

# Outlier Detection with Uncertain Data

Charu C. Aggarwal\*

Philip S. Yu†

## Abstract

In recent years, many new techniques have been developed for mining and managing uncertain data. This is because of the new ways of collecting data which has resulted in enormous amounts of inconsistent or missing data. Such data is often remodeled in the form of uncertain data. In this paper, we will examine the problem of outlier detection with uncertain data sets. The outlier detection problem is particularly challenging for the uncertain case, because the outlier-like behavior of a data point may be a result of the uncertainty added to the data point. Furthermore, the uncertainty added to the other data points may skew the overall data distribution in such a way that true outliers may be masked. Therefore, it is critical to be able to remove the effects of the uncertainty added both at the aggregate level as well as at the level of individual data points. In this paper, we will examine a density based approach to outlier detection, and show how to use it to remove the uncertainty from the underlying data. We present experimental results illustrating the effectiveness of the method.

## 1 Introduction

In recent years, advances in data collection technologies have lead to vast amounts of uncertain data. For example, new hardware technologies such as sensors are able to collect large amounts of data, but with missing or uncertain values. Such data present interesting research challenges for a variety of database and data mining applications [2, 6, 7, 8, 10, 20, 21]. A survey of different uncertain data algorithms and applications may be found in [4]. Some examples in which uncertain data management techniques are relevant are as follows:

- Sometimes the data may be an output of imputation or other statistical techniques such as forecasting. In such cases, the underlying data may be uncertain because of the inherent error of the statistical process.
- In many cases such as sensor networks, the errors may be a result of the imperfections in the hardware used for the collection process.

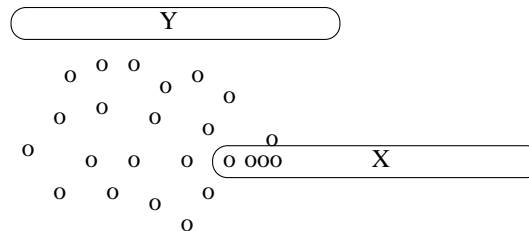


Figure 1: Outlier Detection Issues with Uncertain Data

- In privacy-preserving data mining, the output of the privacy methodology may be uncertain data. For example, when data sets are aggregated, this results in an uncertain representation of the data. Some recent techniques [3] explicitly create uncertain representations of the underlying data for mining purposes.

A key problem in data mining is that of outlier detection. The problem of outlier detection has been widely studied in the data mining community [1, 12, 18, 16]. The addition of noise to the data makes the problem far more difficult from the perspective of uncertainty. In order to explore this point further, we have illustrated a 2-dimensional data set in Figure 1. We have also illustrated two points in the data set along with the corresponding contours of uncertainty. If we did not use the uncertainty behavior across the different attributes, we would be more likely to label *X* as an outlier, since it seems further away from the data set. However, in reality, the data point *Y* is much closer to being an outlier, since the corresponding contours do not overlap with the dense regions in the data. The important point to be noted here is that the relative uncertainty in the different attributes over a given record is important to use in the process of determination of the outlier-like behavior of a given record. In general, a higher level of uncertainty of a given record can result in it behaving like an outlier, even though the record itself may not be an outlier.

Another problem with outlier detection techniques is that an increase in dimensionality results in difficulty in identification of outliers in the full dimensional case. This is because a full dimensional data set is inherently sparse, and therefore every pair of points may be

\*IBM T. J. Watson Research Center, charu@us.ibm.com

†University of Illinois at Chicago, psyu@cs.uic.edu

approximately equidistant from one another. In such cases, the data points do not cluster very well, and every point behaves like an outlier [1]. This problem is much greater in the case of uncertain data because of the additional noise associated with the uncertainty. The noise associated with the uncertainty reduces the clustering behavior of the data set with increasing dimensionality and uncertainty. In many cases, the only way to effectively discover an outlier is to examine subspaces of the data in order to determine regions in which the density of the data is abnormally low. We will use such a subspace exploration process in combination with density estimation in order to determine the outliers in the underlying data.

This paper is organized as follows. In the next section, we will discuss the outlier detection algorithm for uncertain data. We will explore the density estimation techniques which can be leveraged in conjunction with subspace exploration. In section 3, we will present the experimental results. Section 4 contains the conclusions and summary.

## 2 Outlier Detection Approach

We will first define some notations and definitions in order to facilitate further description. Let us assume that the database  $\mathcal{D}$  contains  $N$  records. The  $i$ th record in the database contains  $d$  dimensions, and is denoted by  $\overline{X}_i$ . It is also assumed that each dimension of a record has a probability density function (pdf) associated with it. The probability density function for the record  $\overline{X}_i$  along the  $j$ th dimension is denoted by  $h_i^j(\cdot)$ . Thus, the mean value of  $h_i(\cdot)$  is  $\overline{X}_i$ . It is also assumed that the standard deviation of  $h_i^j(\cdot)$  along the  $j$ th dimension is  $\psi_j(\overline{X}_i)$ .

The outlier detection algorithm relies on the fact that an outlier in high dimensional space shows up in a region of abnormally low density in at least one subspace of the data. Therefore, the outlier detection algorithm constructs a density estimate of the underlying data in various subspaces and uses these density estimates in order to determine the outliers in the data. We note that the presence of the uncertainty in the data affects how the density estimate is constructed. Data points and attributes with less uncertainty tend to be more concentrated in their contribution to the density estimate of the data. In a later section, we will explain how the density estimate of the data is constructed efficiently. First, we will discuss the outlier algorithm assuming that the density estimate is available. Since the uncertainty affects the approach to outlier detection, we need a definition of the outlier detection problem, which takes such uncertainty into account. In order to achieve this goal, we will define a sequence of terms and

notations.

First, we will define the  $\eta$ -probability of a given data point. This defines the probability that a data point lies in a region with (overall data) density at least  $\eta$ . Since the data point  $\overline{X}_i$  is itself uncertain, the probability density of the overall data set will vary along the contour of its uncertainty. The  $\eta$ -probability may be estimated by integrating the probability density function of the data point over those regions of overall data density which are greater than  $\eta$ .

**DEFINITION 1.** *The  $\eta$ -probability of a data point  $\overline{X}_i$  is defined as the probability that the uncertain data point lies in a region with (overall data) density at least  $\eta$ .*

We note that the  $\eta$ -probability of a data point can be computed in any subspace of the data. Clearly, the lower the  $\eta$ -probability, the less likely that the data point lies in a dense region. Such a point is an outlier. We note that the probability that a given data point lies in a region with density estimate at least  $\eta$  can be determined by integrating the density estimate of the probability density function of the data point over regions with probability density at least  $\eta$ . Consider the subspace  $S$  which is defined<sup>1</sup> by a subset of dimensions  $J = \{1 \dots r\}$ . Let the density function of the overall data set over the subspace  $J$  and coordinates  $(x_1 \dots x_r)$  be given by  $G(x_1 \dots x_r)$ . Then, the probability  $p_i$  that the uncertain data point  $\overline{X}_i$  lies in region with data density at least  $\eta$  is given by the following:

$$(2.1) \quad p_i = \int_{G(x_1 \dots x_r) \geq \eta} \prod_{j=1}^r h_i^j(x_j) dx_j$$

We note that the value of  $p_i$  is hard to compute exactly, but we will show that it can be estimated quite accurately. In order for the data point  $\overline{X}_i$  to be an outlier, this value of  $p_i$  needs to be less than a user-defined parameter  $\delta$ . This ensures that the data point has low probability of lying in a dense region of the data, even after the uncertainty in the position of the point is taken into account. Such a point is referred to as a  $(\delta, \eta)$ -outlier.

**DEFINITION 2.** *We will define an uncertain data point  $\overline{X}_i$  to be a  $(\delta, \eta)$ -outlier, if the  $\eta$ -probability of  $\overline{X}_i$  in some subspace is less than  $\delta$ .*

Next, we will discuss the overall algorithm for outlier detection.

<sup>1</sup>We can assume without loss of generality that  $J = \{1 \dots r\}$  by re-ordering the dimensions appropriately. We have used  $J = \{1, \dots, r\}$  for notational convenience only.

**Algorithm** *DetectOutlier*(Data:  $\mathcal{D}$ , Parameters:  $\eta, \delta$   
MaximumDim:  $r$ );

```

begin
   $\mathcal{O} = \{\}$ ;
   $\mathcal{C}_1 =$  All 1-dimensional candidates;
   $i=1$ ;
  while ( $\mathcal{C}_i$  is not null) and ( $i \leq r$ )
    begin
      Determine if any data point in  $\mathcal{D} - \mathcal{O}$ 
      is a  $(\delta, \eta)$ -outlier with respect to
      subspaces in  $\mathcal{C}_i$  and add corresponding point
      to  $\mathcal{O}$ ;
       $\mathcal{C}_{i+1} = \mathcal{C}_i \oplus \mathcal{C}_1$ ;
      { The sign  $\oplus$  corresponds to dimensional
      concatenation of each element of  $\mathcal{C}_i$  with
      those dimensions of  $\mathcal{C}_1$  which are not
      already included in the corresponding subspace
      element in  $\mathcal{C}_i$ ; }
       $i = i + 1$ ;
    end
  end

```

Figure 2: Outlier Detection Algorithm

## 2.1 Subspace Exploration for Outlier Detection

The overall algorithm for outlier detection uses a roll-up approach in which we test the outlier-like behavior of data points in different subspaces. The algorithm uses as input the parameters  $\delta$  and  $\eta$  which define the outlier-like behavior of the data points. In addition, the maximum dimensionality  $r$  of the subspaces is input to the algorithm. In order for the parameter  $\eta$  to have the same significance across different subspaces, we normalize the data, so that the standard deviation along each dimension is one unit.

The overall algorithm for outlier detection is illustrated in Figure 2. The algorithm works by using a bottom up enumeration approach in which it tests the possibility that the different points in the data are outliers. In order to do so, it successively appends dimensions to candidate subspaces, and tests whether different data points are outliers in these subspaces. The algorithm starts off with the outlier set  $\mathcal{O}$  set to null. The set  $\mathcal{C}_i$  denotes the candidate set of subspaces which are used for testing the possibility of a given data point being an outlier. The algorithm uses the concept of  $(\delta, \eta)$ -outlier in order to test whether a given data point is an outlier. In the event that a data point is indeed an outlier it is added to  $\mathcal{O}$ , and we move on to testing subspaces of the next higher dimension. This process continues until either  $\mathcal{C}_i$  is null, or we have exhausted all subspaces of dimensionality up to  $r$ . We note that this approach

**Algorithm** *EstimateProbability*(pdfs:  $h_1^1(\cdot) \dots h_i^r(\cdot)$ ,  
Integer:  $maxsamples$ )

```

begin
  for  $j = 1$  to  $r$  do
    begin
      Construct cumulative function  $H_i^j(\cdot)$ ;
      Construct inverted function  $F_i^j(\cdot)$ ;
    end
     $success = 0$ ;  $trials = 0$ ;
    for  $k = 1$  to  $maxsamples$  do
      begin
        Generate  $r$  random variables in  $[0, 1]$ 
        Denote generated random variables by  $y_1 \dots y_r$ ;
        Let  $q$  be the density at subspace coordinate
        denoted by  $(F_i^1(y_1) \dots F_i^r(y_r))$ ;
        if  $q > \eta$   $success = success + 1$ ;
         $trials = trials + 1$ ;
      end
    return( $success/trials$ );
  end

```

Figure 3: Estimating the  $\eta$ -probability

requires two subroutines which can be significantly difficult to implement:

- We need to determine whether a given data point is a  $(\delta, \eta)$ -outlier. Note that such a determination requires the computation of the integral in Equation 2.1, which can be quite difficult to estimate. Therefore, we will design a technique to estimate the integral in an efficient and accurate way with the use of probabilistic sampling.
- We need repeated density estimation in different subspaces over the entire data set. This can be extremely expensive, if it is not implemented carefully. We will use a cluster-based compression approach in order to improve the efficiency of the density estimation process.

**2.2 Estimating the  $\eta$ -probability** One tricky issue is the estimation of the  $\eta$ -probability of a given data point. We note that the integral of equation 2.1 cannot be easily computed in practice. Therefore, we will use a sampling procedure to estimate the  $\eta$ -probability. Let us assume that we are currently exploring the subspace  $J = \{1 \dots r\}$  in order to estimate the  $\eta$ -probability. In order to estimate the  $\eta$ -probability, we will repeatedly sample from the uncertain probability distribution of the record  $\bar{X}_i$  which is being tested as a possible outlier in subspace  $J$ . The overall data density is then computed at the sampled point, and it is checked if this

value is at least  $\eta$ . We keep track of the fraction of the samples that the value is at least  $\eta$  in order to estimate the corresponding probability. Next, we will discuss the details of the sampling process, and the accuracy in estimation of the  $\eta$ -probability with the use of Chernoff bounds.

As discussed earlier, let the probability density function for the  $j$ th attribute of the record  $\overline{X}_i$  be denoted the  $h_i^j(\cdot)$ . We also assume that the cumulative density function of the  $j$ th attribute for the record  $\overline{X}_i$  is denoted by  $H_i^j(\cdot)$ . We note that for a randomly sampled value  $v$  from the probability distribution function of the  $j$ th attribute of the record  $\overline{X}_i$ , the value of  $H_i^j(v)$  is a random variable with a uniform distribution in the range  $[0, 1]$ . This provides the approach for generating the variable  $v$ . Let  $F_i^j(\cdot)$  denote the inverse function of  $H_i^j(\cdot)$ . Then, in order to generate the  $j$ th attribute of  $\overline{X}_i$ , we first generate a value  $y$  drawn from the uniform distribution in the range  $[0, 1]$ . Then, we generate  $F_i^j(y)$  in order to generate  $v$ .

**OBSERVATION 1.** *If  $y$  is drawn from a uniform distribution in  $[0, 1]$ , then the value of  $F_i^j(y)$  generates a random variable from the distribution  $h_i^j(\cdot)$ .*

We note that the above analysis assumes that the inverse of the distribution can be computed efficiently. This will generally be the case, if the cumulative probability density functions can be expressed in closed form, or if the probability distributions of the underlying data are standard distributions. In such cases, the cumulative density functions are available as standard statistical tables, and are efficiently invertible from a computational point of view. Since most uncertainty descriptions are either drawn from standard probabilistic distributions, or are drawn from histogram tables, the inversion is usually simple from a computational point of view.

Thus, we use the generated samples from the different probability distributions in order to compute the estimates of the densities at different points in the locality of  $\overline{X}_i$ . We maintain two counts denoted by *trials* and *success*. In each iteration, the counter *trials* is incremented by 1. If a sampled density is greater than the threshold  $\eta$ , then we also increment the *success* counter. The overall procedure for estimating the  $\eta$ -probability is illustrated in Figure 3. We have assumed here that the subspace in which the estimation procedure is performed is denoted by  $J = \{1 \dots r\}$ . We can make this assumption without loss of generality, since the subspaces can be appropriately renumbered in order to index them into 1 through  $r$ . Clearly, the accuracy and computational efficiency of the approach is dependent on the number of samples used. This is denoted

by the parameter *maxsamples* in Figure 3. Therefore, it is useful to estimate the accuracy level achieved for a given number of samples. Note that, we are estimating whether a given data point is  $(\delta, \eta)$ -outlier in a given subspace  $J$ . Let us consider the case when the  $\eta$ -probability of the data point is  $\delta'$ . What is the likelihood that the correct determination is made for the subspace that the data point is indeed a  $(\delta, \eta)$ -outlier. Let us consider the case, where we *maxsamples* =  $m$ , and the number of successes from the  $m$  samples is denoted by the random variable  $W$ . Let  $Y_i$  be a binary random variable indicating whether on the  $i$ th sample is a success. Therefore, we have:

$$(2.2) \quad W = \sum_{i=1}^m Y_i$$

We note that the expected value of  $W$  is  $m \cdot \delta'$ , where  $\delta'$  is the true value of the  $\eta$ -probability. We note that if  $\delta'$  is estimated to within  $|\delta - \delta'|$  by the sampling approach, then a correct determination will always be made as to whether the point is a  $(\delta, \eta)$ -outlier. Therefore, by using the Chernoff bound, we have:

$$(2.3) P(|W - E[W]| \geq m \cdot |\delta - \delta'|) \leq 2 \cdot e^{-\frac{m \cdot |\delta - \delta'|^2}{4 \cdot \delta'}}$$

We summarize as follows.

**LEMMA 2.1.** *The probability that a given subspace correctly predicts a data point to be a  $(\delta, \eta)$ -outlier with the use of  $m$  samples is at least  $1 - 2 \cdot e^{-\frac{m \cdot |\delta - \delta'|^2}{4 \cdot \delta'}}$ .*

In order to understand the effectiveness of the approach, let us consider the case when we have  $\delta' = 0.1$ ,  $\delta = 0.15$ , and  $m = 800$ . In this case, the Chernoff bound yields a probability of correct estimation which is at least  $1 - 2 \cdot e^{-5} \approx 0.99$ . It is often possible to estimate whether a given point is a  $(\delta, \eta)$ -outlier correctly with high probability with the use of a modest number of samples. Furthermore, since there is often correlation in the behavior of different subspaces, it may be possible to identify an outlier with far fewer number of samples.

**2.3 Constructing the density estimate** We note that each iteration of the sampling procedure requires an estimate of the density of the data at the sampled point. This can be extremely expensive for large data sets, if it is not implemented carefully. Therefore, we will discuss a technique for density estimate construction of very large data sets efficiently, based on a technique discussed in [2].

The broad idea is to design an intermediate representation of the data which can then be leveraged in order to effectively perform the mining process. This

intermediate representation is in the form of a *adjusted density estimate*. We refer to the density estimate as “adjusted”, since we are taking the uncertainty into account while creating the estimate. The idea in kernel density estimation [19] is to provide a continuous estimate of the density of the data at a given point. The value of the density at a given point is estimated as the sum of the smoothed values of kernel functions  $K'_h(\cdot)$  associated with each point in the data set. Each kernel function is associated with a kernel width  $h$  which determines the level of smoothing created by the function. The kernel estimation  $\bar{f}(x)$  based on  $N$  data points and kernel function  $K'_h(\cdot)$  is defined as follows:

$$(2.4) \quad \bar{f}(x) = (1/N) \cdot \sum_{i=1}^N K'_h(x - \bar{X}_i)$$

Thus, each discrete point  $\bar{X}_i$  in the data set is replaced by a continuous function  $K'_h(\cdot)$  which peaks at  $X_i$  and has a variance which is determined by the smoothing parameter  $h$ . An example of such a distribution would be a gaussian kernel with width  $h$ .

$$(2.5) \quad K'_h(x - \bar{X}_i) = \left( \frac{1}{\sqrt{2\pi} \cdot h} \right) \cdot e^{-(x - \bar{X}_i)^2 / (2h^2)}$$

The result is a continuous distribution in which the random artifacts are suppressed and the density behavior provides a global overview of the dense as well as sparsely populated regions of the data. The estimation error depends upon the kernel width  $h$  which is chosen in a data driven manner. A widely used rule for approximating the bandwidth is the Silverman approximation rule [19] for which  $h$  may be chosen to be  $1.06 \cdot \sigma \cdot N^{-1/5}$ , where  $\sigma^2$  is the variance of the  $N$  data points. It has been shown [19] that for most smooth functions  $K'_h(\cdot)$ , when the number of data points goes to infinity, the estimator  $\bar{f}(x)$  asymptotically converges to the true density function  $f(x)$ , provided that the width  $h$  is chosen using the above relationship. For the  $d$ -dimensional case, the kernel function is chosen to be the product of  $d$  identical kernels  $K_i(\cdot)$ , each with its own smoothing parameter  $h_i$ .

The presence of uncertainty can affect the density function significantly. This is because data points which have low uncertainty affect their immediate locality to a greater degree. On the other hand, data points with larger uncertainty also affect a larger region with their uncertainty behavior. If different attributes have different uncertainty behavior, then the effects on the density computations can be significant. Therefore, the key is to adjust the kernel function in order to account for the error. We define the following error-based kernel  $Q'_h(x - X_i, \psi(\bar{X}_i))$  function, which depends both upon

the error as well as the values of the underlying data points.

$$Q'_h(x - X_i, \psi(\bar{X}_i)) = \frac{1}{\sqrt{2\pi} \cdot (h + \psi(\bar{X}_i))} \cdot e^{\frac{-(x - X_i)^2}{2 \cdot (h^2 + \psi(\bar{X}_i)^2)}}$$

The overall effect of changing the kernel function is that the width of the bandwidth along the corresponding dimension is increased by  $\psi(\bar{X}_i)$ . The intuition behind this choice of modifying the kernel is to adjust the contributions of the kernel function for a point depending upon its (error-based) probability density. As in the previous case, the error-based density at a given data point is defined as the sum of the error-based kernels over different data points. Therefore, we define the error based density  $f^Q$  at point  $x$  as follows:

$$(2.6) \quad f^Q(x, \psi(\bar{X}_i)) = \left( \frac{1}{N} \right) \cdot \sum_{i=1}^N Q'_h(x - \bar{X}_i, \psi(\bar{X}_i))$$

As in the previous case, we can easily generalize the definition to the multi-dimensional case. Specifically, the error for the  $j$ th dimension is denoted by  $\psi_j(\bar{X}_i)$ . The overall kernel function is defined as the product of the kernel function for the different dimensions.

One point to be remembered is that the density function needs to be constructed repeatedly over many subspaces. This can be extremely time consuming if the underlying database is very large. In order to generalize the approach to very large data sets, we condense the data into a smaller number of micro-clusters. We define the concept of uncertain micro-clusters as follows:

**DEFINITION 3.** A *micro-cluster* for a set of  $d$ -dimensional points  $X_{i_1} \dots X_{i_n}$  with time stamps  $T_{i_1} \dots T_{i_n}$  is defined as the  $(3 \cdot d + 1)$  tuple  $(\overline{CF2^x}(\mathcal{C}), \overline{EF2^x}(\mathcal{C}), \overline{CF1^x}(\mathcal{C}), n(\mathcal{C}))$ , wherein  $\overline{CF2^x}(\mathcal{C})$ ,  $\overline{EF2^x}(\mathcal{C})$ , and  $\overline{CF1^x}(\mathcal{C})$  each correspond to a vector of  $d$  entries. The definition of each of these entries is as follows:

- For each dimension, the sum of the squares of the data values is maintained in  $\overline{CF2^x}(\mathcal{C})$ . Thus,  $\overline{CF2^x}(\mathcal{C})$  contains  $d$  values. The  $p$ -th entry of  $\overline{CF2^x}(\mathcal{C})$  is equal to  $\sum_{j=1}^n (x_{i_j}^p)^2$ .
- For each dimension, the sum of the squares of the errors in the data values is maintained in  $\overline{EF2^x}(\mathcal{C})$ . Thus,  $\overline{EF2^x}(\mathcal{C})$  contains  $d$  values. The  $p$ -th entry of  $\overline{EF2^x}(\mathcal{C})$  is equal to  $\sum_{j=1}^n \psi_p(X_{i_j})^2$ .
- For each dimension, the sum of the data values is maintained in  $\overline{CF1^x}(\mathcal{C})$ . Thus,  $\overline{CF1^x}(\mathcal{C})$  contains  $d$  values. The  $p$ -th entry of  $\overline{CF1^x}(\mathcal{C})$  is equal to  $\sum_{j=1}^n x_{i_j}^p$ .
- The number of points in the data is maintained in  $n(\mathcal{C})$ .

The main difference between this definition and that discussed in [5] is the addition of error information to

the micro-clustering definition. The actual maintenance of the micro-clusters is a variation of the approach discussed in [5]. In this variation, we maintain the micro-cluster statistics for the  $q$  different centroids. These  $q$  centroids are chosen randomly. Each incoming data point is always assigned to its closest micro-cluster centroid using a nearest neighbor algorithm, and is never allowed to create a new micro-cluster. We use such an approach to ensure that all data points are included in the micro-cluster statistics. The goal of micro-clustering is to compress the data so that the resulting statistics can be held in main memory for quick density estimation process over different subspaces. By choosing a small number  $q$  of micro-clusters in order to compress the data, it is possible to perform the density estimation efficiently.

Each micro-cluster is treated as a pseudo-point with a new error defined both by the variance of the points in it as well as their individual errors. In order to estimate the error associated with the pseudo-point corresponding to a micro-cluster, we assume that each data point  $\bar{X}$  in a micro-cluster is a mutually independent observation with bias equal to its distance from the centroid and a variance equal to the error  $\psi(\bar{X})$ . Therefore, the true error  $\phi(\bar{X}, \mathcal{C})^2$  of the data point  $\bar{X}$  assuming that it is an instantiation of the pseudo-observation corresponding to the micro-cluster  $\mathcal{C}$  is given by:

$$(2.7) \quad \phi_j(\bar{X}, \mathcal{C})^2 = bias_j(\bar{X}, \mathcal{C})^2 + \psi_j(\bar{X})^2$$

Here  $\phi_j(\bar{X}, \mathcal{C})$  refers to the error in the  $j$ th dimension. We note that this result follows from the well known statistical relationship that the true error is defined by the squared sum of the bias and the variance. It has been shown in [2] that this error can be computed using the summary statistics of the micro-clusters:

LEMMA 2.2. *The true error  $\Delta(\mathcal{C})$  for the pseudo-observation for a micro-cluster  $\mathcal{C} = \{Y_1 \dots Y_r\}$  is given by the relationship:*

$$\Delta_j(\mathcal{C}) = \sum_{i=1}^r \frac{\phi_j(\bar{Y}_i, \mathcal{C})^2}{r} = \frac{CF2_j^x(\mathcal{C})}{r} - \frac{CF1_j^x(\mathcal{C})^2}{r^2} + \frac{EF2_j(\mathcal{C})}{r}$$

We can then use the above result to efficiently compute the kernel function of the micro-cluster. Let the centroid of the cluster  $\mathcal{C}$  be denoted by  $c(\mathcal{C})$ . Correspondingly, we define the kernel function for the micro-cluster  $\mathcal{C}$  in an analogous way to the error-based definition of a data point:

$$Q'_h(x - c(\mathcal{C}), \Delta(\mathcal{C})) = \frac{1}{\sqrt{2\pi} \cdot (h + \Delta(\mathcal{C}))} \cdot e^{-\frac{(x - X_j)^2}{(2 \cdot (h^2 + \Delta(\mathcal{C})^2))}}$$

We define the overall density as the weighted sum of the corresponding kernel functions. The weight is defined by the number of points in the different micro-clusters. Therefore, if the data contains the clusters  $\mathcal{C}_1 \dots \mathcal{C}_m$ , then we define the density estimate at  $x$  as follows:

$$\bar{f}^Q(x, \Delta(\mathcal{C})) = (1/N) \cdot \sum_{i=1}^m n(\mathcal{C}_i) \cdot Q'_h(x - c(\mathcal{C}_i), \Delta(\bar{X}_i))$$

We note that the complexity of the above computation is really dependent only on the number of micro-clusters rather than the base data set. In the next section, we will see that this helps in the efficient and effective computation of the outliers from the base data.

### 3 Experimental Results

In this section, we will present the experimental results illustrating the effectiveness and efficiency of the technique. The algorithms were tested on a number of data sets from the UCI Machine Learning Repository as well as a number of synthetic data sets. We note that the data sets from the UCI Machine Learning Repository were deterministic, and we needed to model and add uncertainty to these data sets.

In order to model the uncertainty added to the data sets, we used a gaussian distribution which models the level of uncertainty added to the data sets. The uncertainty added to each attribute was drawn from a gaussian distribution with zero mean and a standard deviation whose parameter was chosen as follows. For each entry, the standard deviation parameter of the normal distribution was chosen from a uniform distribution in the range  $[0, 2 \cdot f] \cdot \sigma$ , where  $\sigma$  is the standard deviation of that dimension in the underlying data. Thus, by changing the value of  $f$ , it is possible to vary the error level of the data set. We note that when the value of  $f$  is chosen to be 3, the majority of entries in the data are distorted by as much as 3 standard deviations of the original distribution. At this error level, the distorted value of the entry could lie outside the tail of the original distribution even at 99.99% level of confidence under the normal distribution assumption. The probability distribution for the distorted value was the same as that used to add uncertainty to the data, except that we centered it at the distorted value rather than the original value.

For the case of synthetic data sets, we generated the data points by creating Gaussian clusters in the underlying data. The centers of the clusters were generated uniformly in the unit data cube. The number of data points in each cluster was proportional to a random variable drawn from a uniform distribution in  $[0, 1]$ . The radius along each dimension was

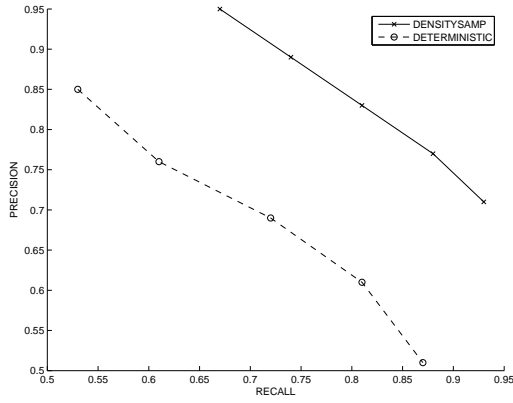


Figure 4: The Precision-Recall Tradeoff for Data Set  $R(0.3).O(0.1).d(10).D100K.U(1.5)$

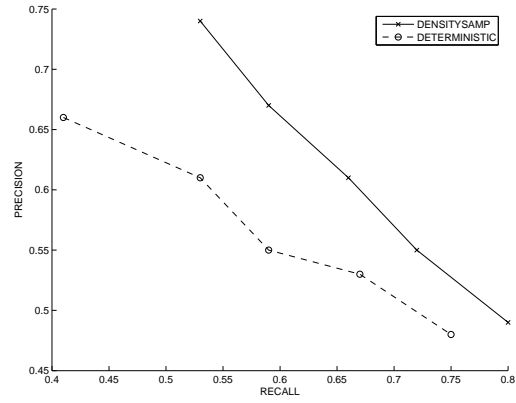


Figure 7: The Precision-Recall Tradeoff for Data Set  $Cancer.U(1.5)$

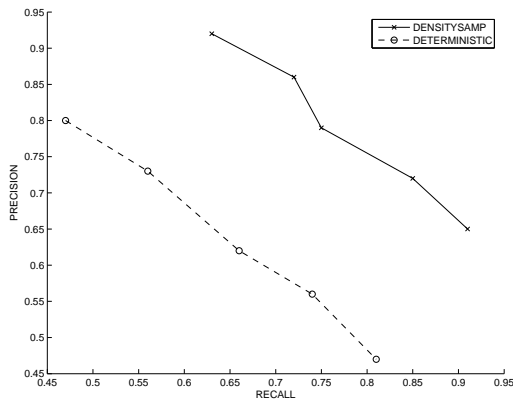


Figure 5: The Precision-Recall Tradeoff for Data Set  $R(0.3).O(0.2).d(10).D100K.U(1.5)$

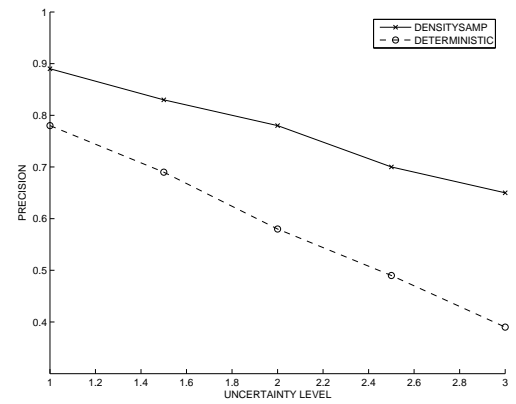


Figure 8: Precision with increasing uncertainty for Data Set  $R(0.3).O(0.1).d(10).D100K.U(x)$

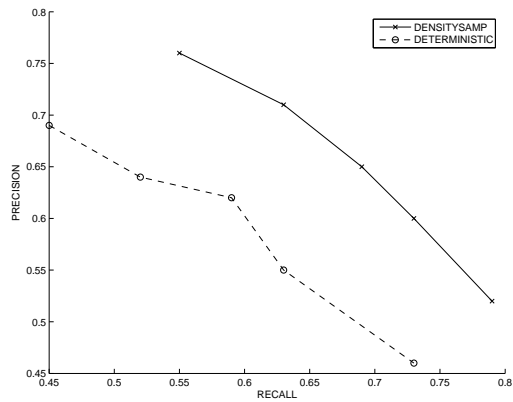


Figure 6: The Precision-Recall Tradeoff for Data Set  $Adult.U(1.5)$

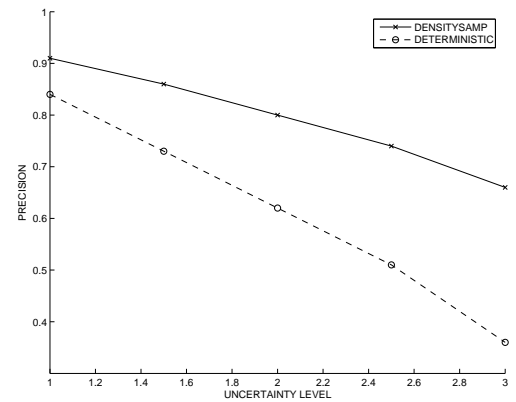


Figure 9: Precision with increasing uncertainty for Data Set  $R(0.3).O(0.2).d(10).D100K.U(x)$

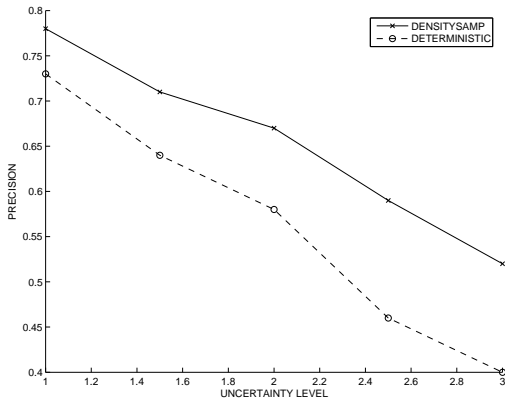


Figure 10: Precision with increasing uncertainty for Data Set  $Adult.U(x)$

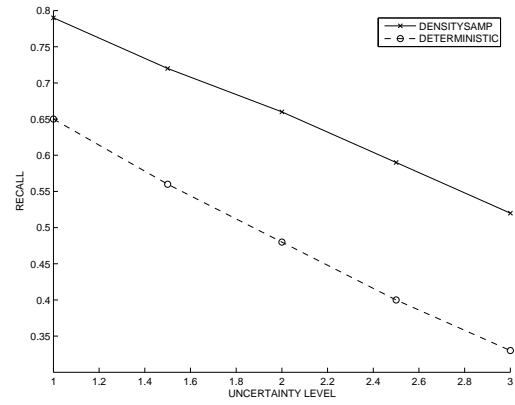


Figure 13: Recall with increasing uncertainty for Data Set  $R(0.3).O(0.2).d(10).D100K.U(x)$

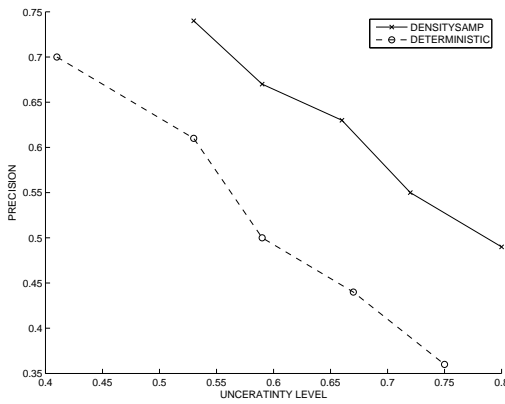


Figure 11: Precision with increasing uncertainty for Data Set  $Cancer.U(x)$

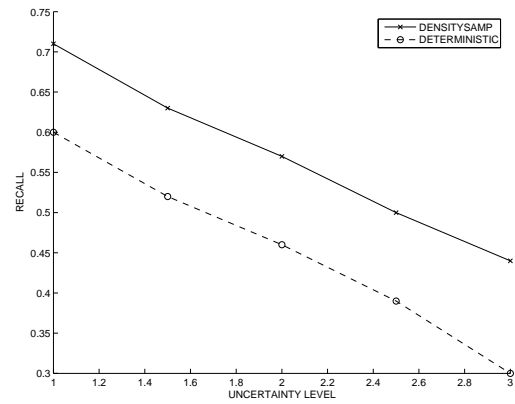


Figure 14: Recall with increasing uncertainty for Data Set  $Adult.U(x)$

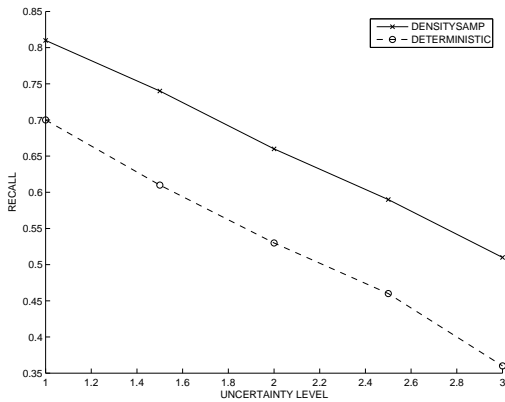


Figure 12: Recall with increasing uncertainty for Data Set  $R(0.3).O(0.1).d(10).D100K.U(x)$

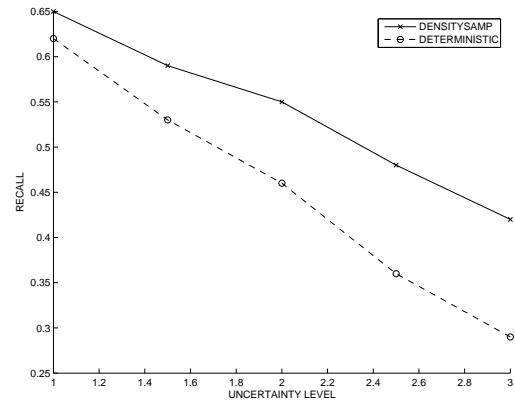


Figure 15: Recall with increasing uncertainty for Data Set  $Cancer.U(x)$

drawn from a uniform distribution in  $[0, r]$ . A fraction  $p$  of the data points were designated as outliers. The outliers were generated anywhere in the data cube. A total of  $N$  data points were generated in  $d$  dimensions. The corresponding data set was denoted by  $R(r).O(p).d(d).D(N)$ . For example, the data with radii drawn from  $[0, 0.3]$ , a fraction 0.2 of outliers, and 100K data points in 10 dimensions was denoted by  $R(0.3).O(0.2).d(10).D100K$ . We generated two data sets corresponding to  $R(0.3).O(0.1).d(10).D100K$  and  $R(0.3).O(0.2).d(10).D100K$  respectively. In order to model an uncertainty level of  $f$ , we add the symbol  $U(f)$  to the data set. Thus, the overall data set is denoted by  $R(r).O(p).d(d).D(N).U(f)$ . In the case of real data sets, we only needed to add the appendage  $U(f)$  to the data set name. Thus, the data set  $\langle \text{dataname} \rangle$  with uncertainty level  $f$  is denoted by  $\langle \text{dataname} \rangle .U(f)$ . All data sets were normalized, so that the standard deviation along each dimension was 1 unit. This was done in order to ensure that the same value of  $\eta$  could be used across different combinations of dimensions.

Since there are no known algorithms for outlier detection with uncertain data, we will use the deterministic algorithm discussed in [1] as a baseline. A second question is the use of the metric needed in order to judge the algorithm effectiveness. For the case of synthetic data sets, since the outliers were known, the precision and recall could be measured explicitly. By varying the value of  $\eta$  different tradeoffs could be reached between precision and recall. Similarly, in the case of the baseline algorithm discussed in [1], the tradeoff between the precision and recall could be measured by varying the sparsity coefficient. By comparing the two tradeoff curves, it is possible to determine the effectiveness of the technique.

In the case of real data sets, the measurement process is much more challenging. This is because we do not have a pre-defined notion of what the outliers ought to be. Therefore, we used the following technique in order to label data points as outliers. We used two different deterministic algorithms [1, 18] in order to determine the outliers on the *original unperturbed data*, and used the results as the ground truth for the method. Since we used the original unperturbed data for determining the (ground truth) outliers, it was assumed that the algorithms provided good quality results on the outliers. Such data could be used as ground truth in order to test the effects of uncertainty on the algorithms. Note that the baseline algorithm [1] has a clear advantage with the use of this kind of approach, since it is one of the algorithms used to determine the ground truth (on the unperturbed data), and the main source of inaccuracy is the additional noise

added to the data. We will show that the probabilistic algorithm proposed in this paper is superior to the algorithm of [1], since it does not degrade quite as much with increasing uncertainty.

**3.1 Effectiveness Results** The quality of the results are measured in terms of the *precision* and *recall* compared to the baseline outliers. The precision was defined as the percentage of outliers found which later turned out to be true outliers. The recall was defined as the percentage of true outliers which were found by the algorithm.

Unless otherwise mentioned, the default values used were  $\delta = 0.1$ ,  $\eta = 0.1$ , and  $r = 2$ . In each case, we used 140 micro-clusters in order to represent the data. In some cases such as the precision-recall tradeoff curve, the value of  $\eta$  was varied in order to create the curve, and therefore the default value was not used. First, we will present these tradeoff curves for the different data sets. The value of  $f$  was fixed at 1.5 for each of these data sets. In Figures 4 and 5 we have illustrated the results for the two data sets denoted by  $R(0.3).O(0.1).d(10).D100K.U(1.5)$  and  $R(0.3).O(0.2).d(10).D100K.U(1.5)$  respectively. We have illustrated the curves for both the probabilistic method and the technique discussed in [1] in Figures 4 and 5. The probabilistic density based sampling method is denoted by *DensitySamp*, and the deterministic outlier detection technique of [1] by *Deterministic*. It is clear that for each point in the curve, the *DensitySamp* method significantly dominates the deterministic outlier detection method of [1]. The difference between the two data sets is that the curve of Figure 4 has somewhat higher precision and recall than that of Figure 5. This is because the latter has a higher percentage of outliers, and is therefore a more noisy data set. For the case of real data sets, we used the Adult data set and the Wisconsin Breast Cancer Data Sets from the UCI Machine Learning Repository. The resulting data sets were denoted by *Adult.U(1.5)* and *Cancer.U(1.5)* respectively. The tradeoff curves for these data sets are denoted in Figures 6 and 7 respectively. In each case, it is clear that the *DensitySamp* method significantly outperformed the technique proposed in [1]. The absolute values of the precision and recall are somewhat lower, since the real data set did not have the well formed properties of the synthetic data sets. Nevertheless, the relative trend between the two methods was preserved in the same way for the real data sets, as the synthetic data.

We also tested the effectiveness of the technique with increasing level of uncertainty. The precision results for the synthetic data sets are illustrated in Figures

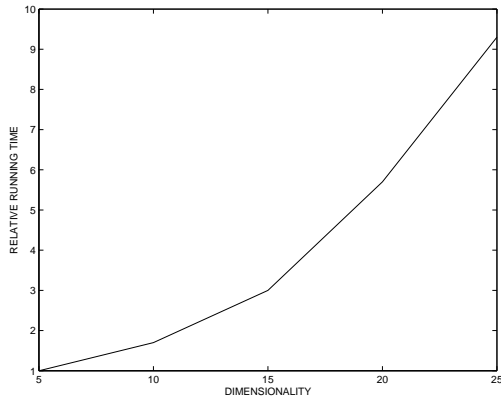


Figure 16: Scalability with Increasing Data Dimensionality

8 and 9. For the case of the *DensitySamp* algorithm, we fixed the value of  $\eta$  to that used at the middle point of the corresponding curves of Figures 4 and 5 respectively. Similarly, in the case of the deterministic algorithm of [1], we used the sparsity coefficient at the middle point of the corresponding curves of Figures 4 and 5. We have denoted the uncertainty level  $f$  on the  $X$ -axis, and the precision on the  $Y$ -axis. It is clear that the precision falls with increasing uncertainty level. We have also illustrated the results for the algorithm in [1] in the same chart. It is clear that the precision results for the *DensitySamp* method are superior to the results of [1] for all uncertainty levels, and the difference between the two increases with increasing uncertainty. At very high uncertainty levels such as  $f = 3$ , the precision results of the deterministic method fall off very sharply. We have illustrated similar results for recall in Figures 12 and 13 respectively. As in the previous case, we have illustrated the uncertainty level  $f$  on the  $X$ -axis, and the recall on the  $Y$ -axis. The results for recall are very similar to the precision results. In each case, the recall falls with increasing uncertainty, but the results for the probabilistic sampling method do not degrade as much with increasing uncertainty. The results for precision and recall are quite similar over different data sets. The precision results for the real data sets are illustrated in Figures 10 and 11 respectively. The corresponding recall results are illustrated in Figures 14 and 15 respectively. The results for the real data sets are quite similar to those of the synthetic data sets.

**3.2 Efficiency** We also tested the efficiency of the technique with increasing data set size and dimensionality. We tested the running time with increasing data dimensionality. The precise data set used for this testing was  $R(0.3).O(0.2).d(x).D(100K).U(1.5)$ , where  $x$  may

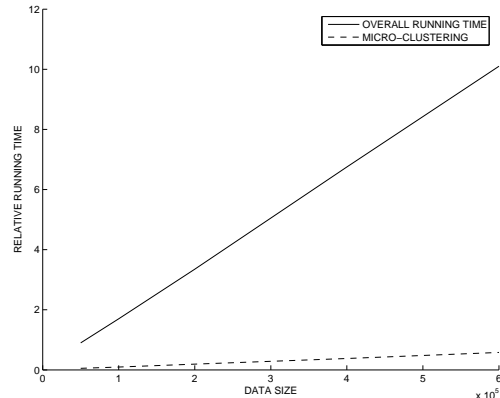


Figure 17: Scalability with Increasing Data Size

very with data dimensionality. The results are illustrated in Figure 16. On the  $X$ -axis, we have illustrated the data dimensionality, whereas the running time is illustrated on the  $Y$ -axis. The algorithm scaled super-linearly with increasing data dimensionality because of the rollout approach used for subspace exploration.

In Figure 17, we have illustrated the running time of the technique with increasing data set size. The specific data set which was used was  $R(0.3).O(0.2).d(10).D(x).U(1.5)$ , where  $x$  may vary with data set size. We have illustrated the overall data set size on the  $X$ -axis, whereas the running time is illustrated on the  $Y$ -axis. In the Figure, we have illustrated the running time for the entire algorithm as well as the pre-processing phase of micro-clustering. It is clear that the overall algorithm scales up linearly with increasing data set size. This linear scaleup is because of the following:

- The pre-processing stage uses micro-clustering, for which the complexity is linear with increasing data set size.
- For each data point, the running time for testing whether it is an outlier is constant. This is because the process of density estimation uses the intermediate micro-clusters in order to perform the probability estimation and subsequent outlier detection. Since the time for testing each data point is a constant, the overall scalability is linear with data set size. We note that if we had not used micro-clustering in order to compact the data, the scalability would have been quadratic with increasing data set size.

We have also illustrated the running time without the micro-clustering phase in the same chart. This shows that the running time without micro-clustering scales linearly with increasing data set size.

## 4 Conclusions and Summary

In this paper, we presented a new algorithm for outlier detection in uncertain data sets. The algorithm uses a probabilistic definition of outliers in conjunction with density estimation and sampling. The new definition uses the probability of presence of the uncertain data point in a sparse region of the data in order to determine whether or not it is an outlier. Different subspaces of the data are checked in order to determine the data points which are the best outliers. We used micro-clustering in combination with density estimation in order to improve the efficiency of the procedure for large data sets. We tested the algorithm against the deterministic technique proposed in [1], and showed that the probabilistic algorithm was significantly superior in terms of the precision-recall tradeoff.

## References

- [1] C. C. Aggarwal, and P. S. Yu, *Outlier detection for high dimensional data*, ACM SIGMOD Conference on Management of Data, (2001).
- [2] C. C. Aggarwal, *On Density Based Transforms for Uncertain Data Mining*, IEEE ICDE Conference, (2007).
- [3] C. C. Aggarwal, *On Unifying Privacy and Uncertain Data Models*, IEEE ICDE Conference, (2008).
- [4] C. C. Aggarwal, and P. S. Yu, *A Survey of Uncertain Data Algorithms and Applications*, IBM Research Report RC24394, (2007).
- [5] C. C. Aggarwal, J. Han, J. Wang, and P. Yu, *A Framework for Clustering Evolving Data Streams*, VLDB Conference, (2003).
- [6] D. Barbara, H. Garcia-Molina, and D. Porter, *The management of probabilistic data*, IEEE Transactions on Knowledge and Data Engineering, 4(5), pp. 487–502, (1992).
- [7] D. Burdick, P. Deshpande, T. S. Jayram, R. Ramakrishnan, and S. Vaithyanathan, *OLAP Over Uncertain and Imprecise Data*, VLDB Conference, pp. 970–981, (2005).
- [8] R. Cheng, D. Kalashnikov, and S. Prabhakar, *Evaluating Probabilistic Queries over Imprecise Data*, ACM SIGMOD Conference on Management of Data, (2003).
- [9] N. Dalvi, and D. Suciu, *Efficient Query Evaluation on Probabilistic Databases*, VLDB Conference, (2004).
- [10] A. Das Sarma, O. Benjelloun, A. Halevy, and J. Widom, *Working Models for Uncertain Data*, IEEE ICDE Conference, (2006).
- [11] A. Hinneburg, C. Aggarwal, and D. Keim, *What is the nearest neighbor in high dimensional space?*, VLDB Conference, (2000).
- [12] M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander, *LOF: Identifying Density-Based Local Outliers*, ACM SIGMOD Conference on Management of Data, (2000).
- [13] H.-P. Kriegel, and M. Pfeifle, *Density-based clustering of uncertain data*, ACM KDD Conference, (2005).
- [14] L. V. S. Lakshmanan, N. Leone, R. Ross, and V. S. Subrahmanian, *ProbView: A Flexible Database System*, ACM Transactions on Database Systems, 22(3):419–469, (1997).
- [15] S. I. McClean, B. W. Scotney, and M. Shapcott, *Aggregation of Imprecise and Uncertain Information*, IEEE Transactions on Knowledge and Data Engineering, 13(6):902–912, (2001).
- [16] E. Knorr, and R. Ng, *Algorithms for Mining Distance-based Outliers in Very Large Databases*, VLDB Conference, (1998).
- [17] D. Pfozer, and C. Jensen, *Capturing the uncertainty of moving object representations*, SSDM Conference, (1999).
- [18] S. Ramaswamy, R. Rastogi, and K. Shim, *Efficient Algorithms for Mining Outliers from Large Data Sets*, ACM SIGMOD Conference on Management of Data, (2000).
- [19] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, (1986).
- [20] S. Singh, C. Mayfield, S. Prabhakar, R. Shah, and S. Hambrusch, *Indexing Uncertain Categorical Data*, IEEE ICDE Conference, (2007).
- [21] Y. Tao, R. Cheng, X. Xiao, W. Ngai, B. Kao, and S. Prabhakar, *Indexing Multi-dimensional Uncertain Data with Arbitrary Probability Density Functions*, VLDB Conference, (2005).