in advance. The globality property reflects the fact that the data being analyzed are viewed as a whole; therefore, the algorithm detects outliers only among objects belonging to the boundary of the dataset. As shown in [4], the *DB*(*p*, *D*) method often ignores outliers that do not belong to the boundary. The necessity in the a priori specification of the parameters *p* and *D* makes the method difficult to use, since this requires a priori hypotheses about the data to be analyzed.

The *k*-nearest neighbors method *kNN* has been suggested in [10]. It is based on the notions of the *k*-neighborhood and *k*-distance. The *k*-neighborhood of an object *o* is a set of the *k* nearest objects, and the *k*-distance of *o* is the maximum distance from *o* to the objects from its *k*-neighborhood. The *kNN* method calculates the *k*-distances for all objects and orders the objects in the descending order of these values, with the first *n* objects being considered as outliers. The *kNN* method has an advantage over the *DB*(*p*, *k*) method in that it orders the objects in terms of their exclusiveness and does not depend on the parameter *D*.

### *3.3. Local Distance-based Algorithms*

Local distance-based algorithms also use distances between objects from the dataset being analyzed. However, unlike the *DB*(*p*, *k*) method, they determine the difference of an object from its nearest neighbors rather than from the entire set of data considered as a whole. To reveal outliers, a special metric, the outlier factor (OF), is introduced. Then, a threshold value is set, and all objects whose outlier factors exceed this value are considered to be outliers. Two metrics—local outlier factor (LOF) and connectivity-based outlier factor (COF), introduced in [11] and [12], respectively—are most often used.

The local outlier factor was first introduced in the OPTICS-OF algorithm [11], which is similar to the OPTICS (ordering points to identify clustering structure) clustering algorithm. A more detailed theoretical substantiation of the LOF metrics has been suggested in [13] Without going into detail, the local outlier factor is a mean value of the ratio of the density distribution estimate in the neighborhood of the object analyzed to the distribution densities of its neighbors [13]. This method is characterized by a high rate of the outlier detection, but it also has quadratic complexity.

### *3.4. Advantages and Disadvantages of Distance-Based Methods*

The basic advantage of distance-based algorithms is that a probabilistic model is not constructed. Parame-

ters of the algorithms have clear meaning. Moreover, the distance-based algorithms can find local outliers.

Basic disadvantages of the distance-based algorithms are as follows. First, the majority of these algorithms have quadratic complexity. It should be noted that the distance-based algorithms, in fact, construct a data model in the course of data mining (like statistical ones), which takes a major part of time. The time complexity of the distance-based outlier detection algorithms can considerably be improved through the incorporation of rougher general models and the use of finer models for the most important part of the dataset [14].

The second difficulty is associated with the fact that the majority of modern information systems contain heterogeneous data of complex structure. Firstly, data objects often have discrete attributes, which makes it difficult to define distances between such objects. Secondly, the structure of the data objects may be complicated by the storage model used. For example, in a relational model, a data object may be represented by a record with nested table attributes; in a multidimensional storage model, an object may be represented as a slice of an $n$-dimensional cube [1]. The definition of a distance on sets of such objects is a nontrivial problem.

The third disadvantage of these algorithms is that they depend on a priori given parameters, such as the number of neighbors $k$, distance $D$, and the like. If these parameters change, the model, as rule, is to be constructed anew.

## 4. OUTLIER DETECTION METHODS USING KERNEL FUNCTIONS

As mentioned above, one of the basic difficulties of the distance-based methods is to define distance for heterogeneous structured data. For this purpose, methods based on kernel functions can be used [15, 16].

### *4.1. Kernel Function and Feature Space*

The basic idea of methods using kernel functions can be formulated as follows. By means of a nonlinear mapping $\varphi(x)$, the input space of objects $X$ is mapped into a Hilbert space $H$ of high (or infinite) dimension,

$$\varphi : X \longrightarrow H, \tag{1}$$

which is referred to as a feature space. The inner product in this space is given by a kernel function

$$K(x, y) = \langle \varphi(x), \varphi(y) \rangle, \tag{2}$$

where $\varphi(x)$ is the image of the object $x$ from $X$. Note that the explicit form of the mapping $\varphi$ does not matter and, as a rule, is not used; the mapping is determined implicitly by the function $K$. This is one of basic advantages of the kernel function method. First, the use of kernel functions allows us to avoid calculating and storing images $\varphi(x)$ in the explicit form, which reduces computational resources and memory required. Second, under certain conditions [15], $K$ may be viewed as a similarity measure; i.e., it is as a real symmetric positive definite function. The fact that the particular form of this function (and, thus, of the mapping $\varphi$) is not fixed makes it possible to adapt various data mining algorithms based on the calculation of inner product or distance for a wide set of problems. Such an approach is referred to as kernel trick [15]. To this end, the inner product in a given algorithm is replaced by a kernel function $K$; then, using different $K$ (in other words, applying the original algorithm in different feature spaces $H$), one can find out which kernel function gives better results. The distance can be defined, for example, as

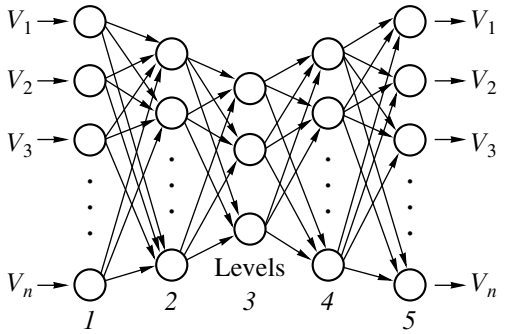

**Fig. 2.** Structure of a replicator neural network.

$$d(x, y) = \sqrt{K(x, x) - 2K(x, y) + K(y, y)}. \tag{3}$$

The selection of a kernel function and, accordingly, a feature space is determined by the applied problem; unfortunately, the question of how to choose the "best" kernel function is still open, even in the case of a particular dataset [15].

### *4.2. Kernel Function for Heterogeneous Structured Data*

Similarity measures, defined in terms of kernel functions, were originally used in pattern recognition problems. In particular, they were used in the methods of minimization of empirical risk, maximal boundary, etc., with the input space, as a rule, being real; i.e., analyzed data objects were represented by numeric vectors of fixed length. Recently, there appeared methods where kernel functions are defined for discrete structured objects, such as strings, trees, graphs, and the like. These methods are successfully used in applied problems of classifying text documents, analyzing medical data, gene chains, temporal series, and so on. In the work [17], an approach to finding kernel functions for objects of complex discrete structure in terms of kernel functions of their components is suggested. More specifically, suppose that $x \in X$ is a compound data structure consisting of $D$ components $x_1, \ldots, x_D$, where $x_d \in X_d$. Let a kernel function $K_d$ be defined on $X_d \times X_d$. Let a relation $R$ on $X_1 \times \ldots \times X_D \times X$ denote that, for a given $x$, $\vec{x} = x_1, \ldots, x_D$ is its component and there exists $R(\vec{x}, x)$.

Then $R^{-1}(x) = \{\vec{x} : R(\vec{x}, x)\}$. It is proved in [17] that, for finite $R$, the function $K$ on $X \times X$ can be defined as

$$K(x, y) = \sum_{\vec{x} \in R^{-1}(x), \vec{y} \in R^{-1}(y)} \prod_{d=1}^{D} K_d(x_d, y_d). \quad (4)$$

Based on this result, the author of this paper suggested a method [18] for determining kernel functions for problems of outlier detection in the case where the structure of a data object is described by a relational scheme with nested relations [19]. Such a representation corresponds to the case where data are stored in a relational or multidimensional database; moreover, it suits the standard of the data representation in data mining systems OLE DB for DM [20]. In this case, each object has the same set of numeric, discrete, and table attributes specified by one scheme. It is shown in [18] that the kernel function for the outlier detection problems can be calculated by the recurrence relation

$$K(x, y) = \prod_{i \in Num} e^{-q_i \frac{(x_i - y_i)^2}{\sigma_i}} \times \prod_{j \in Discr \wedge x_j \neq y_j} e^{\frac{q_j}{N_i^2}\left(\frac{1}{P(x_j)(1-P(x_j))} + \frac{1}{P(y_j)(1-P(y_j))}\right)} \times \prod_{t \in Tab}\left(\sum_{k \in Keus_t} K(x_t(k), y_t(k))\right), \quad (5)$$

where $x_i$ is the value of the $i$th nontabular attribute of the object $x$; $x_t(k)$ is the value of the $t$th table attribute of the precedent $x$ in the $k$th row; *Num* is the set of indices corresponding to the numeric attributes; *Discr* is the set of indices corresponding to the discrete attributes; *Tab* is the set of indices corresponding to the table attributes; $Keys_t$ is the set of primary keys for the $t$th table attribute; $P(x_i)$ is the probability of the value $x_i$ for the $i$th nontabular discrete attribute; $N_i$ is the cardinality of the set of values for the $i$th nontabular discrete attribute; $\sigma_i$ is the variance of the $i$th nontabular attribute; $q_i$ is the weight coefficient of the $i$th nontabular attribute, which can be interpreted as a coefficient showing significance of the given attribute (the greater $q_i$, the greater the contribution of the $i$th attribute in the kernel function).

### *4.3. Algorithms for Outlier Detection in the Feature Space*

The basic idea of algorithms for the outlier detection in the feature space is based on the assumption that the mapping $\varphi$ is topologically correct; i.e., regions of the input space $X$ with high distribution density are mapped into regions of the feature space $H$ that also have high distribution density [15, 21, 22]. Thus, objects from the original space whose images in the feature space belong to domains with low distribution density are considered to be outliers. Hence, in the feature space, conventional distance-based outlier detection algorithms, such as *DB*($p$, $q$) or *kNN*, can be used. The only difference is that distances between objects are calculated by formula (3), with the inner product being given by the function $K$. Moreover, there are more efficient outlier detection algorithms specifically designed for use in the feature space, which explicitly rely on properties of the kernel functions.

One of the algorithms of this kind is the Support Vector Clustering algorithm based on the Support Vector Machine (SVM) approach [15, 21, 22]. This algorithm evaluates a function that takes positive values in the regions of the input space $X$ where the density of the distribution generating the original set of objects is relatively high and negative values in the other regions. In nonstrict terms, the idea of the algorithm can be described as follows. Data objects from the input set $X$ are mapped by means of the kernel function into the high-dimensional feature space. In this space, the algorithm finds a sphere of minimal radius that contains a "major part" of images of the objects from the original space. The size of this "major part" is controlled by a specific parameter $\nu$. The objects whose images lie outside this sphere are considered to be outliers, and those whose images belong to the boundary of the sphere are called support vectors. The mathematical statement of this problem is as follows:

$$\min_{\xi \in \mathcal{R}^m, R \in \mathcal{R}, a \in H}\left[R^2 + \frac{1}{\nu N}\sum_{i=1}^{N}\xi_i\right], \quad \|\varphi(x_i) - a\|^2 \leq R^2 + \xi_i, \ \forall i \in [1, N], \quad (6)$$

where $R$ is radius of the sphere, $a$ is its center, $N$ is the number of objects in $X$, $0 < \nu \leq 1$ is a quantile that controls the number of the outliers, and $\xi_i$ are slack variables.

Problem (6) can be written in the dual form and solved by the Lagrange multiplier method [15, 21],

$$\min_{\beta} \sum_{i,j} \beta_i \beta_j K(x_i, x_j) - \sum_i \beta_i K(x_i, x_j), \quad 0 \leq \beta_i \leq \frac{1}{\nu N} \text{ and } \sum_i \beta_i = 1. \quad (7)$$

The decision function is given by

$$f(x) = \operatorname{sgn}\left(R^2 - \sum_{i,j}\beta_i\beta_j K(x_i, x_j) + 2\sum_i \beta_i K(x_i, x) - K(x, x)\right), \quad (8)$$

where $\beta_i$ are the Lagrange multipliers found by solving (7). They are equal to $(1/\nu N)$ for the outliers, greater than zero and less than $(1/\nu N)$ for the objects whose images belong to the boundary (support vectors), and equal to zero for other objects from $X$. It should be noted that, under certain conditions imposed on the quantile $\nu$ and on the form of the kernel function $K$ and some modification of the decision function (8), the results obtained by this algorithm coincide with those obtained by the Parzen Window method on nonparametric estimation of the distribution density [15, 21]. This implies that, in spite of the fact that the algorithm has explicit geometric interpretation, it is closely related to statistical outlier detection methods.

This algorithm shows nice results for various applied problems [15, 21, 22]. Still, from our point of view, it has several disadvantages from the standpoint of using it in data mining systems. The basic disadvantage the Support Vector Clustering algorithm is that the decision function (8), which describes the outlier factor of a data object, is discrete (takes values –1, 0, and 1, only); i.e., it is impossible to estimate how much one object is "worse" than another, except for the case when one of them is determined as an outlier, and the other is not an outlier. In addition, as shown in [15, 21], the result of the algorithm operation greatly depends on the quantile $\nu$, and it is not an easy task to appropriately select this number for a particular problem (this is the difficulty inherent in all distance-based algorithms).

One of the ways to overcome this difficulty, suggested in [21], is to run the algorithm several times, each time increasing the value of the quantile starting from $\nu = 1/N$ until an appropriate value is obtained. The termination criterion, as well as the quantile increment step, are selected heuristically. Moreover, the solution of problem (7) requires solving a quadratic programming problem of high dimension. Although efficient methods for solving this optimization problem are available (e.g., Sequential Minimal Optimization [3]), the efficiency of these methods is considerably reduced in the case where the value of the quantile changes on each step [15].

## 5. FUZZY APPROACH WITH THE USE OF KERNEL FUNCTIONS

The above-listed difficulties inherent in the SVM algorithm can be overcome by applying a fuzzy approach. The algorithm suggested in this paper combines methods of fuzzy set theory and kernel functions. Such an approach was employed in several recent works as applied to different problems solved with the use of data mining systems. In particular, in [22], a fuzzy method for the multi-class classification problem, called Fuzzy SVM [19], has been suggested; a fuzzy clustering method in the feature space has been suggested in [23].

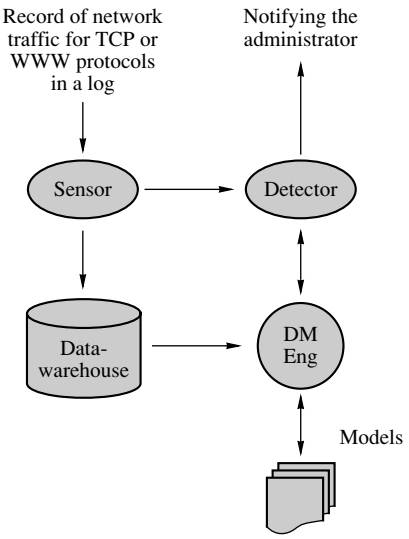


**Fig. 3.** Typical architecture of an intrusion detection system (IDS) based on data mining methods.

In the approach suggested, instead of finding a crisp sphere in the feature space $H$, we suggest to search for one common cluster containing images of all objects from the original space. In this case, the membership degree of an object image with respect to the fuzzy cluster in the feature space may be viewed as a typicalness degree of the object, i.e., a measure opposite to the outlier factor. Objects with a low typicalness degree (less than a threshold determined by the user) are considered to be outliers. It should be noted that the modification of the threshold (i.e., the modification of the outlier factor criterion) does not require model reconstruction, which is the case when the SVM or other distance-based algorithms are used. The mathematical statement of the problem in our case is as follows:

$$\min_{U \in [0,1]^N,\, a \in H} J(U, a),$$
$$J(U, a) = \sum_{i=1}^{N} (u_i)^m (\varphi(x_i) - a)^2 - \eta \sum_{i=1}^{N} (1 - u_i)^m, \quad (9)$$

where $a$ is the center of the fuzzy cluster in the feature space; $N$ is the number of objects in $X$; $U$ is the membership degree vector function; $u_i \in [0, 1]$ is the membership degree of the image $\varphi(x_i)$ with respect to the fuzzy cluster in the feature space and, accordingly, the typicalness degree of the object $x_i$; $m$ is the fuzziness degree; and $\eta$ is the distance from the cluster center, where the typicalness degree of the object is equal to 0.5. It is important to note that, unlike in traditional fuzzy clustering methods used for calculating the typicalness degree in the input space $X$, neither the center of the only (in our case) cluster, nor the values $\varphi(x_i)$ used in the functional (9), can be given in an explicit form. It can be shown, however, that $J(U, a)$ is minimized by the

following simple iteration algorithm described in terms of kernel functions.

Minimization algorithm *J(U, a)*

```
/* l – number of the iteration*/
Step 0. Initialization of U and η
Repeat
Step 1.
```

$$\langle a, a\rangle = \left(\sum_{j=1} N u_j^m \sum_{i=1}^N u_i^m K(x_i, x_j)\right) \Big/ \left(\sum_{i=1}^N u_i^m\right)^2.$$

```
Step 2. For all j from [1, N]:
```

$$\langle \varphi(x_j), a\rangle = \left(\sum_{i=1}^N u_i^m K(x_i, x_j)\right) \Big/ \left(\sum_{i=1}^N u_i^m\right).$$

```
Step 3. For all j from [1, N]:
```

$$u_j = [1 + ((\langle a, a\rangle + K(x_j, x_i) - 2\langle \varphi(x_j), a\rangle)/\eta)^{m-1}]^{-1}.$$

```
Until ||U^l – U^{l-1}|| < ε.
```

In this case, the decision function (i.e., the function that calculates the typicalness degree of a new object) is as follows:

$$u(x) = \frac{1}{1 + \left(\dfrac{(\langle a, a\rangle + K(x, x) - 2(\varphi(x), a))}{\eta}\right)^{m-1}}, \quad (10)$$

$$\langle \varphi(x), a\rangle = \left(\sum_{i=1}^N u_i^m K(x_i, x)\right) \Big/ \left(\sum_{i=1}^N u_i^m\right).$$

As for the initialization of $U$ and $\eta$, we used two following variants. The first variant is based on the use of the SVM clustering with a fixed value of the quantile (e.g., $\nu = 1/N$). It should be noted that, since the selection of the quantile value is not required, one run of the SVM algorithm is sufficient. In this case, we assume that the cluster center coincides with the center of the hypersphere in SVM, and $\eta$ is the hypersphere radius squared. Then, the typicalness degree is equal to 0.5 for the support vectors, less than 0.5 for the outliers, and greater than 0.5 for the points inside the sphere,

$$u_i^0 = \frac{1}{1 + (R^2(x_i)/R^2)^{m-1}}. \quad (11)$$

In the second (simplified) variant, two points at maximum distance from each other are found; $\eta$ is taken equal to the square of the distance between them; and elements of $U$ are taken equal each other, $u_i^0 = 1/N$; i.e., the cluster center coincides with the "center of gravity" of the images of points $x$. For such a selection of $\eta$, the typicalness degree of the objects from the learning set is always greater than 0.5.

The suggested algorithm has a number of advantages compared to the SVM clustering method. As has already been noted, its basic advantage is that the decision function is continuous, which makes it possible, first, to compare objects in terms of their typicalness degree and, second, to modify the outlier factor criterion without reconstructing the model. In addition, the suggested algorithm is considerably simpler than those (e.g., the SMO algorithm) for solving quadratic programming problems. The complexity of the algorithm itself is linear; however, the calculation of the kernel matrix requires $O(N^2)$ operations, where $N$ is the size of the learning set. Moreover, in the working version of the algorithm, a method for reducing the learning set is implemented, which makes it possible to find the outlier factor $u(x)$ for new objects $x$ without using all $N$ objects of the learning set. This method will be discussed in another paper.

## 6. APPROBATION OF THE SUGGESTED ALGORITHM ON THE NETWORK INTRUSION DETECTION PROBLEM

The computer attack detection is a field where the data mining methods are widely used. Currently, several definitions of attacks, or intrusions, are available. In [5], an intrusion into a computer system is defined as any activity that violates system integrity, confidentiality, or data accessibility; in other words, it is an attempt to break into the system or to use it without an authorization. In [24], this term is treated as an action, or a sequence of actions, that, by using vulnerability of the information system, result in the realization of a threat. Both these definitions clearly identify what kind of enemy the intrusion detection systems have to oppose. Some kinds of attacks that are most often met in practice are listed below [25]:

• remote penetration (R2L, unauthorized remote login to machine) is an intrusion into the system that allows one to control the computer attacked through the network;

• local penetration (U2R, unauthorized access to root privileges) is an intrusion by means of which the attacking side gets an access to the computer with an authorized level of privileges;

• denial of service (DoS) is an attack that results in a complete or partial termination of system functioning;

• scanning (Probe) is searching for blots in the system configuration.

The purpose of the intrusion detection system (IDS) is to determine whether the user behavior is correct or may be classified as an intrusion. The basic source of information for such a system is audit logs; however, the IDS can also gather its own statistics of system states and users' activity [4]. Usually (except for special cases), the aim of the IDS is to issue a warning rather than to do anything in response to an intrusion.

Two different approaches are used in the IDS design: misuse detection and anomaly detection. In the

first case, the places and ways of the penetration into the system are assumed to be known, and the security system has to detect attacks that follow a priori known scenarios. The advantage of this approach is that the IDS can focus on details of possible attacks and, therefore, produces few false warning messages. The opposite approach—anomaly detection—is based on the assumption that scenarios of standard user's behavior are known, such that large deviations from these scenarios are considered to be suspicious. This approach makes it possible to detect unforeseen attacks carried out by quite new scenarios.

### *6.1. Anomaly Detection as a Principle of the IDS Functioning*

It is evident that the IDS with manually specified rules for the attack detection can detect only those kinds of attacks that are known to the expert. Clearly, the automatic detection of new types of attacks is impossible in this case, and the IDSs based on the intelligent data mining systems are to be considered. The anomaly detection approach better suits the problem considered. The IDSs based on the anomaly detection principle consider all actions that differ from routine ones as intrusions. Under this approach, there is no difference whether the attack is standard or new; so long as it differs from the normal system behavior, the IDS will detect it.

Two types of algorithms for anomaly detection—supervised and unsupervised learning [22]—are implemented in the intrusion detection systems. The former suggests that the learning set for the IDS contains records of normal users' behavior, and all actions that considerably differ from those in the set are considered to be anomalies. Theoretically, this approach facilitates the construction of the recognition rules. However, the serious disadvantage of this approach is that it is difficult to construct a learning set in which all records are examples of permitted behavior. Therefore, the supervised learning requires manual verification of the training dataset, which is its basic disadvantage. The advantage of the supervised learning is that, in a number of cases, the efficiency of methods based on this approach is higher than that of methods based on the unsupervised learning.

Training sets for algorithms of the unsupervised learning contain records corresponding to both permitted behavior and attacks, with the proportion of the latter being close to that likely to be met in practice. As in the previous case, records made for a certain period of time can be used. Then, outlier detection algorithms are applied, which detect records that considerably differ from the others and classify them as intrusions. After this, the approaches described above can be applied to the original training set. As can easily be seen, the unsupervised learning is based on two following assumptions:

**Table 1**

| Detection rate (%) | DOS | U2R | R2L | Probe |
|---|---|---|---|---|
| *Ripper* | 99.5 | 84.7 | 96.7 | 97.2 |
| *kNN* | 87.9 | 37.5 | 92.7 | 50.1 |
| *SNN* | 99 | 60.2 | 91.2 | 94 |
| Suggested method | 97.7 | 97.3 | 81.2 | 99.8 |

(i) the number of records corresponding to permitted actions is considerably greater than the number of records corresponding to attacks;

(ii) records corresponding to intrusions can easily be separated from those corresponding to the normal behavior.

Note that the last assumption is fundamental for all methods of the anomaly detection used in the IDSs.

### *6.2. Verification Methods for the Algorithms Used in IDSs*

Errors of the IDSs can be classified into two groups: false positive and false negative errors [5, 22, 24]. The standard index of classification correctness—the ratio of the number of objects classified correctly to the total number of objects in the training set—is not suitable for the intrusion detection problem. The point is that, in the tests with the data distribution close to that in practice, the number of records corresponding to intrusions is relatively small. Therefore, it makes more sense to use a combination of two different IDS performance measures, the detection rate and false positive rate [26]. The detection rate is the ratio of the number of intrusions correctly detected by the IDS to the total number of all intrusions available in the training set. The false positive rate is the ratio of the number of false positive errors to the number of records corresponding to permitted users' actions.

Clearly, in order to compare different algorithms, a good test dataset (benchmark), which models situations that are likely to be met in practice and contains many records corresponding to various intrusions, is needed. Currently, for such a dataset, the KDD Cup 99 Data Set is most often used [27].

In 1998, the MIT Lincoln Lab developed a technique for evaluating intrusion detection algorithms, called DAPRA 1998 Intrusion Detection Evaluation Program. It includes a dataset simulating intrusions into a local network. One of the versions of this dataset was used as a test set at KDD Cup, the annual competition of knowledge discovery and data mining systems. In 1999, this event was devoted to the intrusion detection. The test dataset contained four gigabytes of information transmitted through a network during seven weeks. To receive these data, a typical local network, used in the US Air Force, had been modeled. In addition, this network was subjected to many various

**Table 2**

| Algorithm | Detection rate | False positive |
|---|---|---|
| Cluster | 93% | 10% |
| *KNN* outliers | 91% | 8% |
| SVM | 98% | 10% |
| Suggested method | 94% | 5.7% |

attacks. The data have been transformed to five million records corresponding to single connections.

Each record contains information about one connection. The record fields are divided into three following classes differing by the type of information contained in them:

• basic characteristics of the connection, such as protocol type, connection duration, and the number of bytes transmitted;

• information about user's actions during the connection, such as, for example, the number of unsuccessful logins;

• information about amount and behavior of the traffic during the connection.

Each record is marked as either an allowable connection or intrusion. In the latter case, the intrusion type is indicated. All four basic intrusion classes—DoS, R2L, U2R, and Probe—are presented.

The KDD Cup 99 Data Set consists of a training dataset, based on which an IDS is to construct rules for attack determination, and a verification set, in which the IDS has to detect intrusions using the rules constructed. The verification set contains about three hundred thousand records. To better model a real-life situation, the distributions in the training and verification sets are different. Moreover, the training set contains attacks of 22 types, whereas the number of attack types in the verification set is greater by 17. This is in order to test whether a system is capable of detecting attacks executed by new scenarios.

### *6.3. Results of an Experiment on the KDD Cup 99 Data Set*

The aim of the experiment was to compare results obtained on the benchmark KDD Cup 99 by means of the algorithm suggested in this paper with those obtained by other algorithms. Two groups of algorithms are considered. The first group includes three algorithms showing the best results in their domains of application, which are based on different data mining methods and implementing both misuse detection and anomaly detection approaches. These are

• the *Ripper* algorithm [2, 28], which uses association rules mining and decision tree construction methods and operates in the misuse detection mode;

• the *kNN* classifier algorithm [28], which uses the *k*-nearest neighbors classification methods and operates in the misuse detection mode; and

• the *SNN* algorithm [28], which uses clustering methods, operates in the anomaly detection mode, and is based on the unsupervised learning.

The algorithms are compared for different attack types (see Table 1).

**False positive rate**:

• *Ripper* 4.6%;

• *kNN* 4%

• *SNN* 2.5%

• The suggested algorithm 5.7%.

The results of the experiment demonstrate that the detection rate of the suggested algorithm is compared with or better than that of the best intrusion detection algorithms based of different data mining methods, although the false positive rate of the suggested algorithm is higher and the data detection rate for the remote penetration attacks (*r*2*l*) is somewhat lower.

The algorithms from the second group belong to the same class as the suggested algorithm. All these algorithms operate in the anomaly detection mode with supervised learning and use kernel functions for determining distances, although the kernel functions used in them are different from the kernel function (5) employed in the suggested algorithm. These are the SVM clustering algorithm; the modified cluster implementation of the *DB*(*p*, *q*) algorithm, called Cluster [22]; and a modified *K-NN* outlier detection algorithm. These algorithms, as well as the kernel functions used in them, are described in detail in the work [22]. Table 2 shows results related to the algorithms from this group.

The results of the experiment show that the detection rate of the suggested algorithm is about the same as that of all these algorithms (is second only to the SVM); however, the false positive rate of the suggested algorithm is sizably lower than that of all other algorithms of this class.

## 7. CONCLUSIONS

The problem of searching for outliers in databases by means of data mining systems has been considered. The brief survey of statistical and distance-based methods currently used for solving this problem has been given. Advantages and disadvantages of these methods have been outlined. The use of the kernel function methods in the outlier detection algorithms and the method for determining kernel functions for heterogeneous structured data described by a relational scheme with nested relations, which makes it possible to use these methods for a wide class of problems, have been discussed. The outlier detection algorithm SVM, which is one of the most efficient algorithms in this field, has been discussed in a more detail. Its disadvantages when using for data mining in large datasets have been noted.

The new outlier detection algorithm, which is based on the same idea as SVM and uses kernel functions and methods of fuzzy set theory, has been suggested. The suggested algorithm makes it possible to overcome a number of difficulties inherent in the SVM algorithm.

The performance of the suggested algorithms when using in computer security systems, the so-called intrusion detection systems, for solving the applied problem of anomaly detection has been studied. For this purpose, the algorithm has been tested on the benchmark KDD Cup 99, used for the evaluation of the intrusion detection algorithms. The experiments demonstrated that the suggested algorithm is highly competitive with the best algorithms used in intrusion detection systems. Moreover, in the class of the intrusion detection algorithms using kernel functions and operating in the anomaly detection mode with supervised learning, the algorithm has shown the best results.

The implementation of the suggested algorithm supports the OLE DB for DM standard; i.e., the required COM interfaces and SQL-like language are implemented. This makes it possible to easily build it in various users' applications, as well as to use data sources supporting the OLE DB standard, in particular, most of relational and multidimensional databases, spreadsheets, and structured text files.

## ACKNOWLEDGMENTS

This work was supported by the Russian Foundation for Basic Research, project no. 03-01-00745.

## REFERENCES

1. Han, J. and Kamber, M., *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2000.
2. Knorr, E.M. and Ng, R.T., Algorithms for Mining Distance-Based Outliers in Large Datasets, *Proc. 24th VLDB*, 1998.
3. Yamanishi, K, Takeichi, J., and Williams, G., On-Line Unsupervised Outlier Detection Using Finite Mixtures with Discounting Learning Algorithms, *Proc. of the Sixth ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, Boston, 2000, pp. 320–324.
4. Kemmerer, R.A. and Vigna, G., Intrusion Detection: Brief History and Survey, http://kiev-security.org.ua/box/12/-119.shtml.
5. Intrusion Detection Pages, Purdue University, 2003, http://www.cerias.purdue.edu/coast/intrusion-detection/-index.html.
6. Hadi, A.S., A New Measure of Overall Potential Influence in Linear Regression, *Computational Statistics Data Analysis*, 1992, vol. 14, pp. 1–27.
7. Hawkins, S., He, H., Williams, G., and Baxter, R., Outlier Detection Using Replicator Neural Networks, *Proc. of the Fifth Int. Conf. on Data Warehousing and Knowledge Discovery*, 2002.
8. Knorr, E.M. and Ng, R.T., Algorithms for Mining Distance-Based Outliers in Large Datasets, *Proc. 24th VLDB*, 1998.
9. Knorr, E.M., Ng, R.T., and Tucakov, V., Distance-Based Outliers: Algorithms and Applications, *VLDB J.*, 2000, vol. 8, no. 3–4, pp. 237–253.
10. Ramaswamy, S., Rastogi, R., and Shim, K., Efficient Algorithms for Mining Outliers from Large Data Sets, *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, 2000, pp. 427–438.
11. Breunig, M.M., Kriegel, H.-P., Ng, R., and Sander, J., OPTICS-OF: Identifying Local Outliers, *Proc. Conf. on Principles of Data Mining and Knowledge Discovery*, Prague, 1999.
12. Tang, J., Chen, Z., Wai-chee Fu A., and Cheung, D., *A Robust Outlier Detection Scheme for Large Data Sets*, 2001.
13. Breunig, S., Kriegel, H.-P., Ng, R., and Sander, J., LOF: Identifying Density-Based Local Outliers, *ACM SIGMOD Int. Conf. on Management of Data*, Dallas, 2000.
14. Wen Jin, Tung, A.K.H., and Han, J., Mining Top-n Local Outliers in Large Databases, *KDD*, 2001, pp. 293–298.
15. Scholkopf, B. and Smola, A.J., *Learning with Kernels*, Cambridge, London: MIT, 2002.
16. Aizerman, M.A., Braverman, E.M., and Rozonoer, L.I., *Metod potentsial'nykh funktsii v teorii obucheniya mashin* (Kernel Function Method in Machine Learning), Moscow: Nauka, 1970.
17. Haussler, D., Convolution Kernels on Discrete Structures, *Techn. Report CSD-TR-98-11 from Royal Holloway*, Univ. of London, 1999.
18. Petrovskiy, M.I., Similarity Measure for Comparing Precedents in Data Mining Systems Supporting OLEDB Standard in *Programmnye sistemy i instrumenty*, Moscow: Izdatel'skii otdel fakul'teta VMiK MGU, 2002, no. 3, pp. 33–43.
19. Levene, M. and Loizou, G., A Fully Precise Null Extended Nested Relational Algebra, *Fundamenta Informaticae*, 1993, vol. 19, pp. 303–343.
20. *OLE DB for Data Mining Specification*, Microsoft Corp., 2000, http://www.microsoft.com/data/oledb/dm.htm.
21. Ben-Hur, A., Horn, D., Siegelmann, H.T., and Vapnik, V., Support Vector Clustering, *J. Machine Learning Research*, 2001, no. 2, pp. 125–137.
22. Takuya Inoue and Shigeo Abe, Fuzzy Support Vector Machine for Pattern Classification, *Proc. of IJCNN*, 2001, pp. 1449–1455.
23. Girolami, M., Mercer Kernel Based Clustering in Feature Space, *IEEE Trans. Neural Networks*, 2001, vol. 13, no. 4, pp. 780–784.
24. Lukatskii, A.V., *Attack Detection*, St. Petersburg: BKhV-Peterburg, 2003.
25. Mell, P., Computer Attacks: What They Are and How To Defend against Them, *NIST*, Comput. Security Division, 1999.
26. Portnoy, L., Eskin, E., and Stolfo, S.J., Intrusion Detection with Unlabeled Data Using Clustering, *Proc. of ACM CSS*.
27. *MIT Lincoln Lab KDD Cup 99 Data Set*, http://www.ll.mit.edu/IST/ideval/data.
28. Kumar, V., Data Mining for Network Intrusion Detection, *NSF Workshop on Next Generation Data Mining*, 2002.