Detecting Outliers using Transduction and Statistical Testing*

Daniel Barbará
George Mason University
ISE Department, MSN 4A4
Fairfax, VA 22030
dbarbara@gmu.edu

Carlotta Domeniconi
George Mason University
ISE Department, MSN 4A4
Fairfax, VA 22030
carlotta@ise.gmu.edu

James P. Rogers
U.S. Army Engineer Research and Development Center
7701 Telegraph Road
Alexandria, VA 22315
James.P.Rogers.II@erdc.usace.army.mil

ABSTRACT

Outlier detection can uncover malicious behavior in fields like intrusion detection and fraud analysis. Although there has been a significant amount of work in outlier detection, most of the algorithms proposed in the literature are based on a particular definition of outliers (e.g., density-based), and use ad-hoc thresholds to detect them. In this paper we present a novel technique to detect outliers with respect to an existing clustering model. However, the test can also be successfully utilized to recognize outliers when the clustering information is not available. Our method is based on Transductive Confidence Machines, which have been previously proposed as a mechanism to provide individual confidence measures on classification decisions. The test uses hypothesis testing to prove or disprove whether a point is fit to be in each of the clusters of the model. We experimentally demonstrate that the test is highly robust, and produces very few misdiagnosed points, even when no clustering information is available. Furthermore, our experiments demonstrate the robustness of our method under the circumstances of data contaminated by outliers. We finally show that our technique can be successfully applied to identify outliers in a noisy data set for which no information is available (e.g., ground truth, clustering structure, etc.). As such our proposed methodology is capable of bootstrapping from a noisy data set a clean one that can be used to identify future outliers.

Categories and Subject Descriptors

I.5.3 [**Clustering**]: Algorithms

General Terms

Outliers, Anomaly detection, Transduction

*This work has been sponsored by grants from NSF:IIS-0208519, and the Army:W9132V-06-C-0012

KDD'06, August 20–23, 2006, Philadelphia, Pennsylvania, USA.

1. INTRODUCTION

Outlier detection is an important data mining task that deals with the discovery of points that are exceptional when compared with a set of observations that are considered "normal." Applications of outlier detection abound in fields such as credit fraud, criminal investigation, spatio-temporal analysis, and computer intrusions. Outlier detection can reveal points that behave "anomalously" with respect to other observations. Examining such points can reveal clues to solve the problem at hand. In other cases, the sudden appearance of a large number of outliers can point to a change in the underlying process that is generating the data. Hawkins [14] defines an outlier as "an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism." This suggests the possibility of detecting outliers by knowing the "mechanism" by which the normal observations were generated and testing points for "membership" to this mechanism. Indeed, that is the path that early work in outlier detection followed (in the statistical community; see [19] for a comprehensive review): postulate a model for the probability distribution of normal points (e.g., a Gaussian model), and compute the likelihood of a point being generated by the postulated model. Unfortunately, coming up with the right model is, at best, as difficult as the original problem of finding outliers. This approach can be seen as *inducing* a model over the normal data and using it to test points.

Recently, the field of statistical learning theory [31] has developed alternatives to *induction*: instead of using all the available points to induce a model, one can use the data (usually a small subset of it) to estimate unknown properties of points that one wants to test (e.g., membership to a class). This powerful idea leads to elegant algorithms that use standard statistical tests to compute the confidence on the estimation. Using transduction, researchers have built Transductive Confidence Machines (TCM) (see [9]) which are able to estimate the unknown class of a point and attach confidence to the estimate. The transductive reliability estimation process has its theoretical foundations in the algorithmic theory of randomness developed by Kolmogorov [20]. Unlike traditional methods in machine learning, transduction can offer measures of reliability to individual examples, and uses very broad assumptions (it only assumes that the data points are independent and generated by the same stochastic mechanism). These properties make transduction

an ideal mechanism to detect outliers, even though it has never been used before for that purpose. In this paper, we use the ideas of TCM to design a test that determines if a point is an outlier and attach a confidence to the estimation. We assume first that we want to compare new points to a base of "normal" observations, which have been organized into a clustering model.

A method that bases its outlier decisions on a clustering model (i.e., a non-parametric model) brings about a series of obvious questions. What if the clustering model is not perfect, or the clustering information is simply not available? How does one get perfectly "normal" data, i.e. free of outliers, in the first place? And, lastly, if the data available cannot be guaranteed to be outlier-free, how does this affect the outlier discovery technique? We have found answers to each of these questions via experimentation. We show that our technique can still effectively find outliers when we get rid of the clustering model. In addition, we show that our technique can be used to clean data which contain outliers. And finally, we show that even if the cleaning process leaves some outliers within the presumed "normal" set, the technique can still be effective in detecting further outliers.

2. BACKGROUND: TCM

As we have pointed out in the introduction, our method makes use of the concept of *transduction* to find outliers with respect to a series of pre-defined clusters. A simple interpretation of transduction [31] is that unknown estimates for individual points of interest can be made directly from the training data, as opposed to using *induction* to infer a general rule for them. In our case, the estimates we are interested in learning are the likelihood that a point belongs to a given cluster in the current clustering model. In that sense, our "training" examples are the points already clustered. Transduction has been previously used to offer confidence measures for the decision of labelling a point as belonging to a set of pre-defined classes (see [24, 33, 9]). TCM [9] introduced the computation of the confidence using Algorithmic Randomness Theory [20]. (The first proposed application of Algorithmic Randomness Theory to machine learning problems, however, corresponds to Vovk et al. [32].) The confidence measure used in TCM is based upon universal tests for randomness, or their approximation. A Martin-Lof randomness deficiency test [20] based on such tests is a universal version of the standard p-value notion, commonly used in statistics. Martin-Lof proved that there exists a universal test for randomness smaller than any other test up to a multiplicative constant. Unfortunately, universal tests are not computable, and have to be approximated using non-universal tests called p-values. In the literature of significance testing, the p-value is defined as the probability of observing a point in the sample space that can be considered more extreme than a sample of data. This p-value serves as a measure of how well the data supports or not a null hypothesis. The smaller the p-value, the greater the evidence against the null hypothesis. Users of transduction as a test of confidence have approximated a universal test for randomness (which is in its general form, non-computable) by using a p-value function called *strangeness measure* [9] (or non-conformity score [32]). In truth, there is more than a single definition of strangeness measure, and in general, its definition depends on the base model used to construct the TCM. The general idea is that the strangeness measure corresponds to the uncertainty of the point being measured with respect to all the other labelled examples of a class: the higher the strangeness measure, the higher the uncertainty. We show in the next section how to adapt the definition of the strangeness function of TCMs to our context of finding estimates for points to be outliers with respect to clustering models. In doing so, we treat points in a given cluster of the model as belonging to the same "label" or class, even though such class has not been defined. We claim that this is possible since points in the same cluster exhibit enough similarity (as measured by the underlying clustering algorithm) among themselves to be commonly labelled as belonging to a class.

We employ the definition of [24] for strangeness: the strangeness α_i of a point i with respect to a class y is given in Equation 1, where D_i^y is the sequence of distances between point i and points in the class y, D_{ij}^y being the j-th shortest distance. At the same time, D_i^{-y} represents the sequence of distances between i and points in other classes (different from y), D_{ij}^{-y} being the j-th shortest distance. K is a parameter that indicates the number of nearest neighbors we want to consider.

$$\alpha_{iy} = \frac{\sum_{j=1}^{K} D_{ij}^{y}}{\sum_{j=1}^{K} D_{ij}^{-y}} \quad (1)$$

Notice that this definition of strangeness is based on the nearest neighbors to the point (in and out of the particular cluster we are using to test the point against), and thus borrowed from the TCM-kNN (transduction confidence machine that uses k-nearest neighbors). Alternative definitions of strangeness are possible, however, for the purposes of this paper, we will limit ourselves to this definition. In particular, we use the Euclidean distance to compute the distance between pairs of points (as done in the distance based method [18] against which we compare ours).

The strangeness being the ratio of the sum of the K smallest distances from the same class we propose to put i in, to the sum of the K smallest distances from other clusters, measures how "strange" the point in question is with respect to the cluster we want to put it in. The larger the value of α, the stranger the point.

The p-value is calculated as shown in Equation 2, where # represents the cardinality of the set given as an argument.

$$t(z_1, z_2, \cdots, z_n) = \frac{\#\{i = 1, \cdots, n : \alpha_{iy} \geq \alpha_{ny}\}}{n}. \quad (2)$$

If z_n is the point in question, the function $t()$ will measure the probability of having points already in the class with strangeness greater than or equal to that of z_n. In general, a p-value is the maximum probability under the null hypothesis of the test statistic assuming a value equal to the observed outcome or a value just as extreme or more extreme (with respect to the direction indicated by alternative hypothesis) than the observed outcome. So the smaller the p-value is, the smaller is the chance that the test statistic could have assumed a value as incompatible with the null hypothesis if the null hypothesis ("class y is a good fit for point i.") is true. The algorithm TCM-kNN attempts to place a new point in each class of the problem. While do-

Let there be m training examples
For $i = 1$ to m do
    Calculate D_i^y and D_i^{-y}
    Calculate α value for each example
Let there be c classes and u be the new example
(to be classified)
For $j = 1$ to c do
    For every training example t, classified as j do
        If $D_{tk}^j > dist(t,u)$
        (if the largest distance to one of the k-neighbors
        in class j for t is bigger than
        the distance of t to the new example u)
            Recalculate the α value for t (with u in j)
    For every training example t, classified as non-j do
        If $D_{tk}^{-j} > dist(t,u)$
        (if the largest distance to one of the k-neighbors
            outside class j for t is bigger than
        the distance of t to the new example u)
            Recalculate the α value for t (with u in j)
    Calculate α value for u
    Compute the p-value of the point for class j
Predict the class with the largest p-value for u
Output as confidence one minus the 2nd p-value

Figure 1: The TCM-kNN algorithm

ing that, it may force the updating of some of the α values for the training examples (concretely, this happens whenever the distance between the training example and the new point is less than the largest of the k distances that are used to compute the α). It then computes one p-value for each of the attempts (i.e., for each class placement). It then predicts that the point belongs to the class with the largest p-value, with a confidence equal to the complement of the second p-value. The algorithm for TCM-kNN is shown in Figure 1.

3. OUR METHOD

Now we adapt the ideas of TCM-KNN for our purposes of determining if a point is an outlier with respect to a clustering model. We can use the ideas of TCM, by computing the strangeness of any point with respect to a cluster y - instead of a class y- by simply considering points in y as the training examples of a class y. However, we need to realize a fundamental difference between our problem and that solved by TCM. In TCM we are always sure that the point we are examining *belongs* to one of the classes. In our problem, we are trying to determine if the point in question is an outlier, and hence does not belong to any of the clusters that we have. This has consequences in the outcome of the calculation of α, if we follow the Equation we presented in Section 2. The α computed for an outlier (a point that does not belong to any of the clusters) will be the ratio between two large numbers (the distances from the point in question to those in any of the clusters are large). In some cases, this ratio will be small enough to be comparable to the α values for points already in the cluster, leading to false negatives. (This is indeed the case in practice, as our experience has shown.) Therefore, we propose to use a modified definition

Given a point u under consideration:
    1. Compute the p-value of i with respect
    to clusters $1, \cdots, c$.
    2. Sort the p-value list in descending order.
    3. Call p_{max} the highest p-value, and
    p_{next} the next in the list.
    4. If $p_{max} \leq \tau$, reject all
    the null hypotheses H_0^y, for $y = 1, \cdots, c$,
    and therefore declare i an outlier
    with confidence $1 - \tau$.
    5. Else, reject all the alternative hypotheses
    (the point belongs to a cluster in the model).

Figure 2: StrOUD: the algorithm to compute the fitness of a point i with respect to the existing clusters

of α, as follows: the strangeness α_i of a point i with respect to a cluster y is defined as shown in Equation 3.

$$\alpha_{iy} = \sum_{j=1}^{K} D_{ij}^y \tag{3}$$

This new definition will make the strangeness value of a point far away from the cluster considerably larger than the strangeness of points already inside the cluster. This definition has been employed by [1] (see Section 5) as a measure of isolation. Using the α values, we can compute a series of p-values for the new point, one for each cluster $y = 1, \cdots, c$ (where c is the number of clusters). We call the highest p-value in this series p_{max}. This gives us a way of testing the fitness of point u, by testing the null hypothesis H_0^y as "u is fit to be in cluster y." Thus, the alternative hypothesis H_1^y is "u is ill-fit to be in cluster y." Selecting a confidence level $1 - \tau$ (usually 95 %), we can test if $p_{max} \leq \tau$, in which case, we reject all the null hypotheses and declare the point an outlier. Otherwise, we reject all the alternative hypotheses. The pseudo-code of the algorithm, which we call StrOUD (from Strangeness based OUtlier Detection algorithm) is shown in Figure 2. It contains a rule to accept or reject alternative hypotheses for the fitness of u in the existing clusters. Notice that we do not use the algorithm to decide the placement of the point in the clusters; that decision is left to the clustering algorithm of choice).

The operation of changing the α values for some of the examples in the clusters (whenever the new point is closer to the example than at least one of the example's k nearest neighbors), is potentially a costly one. Instead of just keeping the α values for all the points in the clusters, it requires maintaining the information of the k-nearest neighbors for each point, so when a new point is under consideration, these lists can be examined and the distances to the k-nearest neighbors compared to the distance to the new point. However, since we are only interested in diagnosing whether the new point is an outlier or not, we can do away with all this information and drop that operation all together, working instead with the original α values of the clustered points. In doing so, we only risk working with some value of α that is larger than it really ought to be. To understand this, consider a point u to be tested. Suppose

u is close to at least one point i in cluster y, so that there exist a point j in y, among the k-nearest neighbors of i, that satisfies $d(i,j) > d(i,u)$. In this case, u will replace j among the nearest neighbors of i when tested in cluster y, making the α_i smaller than the value previously calculated (without the inclusion of u in the cluster). This replacement would make i less strange than previously calculated. As a consequence, if we do not modify α_i, the algorithm will consider i more strange than really is, thus increasing the p-value for u. In other words, we risk lowering the chance of declaring the new point u an outlier. However, this only happens if the new point is close enough to examples in the cluster (indeed, closer than some of the neighbors of the example), and by definition, that point should not qualify as an outlier (there are points already in the cluster that are farther away than this one). So, we can sacrifice precision in the calculation of the p-values to gain speed in the diagnose of the outliers. Experimental results have shown that this change has no effect in the capacity of the algorithm of detecting true outliers. It is important to remark that this argument only holds if **every new point is considered in isolation**. If the new point is incorporated in any of the clusters, the α values of the other points already there may need to be recomputed. We are only interested here in processing every point in a new batch **separately**, and decide, individually whether each point is an outlier under the current clustering model or not. Once this decision is made for the whole batch of points, one needs to undergo a process by which the new batch of points is incorporated to the clustering model (which is not the objective of this paper). During that process, the α values of the points in the clustering model will be re-computed.

In summary, our method works as follows. Given a batch of new points, we determine for each one of them separately, whether the point is an outlier or not. We determine if the point u is an outlier by testing c null hypotheses of the form " u is fit to belong in cluster y" ($y = 1, \cdots, c$). If we can reject all these c hypothesis, the point is declared an outlier.

The other modification we make concerns the value of τ. Since we are performing multiple tests, one for every cluster (c in total), the actual confidence value will be $(1-\tau)^c$, which is smaller than $1-\tau$. In order to obtain the confidence level δ we want, we then select a value of τ, such that $(1-\tau)^c = \delta$.

When no clusters are available, the test can be administered to the data as a whole (as if it all belonged to one cluster). Doing that, of course, requires a single α_i per point (as opposed to computing one per cluster), and the τ used directly reflects the confidence level that is required ($\delta = 1 - \tau$).

Since the method requires to find K nearest neighbors on each cluster, we need $O(m)$ distance computations, per each point to be diagnosed, where m is the number of data points in the normal data set. (Hence, to diagnose n points, the complexity would be $O(mn)$.) Moreover, to find the distribution of α values for the normal data set, we require $O(m^2)$ comparisons. We observe that this step is done off-line and only once, before the diagnose of outliers starts. However, if m is very large, the off-line computation may still be too costly. In this case, to alleviate the complexity, one may sample the data set, and perform comparisons only with the sampled data. We perform experiments that show that this is a reasonable approach to handle large data sets, with little or no significant deterioration of the results.

4. EXPERIMENTAL RESULTS

This section describes experiments with real and synthetic data sets. In all the cases, the data sets were used as they were encountered, without applying pre-processing to them.

4.1 A synthetic data set

We designed a data set consisting of points grouped into three clusters according to normal distributions. The points have four dimensions and each cluster has 500 points. The centroids of the three clusters are $(1,1,1,1)$, $(7,1,1,1)$, and $(3,3,3,3)$, and the standard deviations used to generate the points are $(1,2,1,2)$, $(2,1,2,1)$, and $(1,1,2,3)$, respectively. We then constructed a test set consisting of 50 points that were more than 3 standard deviations away from the centroids (outliers) and 50 points that follow the same distribution as the cluster points (non-outliers). We use this set to compare our technique to the technique described in [18] of Distance-Based (DB) outliers. (A more detailed description can be found in Section 5.) We choose DB for the comparison, since, as ours, is distance based, does not make assumptions on the distribution of data, and it is applicable to multidimensional data. The results are summarized in Table 1. Across a large range of choices for K (2,5,10) and global confidence (90 %, 95 %, which result from choosing $\tau = 0.035$, and 0.017 respectively), all the outliers (100 % true positives) are flagged. For DB we varied one of the parameters of the method (i.e., p: the minimum fraction of tuples lying outside the D-neighborhood of an outlier; see description on Section 5) from 0.7 to 0.9. For all the cases, the true positives remained at 100 %. The false positives were 0, except for the case of $p = 0.7$, which shows 2 % of false positives. The results are comparable to those obtained by our technique.

4.2 The Fisher-Iris data set

The Fisher-Iris data set is a widely used, classic data set for supervised learning, which can be found in the UCI repository of machine learning data sets [30]. The data contains three classes of records, each class corresponding to a type of Iris plant. There are 50 records of each class in the set. We chose this set, as one of the classes (Iris Setosa) is highly separable from the other two. Taking away the class attribute, we form two clusters with 45 records from each of the two classes Virginica and Versicolor, and use the records of the Setosa class, plus the remaining records of the other two (5 each), to test our outlier detection algorithm. A good outlier detection technique should be able to recognize the Setosa records as outliers, without giving false alarms. Our experiment resulted in 100 % of the Setosa records declared as outliers. The other 10 records were not flagged as outliers. We have computed for each Setosa point the distance to the closest centroid (of the two clusters representing the Versicolor and Virginica classes), and found that these distances are in a small range: 2.86 to 3.79, with a mean of 3.24, and a small standard deviation: 0.20. The range for the "normal" data (Versicolor and Virginica points) is 0.23 to 1.80, with a mean of 0.73, and a standard deviation of 0.37.

To maintain a confidence of 95% we have set the $\tau = 0.025$ ($0.975^2 \approx 0.95$). These results hold even when we do not use the clustering information (i.e., we put all the 90 records of Virginica and Versicolor in one cluster) using $\tau = 0.05$. For both experiments we used $k = 5$.

Table 1: True and False positives rate (TP, FP) for the synthetic data.

	Our technique							DB			
	3 clusters						no clusters				
	$\delta = 90\%$			$\delta = 95\%$			$\delta = 95\%$				
k	2	5	10	2	5	10	5	p	0.7	0.8	0.9
TP	100%	100%	100%	100%	100%	100%	100%	TP	100%	100%	100%
FP	0%	0%	0%	0%	0%	0%	2%	FP	2%	0%	0%

4.3 The on-line bookstore data

This data set was generated from an e-commerce workload and has been used previously in [22]. The logs correspond to a couple of weekdays in which a large number of HTTP requests were processed. Entries corresponding to images, errors, etc., were deleted and the URLs of the remaining entries in the log were mapped to one of 12 e-business functions such as "add item to cart," and "pay." A session vector was generated for each session. This vector indicates the number of times that each of the 12 different functions (e.g., Search, Browse, Add) was invoked during the session (12 dimensions). The "ground truth" in this experiment consists of the domain knowledge that robot sessions are those with a total number of "clicks" bigger than or equal to 50. Those records were separated from the data set, along with some non-outliers used to test our algorithm. The basic clustering model was found using K-means over a set of records that do not contain any robot sessions, using $c = 3$. The members of each cluster differ on the intensity and characteristics of the sessions. The first cluster contains records with low activity (few clicks), the second cluster, records with moderate activity, and the third one with high activity. The last two clusters contain records of sessions in which the "Add" function was performed considerably more often than in the sessions represented by the first cluster records. (e.g., cluster 3 sessions correspond to "heavy buyers" in the bookstore). We used 101,808 records to form the original cluster model. Then we tested the technique with 65,536 records, of which 360 were known to be outliers and the rest (65,176) non-outliers.

Table 2 shows the results of running our technique on this data set. We used $\tau = 0.017$ to obtain a total confidence level of 0.95 ($0.983^3 \approx 0.95$). A large percentage of the outliers are detected by the test, while the false positive rate is kept low. The results throughout the range of K are stable.

Figure 3 shows the histogram of distances to the closest centroid for the non-outliers and outliers in the test data. The distributions are very different, with the bulk of the outliers concentrating at distances bigger than 7, while most of the non-outliers have distances smaller than 7.

We also conducted experiments varying the clustering model: using K-Means and different seeds (we tried 3 different seeds), we were able to obtain different clusterings for the baseline data. We compared the diagnoses of our method for each clustering, and found out that on the average 98% of the test points received the same diagnoses in the presence of two different clusterings of the same baseline data. This proves that the method is very robust with respect to changes in the clustering model.

In order to compare our technique with DB for this data set, we performed Principal Component Analysis and represented the data using the first four principal components (we do this because the code we have for DB only handles 4 dimensions). Table 3 shows the results for our technique and DB. The results are comparable to those shown in Table 2 (the fraction of true positives is less than 1 % smaller than the one found in the case of the full data set; the fraction of false positives is around 1 % larger than the one found in the case of the full data set). For DB, we discovered that, even when we decreased the value of p, it was possible to identify only a small fraction of the true outliers (less than 7%). The technique however, does not introduce any false positives for this data set.

Table 2: Results of the algorithm StrOUD on the bookstore data (all numbers are %).

k	1	5	10	20
TP	97.78	99.44	99.72	100
FP	0.26	0.45	0.50	0.53

Table 3: Results on the bookstore data (with PCA) (%) .

	StrOUD			DB		
	$k = 2$	$k = 5$	$k = 10$	$p = 0.7$	$p = 0.8$	$p = 0.9$
TP	97.80	98.99	99.44	6.39	2.22	2.22
FP	1.27	1.59	1.72	0	0	0

Table 4 shows the results of experiments that do not use the clustering information. In the first one (labelled 'original' in the Table), we conducted the test on the original bookstore data without the cluster information. The True Positive rate is almost as good as the one obtained using the clusters (99.72%), while the False Positive rate is larger but still very manageable (4.97%). The second test uses the 4-dimensional data obtained by PCA. The results are comparable to the original data (TP=100%, FP=4.90%). The last two tests were conducted by using a sample of the original data (as the normal set). The sample sizes were 1% and 10% respectively. The results show perfect recognition of the outliers in both cases (TP=100%), with a very slight increase of the False Positive rate (5.37% and 5.80%, for the 1% and 10% sample respectively). These experiments demonstrate that when the data set is large, it is possible to use a sample of the normal data to capture outliers without significant loss of accuracy.

4.4 Texture data

This is one of the real data sets of the Elena project, which can be found in [6]. The data set was generated by

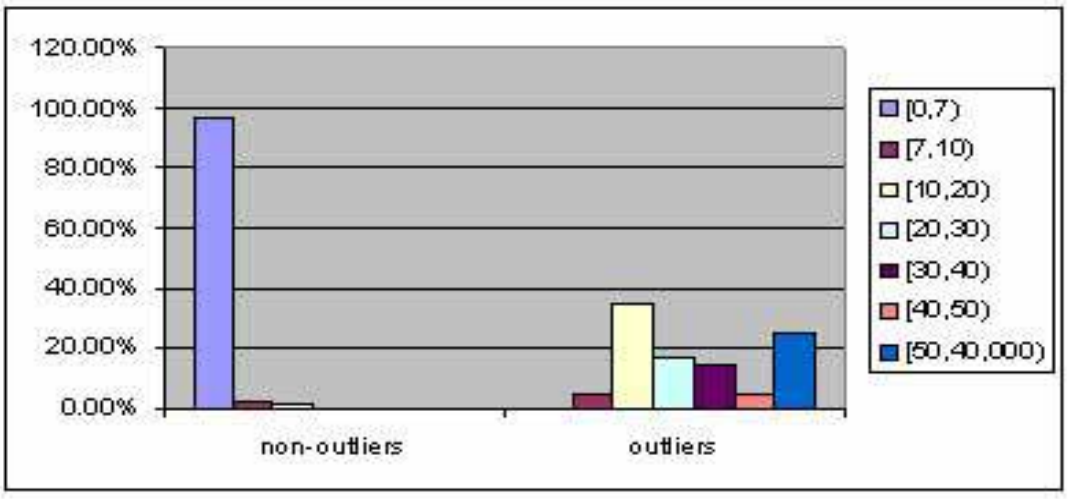

Figure 3: Distance histogram for the bookstore data. Each bar shows the percentage of points within each group (non-outliers, outliers) with distances to the closest centroid in the range indicated.

Table 4: Results of the algorithm StrOUD on the bookstore without clusters, for the original set, the PCA-reduced set, and samples.

	original	PCA	1% sample	10 % sample
TP	99.72	100	100	100
FP	4.97	4.90	5.37	5.80

Table 5: Results of the algorithm StrOUD on the texture data with and without clusters ($k = 5$).

	10 clusters	no clusters
TP	98.40	100
FP	0	4.93

Table 6: Results for the texture data (with PCA).

	StrOUD			DB		
	$k = 2$	$k = 5$	$k = 10$	$p = 0.7$	$p = 0.8$	$p = 0.9$
TP	93.60	87.20	83.20	100.00	14.80	0.40
FP	0	0	0	42.74	12.87	0.22

the Laboratory of Image Processing and Pattern Recognition (INPG-LTIRF) in Grenoble, France, using as the original source the material in [5], and referenced in [10, 11]. The data set contains a large number of classes (11) and a high dimensionality (40). The original aim was to distinguish between 11 different textures (Grass lawn, Pressed calf leather, Handmade paper, Raffia looped to a high pile, Cotton canvas, etc.), each pattern (pixel) being characterized by 40 attributes built by the estimation of fourth order modified moments in four orientations: 0, 45, 90 and 135 degrees. Again, we discard the class attribute and aim to detect the points in one class as outliers with respect to the points in other classes. Summarizing, this experiment used all the records of the texture data as the initial points, with the exception of: a) those that belong to one of the classes (500), b) A sample of five percent (223 in total) of records of other classes (which were then included in the test set). Each of the other classes was represented by a cluster, 10 clusters , 4,250 records in total. With $K = 5$ and $\tau = 0.005$ (to reach a final confidence of 95%, as $0.995^{10} \approx 0.95$), we obtain a true positive rate of 98.4%, with no false positives. When the clustering information is ignored, the true positive rate goes to 100% with 4.93% false positives. (See Table 5.)

Figure 4 shows the histograms of distances to the closest centroid for the points that are declared as outliers and those that are not in the test set. It is easy to see that the two distributions are radically different with little overlap on the distances.

In order to compare our technique with DB for this data set, we performed Principal Component Analysis and represented the data using the first four principal components. Table 6 shows the results for our technique and for DB. Although the percentage of true positives (outliers) captured is smaller than that obtained when using the entire data set, the technique still captures a large fraction of the outliers, with no false positives to report. For DB, since the 500 points of class 13 are somewhat close to each other, it is necessary to lower the value of p to capture them as outliers. By doing so, though, false positives are introduced.

4.5 National Hockey League data

We performed a test using NHL player's statistics (they can be obtained from Web sites such as nhlstatistics.hypermart.net) for the 1994 season. The reason for this test is that we wanted to compare the behavior of our technique with DB and the NHL 1994 data set is one of the case studies they investigated. We use a four-dimensional

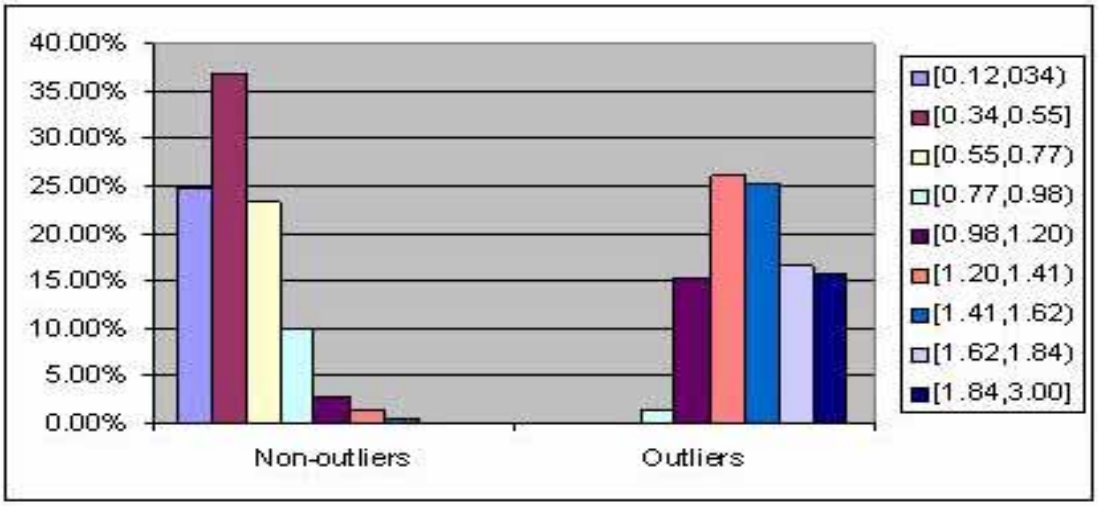

Figure 4: Histogram of distances to the closest centroid for the texture data. Each bar shows the percentage of points in the group that have distances to the closest centroid in the range indicated.

description for each player with the following attributes: the plus-minus statistic (indicates how many more event-strength goals were scored by the team when the player was in the ice), the number of penalty minutes, the percentage points, and goals scored. After tuning the parameters for the distance-based outliers code, we obtained seven outliers for the distance-based code. We took these seven records and added 27 other players (selected at random) to perform a test with our technique. The rest of the players in the data set (834) were grouped into 3 clusters (using K-means). We used $\tau = 0.025$ for a total confidence of 95 %. Our technique found the seven outliers and added 3 more to the list. (For a total of 10 outliers out of the 34 records tested.) Figure 5 shows the histograms of the distance of the points to the closest. centroid. The figure shows that the outliers found by our technique are indeed radically different than the points declared non-outliers, justifying the finding of the extra 3 outliers.

4.6 Cleaning the data

In this section we show the results of experiments aimed to clean a data set containing outliers, without relying on the presence of an outlier-free sample. To do so, we used the texture data, and "contaminated" the normal data with a number of outliers (from the class chosen to act as outliers). Then, we used this data both as the "normal" and the test sets. The only difference is that, in order to be transductive, whenever we test a data point we compute the α values of the training set **without including that point**. The results, shown in Table 7, indicate that we can effectively determine almost all the outliers present in the data set, especially when we use a high value of K. As expected intuitively, the more outliers we have in the set, the larger K needs to be. Of course, we do not know how many outliers we have in the data set in advance, so a "rule of thumb" has to be used to select K. (In nearest-neighbor classification, it is customary to set K as high as 10% of the number of examples in the data set.) The experiments show that a relatively low value of K (100, or around 2.3%) is enough to catch a high volume of the outliers (more than 95%) while maintaining a low false positive rate (less than 4%).

4.7 Using a noisy data set

In this section we show what happens if the "normal" data is contaminated with outliers and yet it is used to further detect other outliers. We conducted these experiments using the texture data and contaminating the "normal" data with outliers from the chosen class. We then used this data set as the basis for diagnosing the test set, without the outliers that were already used to contaminate the "normal" data. As a result, the test set contains the rest of the examples from the chosen class (not in the "normal" set) plus 223 examples of non-outliers. Table 8 shows the results of these experiments. The number of outliers used to contaminate the normal set is shown below the row with the values of K. Again, we diagnose a high percentage of the outliers in the test data (more than 95%) with a low false positive rate (less than 5%) when a value of K, commensurable with the number of contaminants is used. As we do not know this value, a "rule of thumb" value for K must be used, but the experiments show that a relative low value of K (100, or 2.3% of the normal data set examples) can effectively be used to diagnose further outliers.

5. RELATED WORK

Early work in outlier detection was done in the field of statistics (see [19]). However, these methods largely work with univariate data, and all of them assume knowledge of the underlying distribution of the data, which in practice is very restrictive. More recently a technique for spatial outlier detection was proposed in [28]. The method uses the difference between the attribute value at location x and the average attribute value of x's neighbors to determine

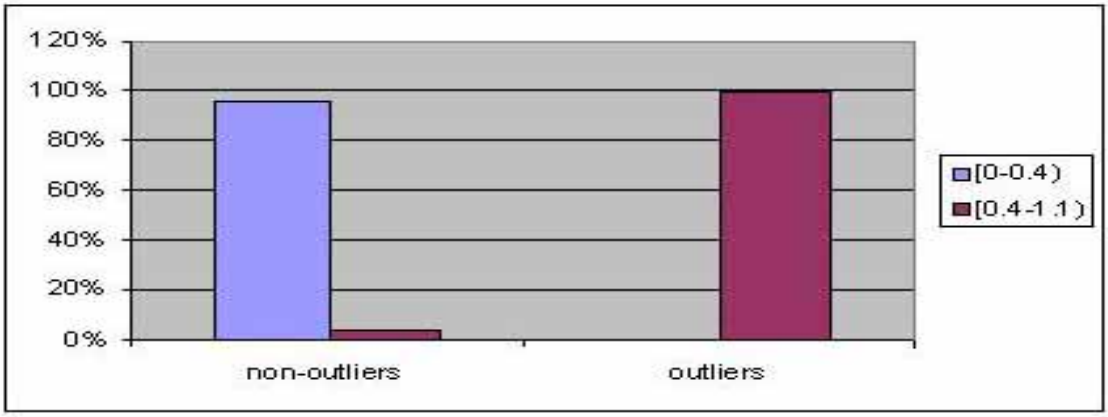

Figure 5: Histogram of distances to the closest centroid for the NHL data. Each bar shows the percentage of points in the group that have distances to the closest centroid in the range indicated.

if x is an outlier. It works only for univariate data (one attribute besides the spatial coordinates) , and assumes a normal distribution for the non-spatial attribute value.

An exception to the univariate restriction in the statistics techniques is the work of Rousseeuw et al (whose most recent survey can be found in [15]). Their approach is to find a robust fit model, i.e., one that is similar to the fit that would have been found without the outliers, and use it to diagnose which points do not fit the model well. Their choice is to compute the *Minimum Covariance Determinant* estimator or **MCD**, which can be briefly described as finding a subset of the examples in the data whose covariance matrix has the lowest possible determinant. Their algorithm, *FAST-MCD* uses randomization to speed up the high-complexity calculation. Outliers are detected by computing the Mahalanobis distance (using the covariance matrix computed by MCD) and finding points that are "too far away" from the robust centroid. One can argue that estimating a mean and covariance matrix from the data (or rather in MCD, a sub-sample of the data) corresponds to assuming a Gaussian distribution of the data. Also, looking for points with Mahalanobis distance that is "too large" calls for a threshold, i.e., it is equivalent to consider as outliers those points that are 3 or more standard deviations away from the centroid, which is the common practice of univariate statistical techniques. Moreover, in [13], which uses a technique based on MCD to find multiple clusters and diagnose outliers, the authors argue that not every data set will give rise to an obvious separation of outliers and non-outliers by using robust estimators.

Recently, Angiulli and Pizzuli [1] proposed an outlier-discovery algorithm that uses the sum of the distances to the K-nearest neighbors as a measure of isolation. This sum is called *weight* by the authors, and corresponds exactly to the α definition we use in this paper. The algorithm, however, aims to discover the top n outliers by sorting points by their weight and declaring the n points with largest weight as outliers. It is obviously difficult to set the value of n in advance: for a given value it is very possible that some of the points are not outliers. Our algorithm avoids this by providing a statistical test that will determine if a value of the weight (α) is indeed strange enough to declare the point an outlier. Their paper, however, describes a very useful technique that uses space-filling curves to avoid the costly computation of the K-nearest neighbors of each data point. The technique is based on transforming the d-dimensional points into one-dimensional points lying on a Hilbert or z-curve [25]. The transformation guarantees that two points that are close in the curve are also close in the d space. However, two points in the d space are not necessarily close in the Hilbert curve, so it is necessary to produce multiple transformations by shifting the curves. The idea of shifted Hilbert curves, described in [21] guarantees that a search for a nearest neighbor of a query point conducted over several lists of linearized points (each list corresponding to a shifted Hilbert curve) finds the true nearest neighbor with a high accuracy. Such technique can easily be employed in StrOUD to speed the computation.

A large body of work has been published in the area of discovering outliers with respect to clustering models (e.g., [7, 23, 27, 29, 12]). However, most of these algorithms do not aim at the discovery of outliers, but rather offer ways to deal with them. And, in all the cases, the discovery of outliers is done by careful setting of the algorithm parameters.

Some outlier detection schemes that do not assume a clustering model or a known distribution have been proposed. They fall under two categories. The first is *distance-based* techniques (see [2, 18, 26]. The second is *density-based* techniques (see [4, 17]). Again, in all these algorithms one must threshold parameters to obtain the set of outliers.

The method of [18], which we use here to provide a comparison to ours, uses distance and density calculations to discover the outliers in a data set. Briefly, the method aims to answer a nearest neighbor query with radius D for each point in the data set and decide if there are enough neighbors to the point in this D-neighborhood. This decision is controlled by a parameter p, which thresholds the minimum fraction of records that must be found outside of the D-neighborhood for the point to be an outlier. These leaves two parameters to be adjusted (D and p). However, the code of DB overwrites a badly chosen D by computing a reasonable one using sampling. So we concentrated in varying the choice of p in our experiments.

Table 7: Results of cleaning the texture data contaminated with a number of outliers of the chosen class. The number of outliers in the data set, and its corresponding percentage of the entire set are shown.

	10 (0.23%)		20 (0.46%)		50 (1.16 %)			100 (2.29%)	
k	5	10	5	10	10	20	40	40	100
TP	100	100	90	100	74	94	100	60	95
FP	2.23	3.48	2.16	3.27	2.63	3.27	3.62	3.27	3.62

Table 8: Results of diagnosing the texture data by using a contaminated "normal" set. The number of outliers in the "normal"data set, and its corresponding percentage of the entire set are shown.

	10 (0.23%)		20 (0.46%)		50 (1.16%)			100 (2.29%)	
k	5	10	5	10	10	20	100	40	100
TP	100	100	99.58	100	88.22	94.44	99.77	64.75	94.75
FP	4.93	4.93	4.93	4.93	4.03	4.93	3.14	4.04	3.14

6. CONCLUSIONS

We have presented here a novel technique to detect outliers based on statistical testing and the application of transduction. The technique does not make assumptions about the data distributions. It only requires that the number of neighbors (K) utilized in the distance calculation, and the confidence level, be provided. We claim that neither of these two parameters requires careful tuning (as the experiments show). People intuitively aim for a desired confidence level (95 % is a popular choice) and there are rules-of-thumb to set the number of neighbors. We have shown using extensive experimentation that the technique is robust with respect to the choice of K and that it obtains extremely good results for the standard choice of 95 % confidence level. We observe that the definition of strangeness does depend in turn on the distance metric used.

In this work we have fixed the metric to be the Euclidean distance to allow a fair comparison with the DB technique. Additional distance measures will be investigated in our future work. We have shown that discarding the clustering information has little effects on the results (generally a small increase on false positives), so the method can be used also in cases when the cluster information is not available. (And therefore we can claim that the method is robust even if the clusters are not.) We have compared the technique with the distance-based algorithm (DB) presented in [18], and the results show that our technique sometimes outperforms DB and is never outperformed by it.

We have also shown that it is possible to use our method to effectively clean a data set from outliers, thereby providing a sample of data that is, at least in a large percentage, free of outliers. Moreover, we have demonstrated that even if the "normal" data set is contaminated by outliers, it can be used to diagnose further outliers. We have shown that, for large data sets, it is possible to drastically improve the performance of the algorithm by using sampling, while maintaining good results. In our future work, we plan to investigate the idea of representing the clusters of normal data (or the whole data) by means of properly selected *representatives* (instead of uniformly sampling the data). We expect that this technique will result in a decrease in false positives with respect to the rates obtained by uniform sampling. Multidimensional search structures, such as Kd-trees [3] can also be utilized to speed the search for k-nearest neighbors. A more promising approach is to employ the space-filling curves techniques in [1], based on maintaining lists of the points ordered by their position on a series of Hilbert-curves, each shifted or rotated, as proposed in [21], to avoid the costly K-nearest neighbor computations for each point.

Transduction, as used here and by TCMs, only works if the data points are independently distributed. (For all the data sets we have experimented with, this assumption holds.) However, in many real situations, such as spatio-temporal data, this assumption is too strong. We plan to modify our method to deal with correlated data by introducing a technique that has been used successfully in the field of Relational Data Mining (e.g. [16]). Randomization testing [8] is a type of statistical test which involves generating replicates of the actual data set -called pseudo-samples. Randomization has been proven effective in compensating the effects of autocorrelation.

7. ACKNOWLEDGMENTS

We would like to thank Ed Knorr and Raymond Ng for sharing the DB code with us.

8. REFERENCES

[1] Angiulli, F. and Pizzuti, C. (2005) Outlier mining in large high-dimensional data sets. IEEE Transactions on Knowledge and Data Engineering, 17(2): 203-215.

[2] Bay, S.D., and Schwabacher, M. (2003) Mining distance-based outliers in near linear time with randomization and a simple pruning rule. *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining,* 29-38.

[3] Bentley, J.L. (1975) Multidimensional binary search trees used for associative searching. Communications of the ACM, 18(9):509-517.

[4] Breunig, M., Kriegel, H., Ng, R., Sander, J. (2000) LOF: Identifying Density-Based Local Outliers. *Proc. of the ACM SIGMOD Conference on Management of Data,* 427-438.

[5] Brodatz, P. (1966) Textures: A Photographic Album for Artists and Designers, Dover Publications,Inc.,New York.

[6] Elena project data. ftp://ftp.dice.ucl.ac.be/pub/neural-nets/ELENA/databases/

[7] Ester, M., Kriegel, H., Sander, J., and Xu, X. (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. *Proc. of the 2nd Intl. Conference on Knowledge Discovery and Data Mining.* 226-231.

[8] Edgington, E. (1980) Randomization Tests. New York: Marcel Dekker.

[9] Gammerman, A., and Vovk, V. (2002) Prediction algorithms and confidence measures based on algorithmic randomness theory. Theoretical Computer Science. 287: 209-217.

[10] Guerin-Dugue, A., and Aviles-Cruz, C. (1993) High Order Statistics from Natural Textured Images, *ATHOS workshop on System Identification and High Order Statistics*. Sophia-Antipolis, France.

[11] Guerin-Dugue, A. et al., (1995) Deliverable R3-B4-P - Task B4: Benchmarks, Technical report, Elena-NervesII "Enhanced Learning for Evolutive Neural Architecture", ESPRIT-Basic Research Project Number 6891.

[12] Guha, S., Rastogi, R., and Shim, K. (1998) CURE: an efficient clustering algorithm for large databases. *Proc. of ACM SIGMOD Conf. on Management of Data,* 73-84.

[13] Hardin, J., and Rocke, D.M. (2004) Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator. Computational statistics and data analysis, Vol 44, pp. 625-638.

[14] Hawkins, D. (1980) Identification of Outliers. Chapman and Hall, London.

[15] Hubert, M., Rousseeuw, P.J, and Van Aelst, S. (2005) Multivariate Outlier Detection and Robustness. In *Handbook of Statistics*, Vol. 24, C.R. Rao, E. Wegman, J. Solka, editors. Elsevier.

[16] Neville, J., Jensen, D., Friedland, L., and Hay, M. (2003) Learning Relational Probability Trees, *Proc. of the 9th ACM-SIGKDD Intl. Conference on Knowledge Discovery and Data Mining,* Washington, DC, 625 - 630.

[17] Jin, W., Tung, A., and Han, J. Mining (2001) Top-n local outliers in large databases. *Proc. of the Intl. Conference on Knowledge Discovery and Data Mining,* 293-298.

[18] Knorr, E., Ng, R. (1998) Algorithms for Mining Distance-Based Outliers in Large Datasets. *Proceedings of the 24th Intl. Conference on Very Large Databases,* 392-403.

[19] Lewis, B.V. (1994) Outliers in Statistical Data. John Wiley.

[20] Li, M., and Vitanyi, P. (1997) Introduction to Kolmogorov Complexity and its Applications. 2nd Edition, Springer Verlag.

[21] Liao, S., Lopez, M.A., and Leutenegger, S.T. (2001) High dimensional similarity search with space filling curves. Proc. International Conference on Data Engineering, 615-622.

[22] Menasce, D., Abraho, B., Barbará, D., Almeida, V., Ribeiro, F. (2002) Fractal Characterization of Web Workloads. *Proceedings of the "Web Engineering" Track of WWW2002, Honolulu, Hawaii, USA*, 7-11.

[23] Ng, R., and Han, J. (1994) Efficient and effective clustering methods for spatial data mining. *Proceedings of the 20th Intl. Conference on Very Large Data Bases*, 144-155.

[24] Proedru, K., Nouretdinov, I., Vovk, V., Gammerman, A. (2002) Transductive confidence machine for pattern recognition. *Proc. 13th European conference on Machine Learning*. 2430:381-390.

[25] Sagan, H. (1994)Space Filling Curves. Springer-Verlag.

[26] Ramaswamy, S., Rastogi, R., and Shim, K. (2000) Efficient algorithms for mining outliers from large data sets. *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 427-438.

[27] Sheikholesami, G., Chatterjee, S., and Zhang, A. (1998) WaveCluster: a multi-resolution clustering approach for very large spatial databases. *Proc. of the 24th Intl. Conference on Very Large Databases.* 428-439.

[28] Shekhar, S., Lu, C.T., and Zhang, P. (2003) A Unified Approach to Spatial Outliers Detection, *Geoinformatica*, 7(2).

[29] Tang, J., Chen, Z., Fu, A., and Cheung, D. (2002) Enhancing Effectiveness of Outlier Detection for Low Density Patterns. *Proc. of PAKDD'02*, 535-548.

[30] UCI Machine Learning Repository. http://www.ics.uci.edu/ mlearn/MLRepository.html

[31] Vapnik. V. (1998) Statistical Learning Theory, New York: Wiley.

[32] Vovk, V., Gammerman, A., and Saunders, C. (1999) Machine learning applications of algorithmic randomness. *Proceedings of the 16th Intl. Conference on Machine Learning*. 444-453.

[33] Ho, S.S., and Wechsler, H. (2003) Transductive Confidence Machine for Active Learning, *Int. Joint Conf. on Neural Networks*, Portland, OR.