An Efficient Approximation Scheme for Data Mining Tasks

George Kollios
Boston University
gkollios@cs.bu.edu

Dimitrios Gunopulos*
UC Riverside
dg@cs.ucr.edu

Nick Koudas
AT&T Laboratories
koudas@research.att.com

Stefan Berchtold
stb gmbh
berchtold@stb-gmbh.de

Abstract

We investigate the use of biased sampling according to the density of the dataset, to speed up the operation of general data mining tasks, such as clustering and outlier detection in large multidimensional datasets. In density-biased sampling, the probability that a given point will be included in the sample depends on the local density of the dataset. We propose a general technique for density-biased sampling that can factor in user requirements to sample for properties of interest, and can be tuned for specific data mining tasks. This allows great flexibility, and improved accuracy of the results over simple random sampling. We describe our approach in detail, we analytically evaluate it, and show how it can be optimized for approximate clustering and outlier detection. Finally we present a thorough experimental evaluation of the proposed method, applying density-biased sampling on real and synthetic data sets, and employing clustering and outlier detection algorithms, thus highlighting the utility of our approach.

1 Introduction

Clustering and detecting outliers in large multidimensional data sets are important data analysis tasks. A variety of algorithmic techniques have been proposed for their solution. The inherent complexity of both problems however is high, and the application of known techniques on large data sets is time and resource consuming. Many data intensive organizations, such as AT&T, collect lots of different data sets of cumulative size in the order of hundreds of gigabytes over specific time frames. Application of known techniques to analyze the data collected even over a single time frame, would require a large resource commitment and would be time consuming. Additionally, from a data analysis point of view not all data sets are of equal importance. Some data sets might contain clusters or outliers, others might not. To date, one has no way of knowing that, unless one resorts to executing the appropriate algorithms on the data. A powerful paradigm to improve the efficiency of a given data analysis task is to reduce the size of the dataset (often called *Data Reduction* [3]), while keeping the accuracy loss as small as possible. The difficulty lies in designing the appropriate approximation technique for the specific task and datasets.

In this paper, we propose to use *biased sampling* as a data reduction technique to efficiently provide approximate solutions to data mining operations such as cluster and outlier detection on large data set collections. Biased sampling is a generalization of uniform random sampling, which has been used extensively to speed up the execution of data mining tasks. In uniform random sampling, every point has the same probability to be included in the sample [26]. In contrast, in *biased* sampling, the probability that a data point is added to the sample is different from point to point. In our case, this probability depends on the value of prespecified data characteristics and the specific analysis requirements. Once we derive a sampling probability distribution that takes into account the data characteristics and the analysis objectives, we can extract a biased sample from the dataset and apply standard algorithmic techniques on the sample. This allows the user to find approximate solutions to different analysis tasks, and gives a quick way to decide if the dataset is worthy of further exploration.

Our main observation is that the density of the dataset contains enough information to derive the desired sampling probability distribution. To identify the clusters when the level of noise is high, or to identify and collect reliable statistics for the large clusters, we can oversample the dense regions. To identify all possible clusters, we can oversample the sparser regions. It now becomes much more likely to identify sparser or smaller clusters. On the other hand, we can make sure that denser regions in the original dataset continue to be denser in the sample, so that we do not loose any of the dense clusters. To identify outliers we can sample the very sparse regions.

We propose a new, flexible, and tunable technique to perform density-biased sampling. In this technique we calculate, for each point, the density of the local space around the point, and we put the point into the sample with probability that is a function of this local density. For succinct (but approximate) representation of the density of a dataset, we choose kernel density estimation methods. We apply our technique to the problems of cluster and outlier detection, and report our experimental results. Our approach has the following benefits: (1) **Flexibility:** The probability that a

*This research has been supported by NSF CAREER Award 9984729 and NSF IIS-9907477.

point is included in the sample can be optimized for different data mining tasks, (2) **Effectiveness:** Density-biased sampling can be significantly more effective than random sampling as a data reduction technique, and (3) **Efficiency:** It makes use of efficient density estimation techniques (requiring one dataset pass), and requires one or two additional passes to collect the biased sample.

In section 2 we analytically evaluate the benefits of the proposed approach over random sampling and present our approach. In section 3 we discuss the issues involved when combining our sampling technique with clustering and outlier detection algorithms in order to speed up the corresponding tasks. In section 4 we present the results of a detailed experimental evaluation using real and synthetic datasets, analyzing the benefits of our approach for various parameters of interest.

1.1 Related Work

Clustering and outlier detection are very important and widely used techniques, with a large variety of techniques proposed for their solution. However, the approach closer to ours is the technique of Palmer et. al. [22]. They developed an algorithm to sample for clusters by using density information, under the assumption that clusters have a zipfian distribution. Their technique is designed to find clusters when they differ a lot in size and density. The technique undersamples dense areas and oversamples sparse areas. Their approach tightly couples the extraction of density information and the actual sampling. It uses a hash based approach, and the quality of the sample degrades with collisions implicit to any hash based approach. The approach presented herein, decouples density estimation and biased sampling. Our use of kernels for density estimation is orthogonal to our framework, which is independent on any assumptions about cluster distribution. Any density estimation technique can be used instead and we suggest a hash free approach to eliminate the impact of collisions to density estimation. In addition, our approach is tunable for different cases, such as when there is a lot of noise, or when we are looking for outliers. We present an experimental comparison between the techniques in section 4.

2 A Flexible and Efficient Technique for Density Biased Sampling

Sampling [27, 29] is a well recognized and widely used statistical technique. In the context of clustering in databases, both partitional [20] and hierarchical [8] clustering techniques employ uniform random sampling to improve the efficiency of the techniques. Toivonen [28] examined the problem of using sampling during the discovery of association rules. Sampling has also been recently successfully applied in query optimization [6, 5], as well as approximate query answering [7, 1].

The focus of this paper is in the design of alternative forms of sampling that can be used to expedite data mining tasks such as cluster and outlier detection. A natural question arises regarding the potential benefits of such sampling techniques. In other words, why is uniform random sampling not sufficient for such a purpose? Guha et. al, [8] presented a theorem linking the sample size with the probability that a fraction of the cluster is included in the sample. Let D be a data set of size n, and u a cluster of size $|u|$. Guha et. al, consider a cluster u to be included in the sample, when more than $\phi * |u|$ points from the cluster are in the sample, for $0 \leq \phi \leq 1$. Then the sample size s, required by uniform random sampling, to guarantee that u is included in the sample with probability less than δ, $0 \leq \delta \leq 1$ is:

$$s \geq \phi * n + \frac{n}{|u|} \log(\frac{1}{\delta}) + \frac{n}{|u|} \sqrt{(\log(\frac{1}{\delta}))^2 + 2 * \phi * |u| * \log(\frac{1}{\delta})}$$

For example, in order to guarantee with probability 90% that a fraction $\phi = 0.2$ of a cluster with 1000 points is in the sample, we need to sample 25% of the dataset. Clearly, one must be willing to include a very large fraction of the dataset in the sample, in order to provide good guarantees with uniform random sampling. This requirement, however, defies the motivation for sampling.

Instead of employing uniform sampling, let us assume that we sample according to the following rule. Let $\pi_1, \ldots, \pi_n$ be the points in the dataset. Let u be a cluster. Assume that we had the ability to bias our sampling process according to the following rule, R:

$$P(\pi_i \; is \; included \; in \; the \; sample) = \begin{cases} p \; if \; \pi_i \in u, \\ 1 - p \; otherwise \end{cases}$$

for $0 \leq p \leq 1$. We can state the following theorem (the proof appears in [15]):

Theorem 1 *Let D be a data set of size n, u a cluster of size $|u|$. A cluster u is included in the sample, when more than $\phi * |u|$ points from the cluster are in the sample, for $0 \leq \phi \leq 1$. Let s_R be the sample size required by sampling according to R, to guarantee that u is included in the sample with probability less than δ, $0 \leq \delta \leq 1$ and s the sample size required by uniform random sampling to provide the same guarantee. Then: $s_R \leq s \; if \; and \; only \; if \; p \geq \frac{|u|}{n}$*

Theorem 1 provides a means to overcome the limitations of uniform random sampling since we can assure, with the same probability, that the same fraction of the cluster is in the sample, by taking a sample of smaller size. The serious limitation, however, is that we have no clear way to bias our sampling towards points in the clusters, since we do not know these points a priori.

We observe that knowledge of the probability density function of the underlying dataset, can provide enough information to locate those points. Clusters are located in dense areas of the dataset and outliers in sparse areas. By adapting our sampling, according to the density of the dataset we can bias our sampling probabilities towards dense areas in the case of clusters, and sparse areas in the case of outliers. All that is needed, is a mapping function that maps density to sampling probability, in a way that depends on the objective of our sampling process. For the case of clusters, this function can be as simple as sampling according to the probability density function of the underlying data set. For outliers, it can be as simple as sampling according to the inverse of this function. Since the sample is biased towards specific areas it has the potential, due to Theorem 1, to require smaller sample sizes to give certain guarantees.

2.1 Density Estimation Techniques

An efficient and accurate technique for estimating the density of a dataset is paramount for the success of our method. There is a number of methods suggested in the literature employing different ways to find the density estimator for multi-dimensional datasets. These include, computing multi-dimensional histograms [23] [6] [16] [2], using various transforms, like the wavelet transformation [30] [19], SVD [23] or the discrete cosine transform [17] on the data, using kernel estimators [4], as well as sampling [21] [18] [10].

Although our biased-sampling technique can use any density estimation method, kernel density estimators provide the best solution because they are accurate and can be computed efficiently. Work on query approximation [9] and in the statistics literature [25], [24] shows that kernel functions always estimate the density of the dataset more accurately than using a random sample of the dataset. Finding a kernel density estimator can be done in one dataset pass. The technique is presented in detail in [9].

2.2 The Proposed Biased Sampling Technique

In this section we outline our technique for density-biased sampling. Let D be a d-dimensional dataset with n points. For simplicity we assume that the space domain is $[0,1]^d$, otherwise we can scale the attributes. Let $f(x_1,\ldots,x_d) : [0,1]^d \rightarrow \mathcal{R}$ be a density estimator for the dataset D. That is, for a given region $R \subset [0,1]^d$, the integral $\int_R f(x_1,\ldots,x_d)dx_1\ldots dx_d$ is approximately equal to the number of points in R. We want to produce a biased sample of D, with the following properties:

Property 1 *The probability that a point* $\mathbf{x}$ *in* D *will be included in the sample is a function of the local density of the data space around the point* p.

Property 2 *The expected size of the sample should be* b*, where* b *is a parameter set by the user.*

Let's consider the case of uniform sampling first. A point in D, will be in the sample with probability $\frac{b}{n}$. By the definition of f, if $f(x_1,\ldots,x_d) > 1$ then the local density around the point $(x_1,\ldots,x_d)$ is larger than the average density of the space, defined as

$$\frac{\int_{[0,1]^d} f(x_1,\ldots,x_d)dx_1\ldots dx_d}{Volume([0,1]^d)} = 1$$

In density biased sampling, we want the probability of a given point $(x_1 \ldots x_d)$ to be included in the sample, to be a function of the local density of the space around the point $(x_1,\ldots,x_n)$. This is given by the value of $f(x_1\ldots x_d)$. In addition we want the technique to be tunable, so we introduce the function $f'(x_1,\ldots,x_d) = (f(x_1,\ldots,x_d))^a$ for some $a \in \mathcal{R}$. The basic idea of the sampling technique is to sample a point $(x_1,\ldots,x_d)$ with probability that is proportional to $\frac{b}{n}f'(x_1,\ldots,x_d)$. The value of a controls the biased sampling process. Let $k = \sum_{(x_1,\ldots,x_n)\in D} f'(x_1,\ldots,x_n)$. We define $f^*(x_1,\ldots,x_d) = \frac{n}{k}f'(x_1,\ldots,x_d)$.
Thus f^* has the additional property: $\sum_{(x_{i1}\ldots x_{ik})\in D} f^*(x_{i1}\ldots x_{ik}) = n$.
Having defined f^* thus, we include each point $(x_1,\ldots,x_d) \in D$ in the sample with probability $\frac{b}{n}f^*(x_1,\ldots,x_d)$.

Given a dataset D, a density estimator f for D, a value of α and a sample size b set by the user,

1. Compute in one pass the value of k, where $k = \sum_{(x_1,\ldots,x_n)\in D} f'(x_1,\ldots,x_n)$.
2. Add each point $\mathbf{x} \in D$ in the sample with probability $\frac{b}{|D|}f^*(\mathbf{x})$.

Figure 1. The Proposed Biased Sampling Technique

Our technique for density-biased sampling is shown in Figure 1. It can be shown that this algorithm satisfies properties 1 and 2. The proof of this and the following lemmas appears in [15]. One of the main advantages of the proposed technique is that it is very flexible, and can be tuned for different applications. Using different values of a we can oversample either the dense regions (for $a > 0$), or the sparse regions (for $a < 0$). We consider the following cases:

1. $a = 0$: each point is included in the sample with probability $\frac{b}{n}$, and therefore we have uniform sampling.

2. $0 < a$: In this case $f^a(\mathbf{x}) > f^a(\mathbf{y})$ if and only if $f(\mathbf{x}) > f(\mathbf{y})$. It follows that the regions of higher density are sampled at a higher rate than regions of lower density. In particular, regions with density that is higher than the average density in the data space, are sampled at a rate that is higher, than the uniform sampling rate, and regions where the density is lower than the data space average, are sampled at a rate that is lower than the uniform sampling rate.

3. $0 > a$: In this case the opposite is true. Since $f^a(\mathbf{x}) > f^a(\mathbf{y})$ if and only if $f(\mathbf{x}) < f(\mathbf{y})$, the regions with lower density than average are sampled at a higher rate than uniform, and the regions of higher density are sampled with a lower rate than uniform.

4. $a = -1$: there is the same expected number of sample points in any two regions of the same volume.

The sampling technique we described has an additional fundamental property, namely it preserves, in the sample, the relative densities of different regions in D. If regions A and B in D are dense, they should appear dense in the sample too. Moreover, if A has higher density than B then the same should be true in the sample. We state and prove the following lemma:

Lemma 1 *If $a > -1$, relative densities are preserved with high probability. That is, if a region A has higher density than region B in the data set, then, with high probability (depending on the size of the regions), the same is true for the sample.*

Lemma 1 allows us to use density-biased sampling to discover small clusters that are dominated by very large and very dense clusters. In the case of $a > 0$, dense areas are oversampled, but areas of lower density are sampled enough, such that their density relative to the sampled dense areas is preserved. By setting $-1 < a < 0$, we oversample areas that are less dense, but still sample the very dense areas enough so that they remain dense in the sample.

In our presentation of the sampling technique we choose for simplicity to separate the computation of the biased sampling probability with the actual sampling step. It is possible to integrate both steps in one, thus deriving the biased sample in a single pass over the database. In this case however we only compute an approximation of the sampling probability. The details of this integration are presented in [15].

3 Using Biased Sampling for Cluster and Outlier Detection

We describe how density biased sampling can be integrated with cluster and outlier detection algorithms. Our emphasis is on evaluating how biased sampling can be used in conjunction with "off the shelf" clustering and outlier detection algorithms to provide accurate approximate solutions to these problems.

3.1 Clustering

We run clustering algorithms on the density-biased sample to evaluate the accuracy of the proposed technique. To cluster the sample points we use a hierarchical clustering algorithm. The algorithm starts with each point in a separate cluster and at each step merges the closest clusters together. The technique is similar to CURE [8] in that it keeps a set of points as a representative of each cluster of points.

The hierarchical clustering approach has several advantages over K-means or K-medoids algorithms. It has been shown experimentally that it can be used to discover clusters of arbitrary shape, unlike K-means or K-medoids techniques, which typically perform best when the clusters are spherical [8]. It has also been shown that a hierarchical clustering approach is insensitive to the size of the cluster. On the other hand K-means or K-medoids algorithms tend to split up very large clusters, and merge the pieces with smaller clusters. The main disadvantage of the hierarchical approach is that the run time complexity is quadratic. Running a quadratic algorithm even on a few thousand data points can be problematic. A viable approach to overcome this difficulty, is to reduce the size of the dataset in a manner that preserves the information of interest to the analysis without harming the overall technique.

A hierarchical clustering algorithm that is running off the biased sample is in effect an approximation algorithm for the clustering problem on the original dataset. To this respect, our technique is analogous to the new results on approximation clustering algorithms [12] [11], since these algorithms also run on a (uniform random) sample to efficiently obtain the approximate clusterings. The techniques are not directly comparable, however, because these algorithms are approximating the optimal K-medoids partitioning, which can be very different from the hierarchical clusterings we find.

We note here that we can also use biased sampling in conjunction with K-means or K-medoids algorithms if this is desirable. However, K-means, or K-medoids algorithms optimize a criterion that puts equal weight to each point in the dataset. For example, for the K-means clustering algorithm, assuming K clusters the optimization criterion is minimization of the following function: $\sum_{\text{(cluster i)}} \sum_{\text{(j} \in \text{i)}} dist(j, m(i))$, where for the K-means algorithm $m(i)$ is the mean of cluster i, and for the K-medoids algorithm $m(i)$ is the medoid of cluster i. To use density biased sampling in this case, we have to weight the sample points with the inverse of the probability that each was sampled.

3.2 Distance Based Outliers

Distance based outliers were introduced by Knorr and Ng [13, 14].

Definition 1 *[13] An object O in a dataset D is a $DB(p, k)$ outlier if at most p objects in D lie at distance at most k from O.*

Knorr and Ng [13] show that this definition is compatible with empirical definitions of what is an outlier, and it also generalizes statistical definitions of outliers. Note that in the definition of a $DB(p, k)$-outlier, the values of k and p are parameters set by the user. The number of objects that can be at distance at most k can also be specified as a fraction fr of the dataset size, in which case $p = fr\ |D|$. The technique we present has the additional advantage that it can estimate the number of $DB(p, k)$-outliers in a dataset D very efficiently, in one dataset pass. This gives the opportunity for experimental exploration of k and p, in order to derive sufficient values for the specific dataset, and the task at hand.

To facilitate the description of our algorithm, we also define the function $N_D(O, k)$ to be the number of points that lie at distance at most k from the point O in the dataset D.

The basic idea of the algorithm is to sample the regions on which the data point density is very low. Since we are dealing with datasets with numerical attributes, we assume, in the description of the algorithm that the distance function between points is the Euclidean distance. However different distance metrics (for example the L_1 or Manhattan metric) can be used equally well.

Let D be a dataset with $|D| = n$ points and d numerical dimensions. Let f be a density estimator of the dataset D. Given input parameters k (the maximum distance of neighboring points) and p (the maximum number of the dataset points that can be at distance at most k away), we compute, for each point O, the expected number of points in a ball with radius k, that is centered at the point. Thus we estimate the value of $N_D(O, k)$ by

$$N'_D(O, k) = \int_{Ball(O,\ k)} f(x_1, \ldots, x_d) dx_1 \ldots dx_d$$

We keep the points that have smaller expected number of neighbors than $O(k)$. These are the likely outliers. Then, we make another pass over the data, and verify the number of neighbors for each of the likely outliers. For mode details about this algorithm we refer to [15].

4 Experiments

We present an experimental evaluation of our approach. Due to space constraints, we present here only a small subset of our evaluation and we refer to [15] for more results. We run clustering and outlier detection tasks in both synthetic and real datasets and we intent to explore the following issues: (i) determine whether biased sampling indeed offers advantages over uniform random sampling for cluster and outlier detection, and quantify this advantage, (ii) determine the accuracy of the approximate solution that we obtain if we run cluster and outlier detection algorithms on only the biased sample, as opposed to running on the entire dataset.

We use the same algorithm for clustering the density-biased sample and the uniform random sample in order to make the comparison objective. We used a hierarchical clustering algorithm based on CURE [8], but not the original implementation by the authors. In this algorithm each cluster is represented by a set of points that have been carefully selected in order to represent the shape of the cluster (well scattered points). Therefore, when the hierarchical algorithm is running on the uniform random sample, the clusters it produces are the ones CURE would have found. We compare these results with the ones obtained when the hierarchical algorithm is run on a density-biased sample (with different values of a). We also include the BIRCH [31] clustering algorithm in the comparison since it is generally considered as one of the best clustering methods proposed so far for very large databases. We allow BIRCH to use as much space as the size of the sample to keep the CF-tree (Section 2) which describes the dataset. Although we constrain the size of the CF-tree, we allow BIRCH to work on the entire dataset for clustering, that is we do not run BIRCH on our samples. By comparing with BIRCH, we intent to explore the effectiveness of our summarization technique for identifying clusters to that of BIRCH.

4.1 Description Of Datasets

Synthetic Datasets We used a synthetic data generator to produce two to five dimensional datasets. The number of clusters is a parameter, varied from 10 to 100 in our experiments. Due to space constraints we report only the results for 10 clusters. The rest of our results with datasets containing larger number of clusters are qualitatively similar. Each cluster is defined as a hyper-rectangle, and the points in the interior of the cluster are uniformly distributed. The clusters can have non-spherical shapes, different sizes (number of points in each cluster) and different average densities. We also generate and use in our experiments datasets with noise as follows: Let D be a dataset of size $|D|$ containing k synthetically generated clusters. We add $l * |D|, 0 \leq l \leq 1$ uniformly distributed points in D as noise and we say that D contains $f_n = 100 * l$ noise. We vary f_n from 5% to 80% in our experiments. In addition we used synthetic datasets from [8] (see Figure 3.)

Geospatial Datasets We also used real life datasets from spatial databases. The NorthEastern dataset represents 130000 postal addresses in the North East States. The California dataset represents postal addresses in California and has a size of 62553 points. Both datasets contain two dimensional points. Another dataset is the Forest Cover Dataset (59000 points) from the UCI KDD archive[1]. This was obtained from the US Forest Service (USFS).

4.2 Implementation Details And Parameter Setting.

For the density approximation function, we use the kernel density estimator technique. In our experiments we use $\beta = 1000$ sample points to initialize the kernel centers and we use the Epanechnikov kernel function. We performed extensive experiments [15] evaluating the impact of the number of kernels in our approach. The value of 1000 gave very good results across a wide range of datasets with different statistical characteristics and sizes and we propose the use of this value in practice. We extract two types of samples:

- *Uniform Sampling:* We extract a uniform random sample of size b, by first reading the size, N, of the dataset and then sequentially scanning the dataset and choosing a point with probability $\frac{b}{N}$. Thus, the expected size of the sample is b.

- *Biased sampling:* We implemented the algorithm described in section 2.2. There are three parameters affecting this algorithm: (a) the number of sample points that we use for the kernel estimation (ks) (b) the number of sample points b, and (c) the parameter α, that controls the rate of sampling for areas with different densities. We experiment with these parameters in the sequel.

  We also compare out technique with the technique of [22][2]. Since this technique is designed specifically to identify clusters of different sizes and densities, by oversampling sparse regions and undersample sparse regions, we run the comparison for this case only.

We employ the following clustering algorithms:

- *BIRCH:* We used an implementation of the BIRCH algorithm, provided to as by the authors [31]. We set the parameter values as suggested in [31]. Thus, we set the page size to 1024 bytes, initial threshold to 0 and input range to 1000.

- *Hierarchical Clustering algorithm:* We follow the suggestion of the experimental study in [8] and we set the parameters accordingly. Parameter α (the shrink factor) is set to 0.3, we use one partition and the default value for the representative points is 10.

4.3 Clustering Experiments

Running time experiments In the first set of experiments we investigated the effect of the number of kernels and the size of the dataset to the running time of our algorithm. Not suprisingly, our algorithm scales linearly to the number of kernels and the size of the datasets. For mores results we refer to [15].

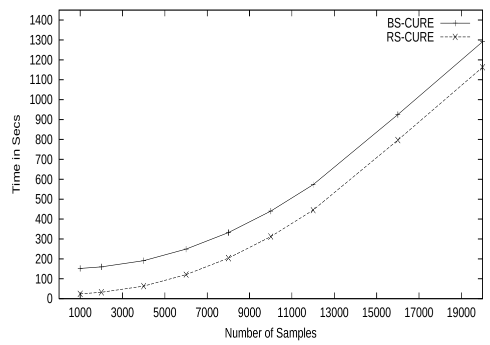


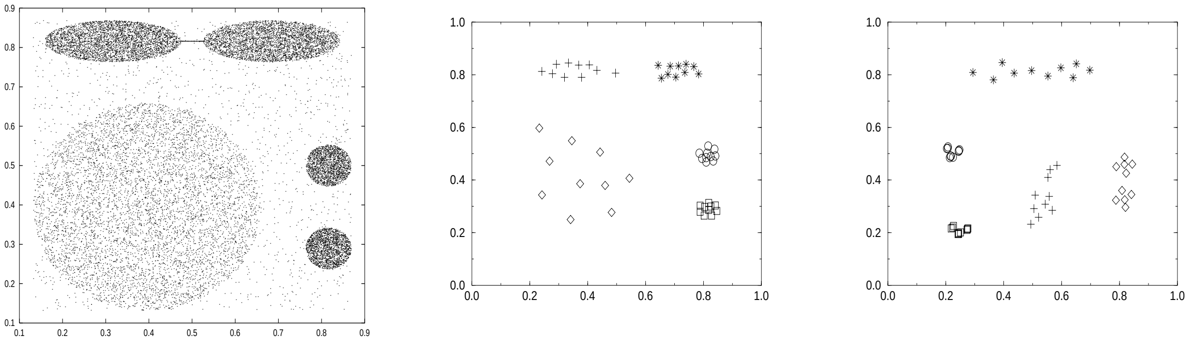
Figure 2. Running time of the clustering algorithm.

In Figure 2 we plot the total running time of the clustering algorithm for biased and random sampling. We used a dataset with 1 million points and 1000 kernels. Note that the running time of the hierarchical algorithm is quadratic. Also note that the time we spend for building the density estimator and for doing another pass for the actual sampling, is more than offset by the time we save because we can run a quadratic algorithm on a smaller sample. For example, if we run the algorithm on a 1% uniform random sample, the running time is $O(n + (n/100)^2)$. If we run the clustering algorithm on a 0.5% biased sample, the running time is $O(2n + (n/200)^2)$. For one million points, our method with a 0.4% or 0.8% sample is faster than the hierarchical algorithm with 0.8% or 1.1% uniform random sample respectively.

Biased Sampling vs Uniform Random Sampling In this experiment we explore the effects of using density-biased sampling as opposed to uniform random sampling in order to identify clusters. We used a dataset (dataset1) used in [8]. This dataset has 5 clusters with different shapes and densities. We draw a uniform random sample as well as a biased sample with $\alpha = -0.5$, both of size 1000, and we run the hierarchical clustering method on both samples.

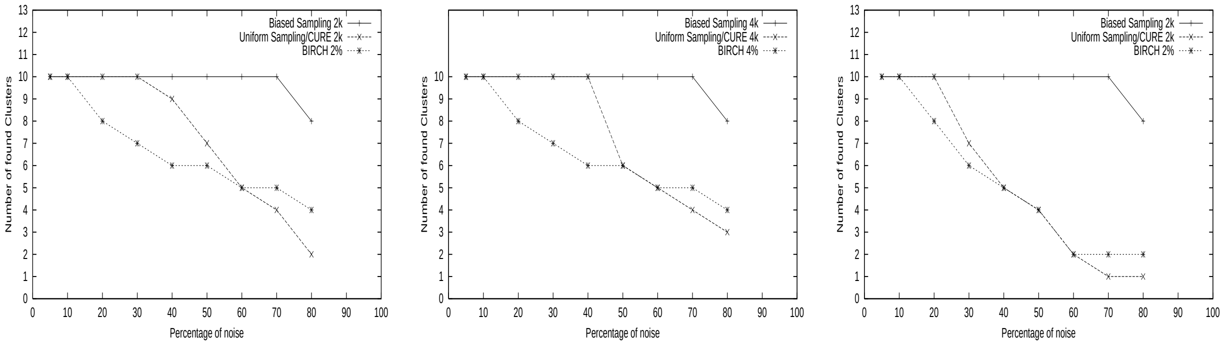
[1]available from kdd.ics.uci.edu/summary.data.type.html

[2]the code is available in: www.cs.cmu.edu/People/crpalmer/dbs-current.tar.gz



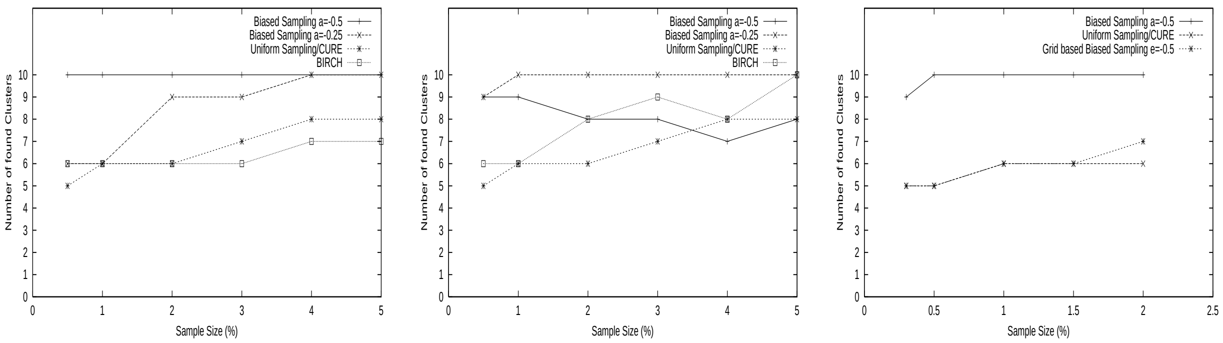
(a) Dataset1 [8] (b) Biased Sample (c) Uniform Sample

Figure 3. Clustering using the hierarchical algorithm, samples of size 1000 points



(a) 2 Dim, Sample Size 2% (b) 2 Dim, Sample Size 4% (c) 3 Dim, Sample Size 2%

Figure 4. Varying Noise in 2 and 3 dimensions (ranging f_n from 5% to 80%)



(a) 2 dimensions, Noise 10% (b) 2 dimensions, Noise 20% (c) 5 dimensions, Noise 10%

Figure 5. Finding clusters of variable density, in the presence of noise.

In Figure 3(a) we present the dataset and in figure 3(b) we plot 10 representatives of each cluster found using the biased sample. The centers of all 5 clusters have been discovered. Figure 3(c) plots 10 representatives of each cluster found using the uniform random sample. Note that the large cluster is split into three smaller ones, and two pairs of neighboring clusters are merged into one. Clearly, the uniform random sample fails to distinguish the clusters.

It is worth noticing that by increasing the size of the uniform random sample, there exists a sample size (well above 2000 points in size) on which the hierarchical technique can now correctly identify all clusters. This is consistent with our analytical observation in theorem 1. A much larger sample (in this case twice the size of the biased sample) is required by uniform random sampling to achieve the same effects.

The Effects Of Noise On Accuracy In this set of experiments we show the advantage of density-biased sampling with $a > 0$ over uniform sampling in the presence of noise in the dataset.

We employ our sampling technique and uniform random sampling on noisy datasets, and we compare the results of the two sampling algorithms. We use synthetic datasets in two and three dimensions, and we vary the noise by varying f_n from 5% to 80% points. The datasets (without noise added) contain 100K points distributed into 10 clusters of different densities. Since the objective is to find clusters in the presence of strong noise, we oversample the dense regions in biased sampling. In the experiments, we use two different values for a, 0.5 and 1 (note that if $a > 0$, we oversample the dense areas).

We run experiments using sample sizes of 2% and 4% in 2-dimensions and using a sample size of 2% in 3-dimensions. Figures 4(a)(b)(c) summarize the results for $a = 1$. The results where similar in the $a = 0.5$ case; they are available elsewhere [15]. To evaluate the results of the hierarchical algorithm, a cluster is found if at least 90% of its representative points are in the interior of the same cluster in the synthetic dataset. Since BIRCH reports cluster centers and radiuses, if it reports a cluster center that lies in the interior of a cluster in the synthetic dataset, we assume that this cluster is found by BIRCH.

The results show that using biased sampling, clusters can be identified efficiently even in the presence of heavy noise. It is evident that biased sampling can identify all clusters present in the dataset, even with $f_n = 70\%$ and misses one cluster at $f_n = 80\%$ noise. The accuracy of uniform random sampling drops quickly as more noise is added. The advantage of biased sampling is even more pronounced in three dimensions as shown in figure 4(c). The effectiveness of biased sampling to identify clusters is robust to the increase of dimensionality.

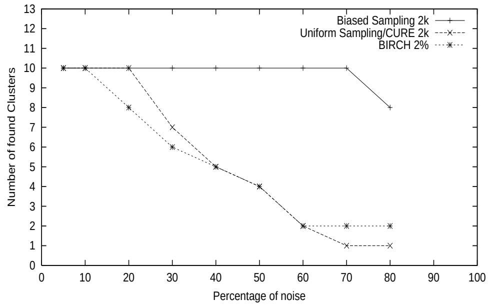


Figure 6. Varying Noise in 3-dimensions, (f_n from 5% to 80%), Sample Size 2%

Clusters With Variable Densities In this set of experiments we show the advantage of biased sampling with $0 > a$, in the case that the size and density of some clusters is very small in relation to other clusters. The difficulty in sampling datasets with this distribution lies in the fact that if the clusters are small, there may be too few points from a cluster in the sample, and therefore the algorithm may dismiss these points. We evaluate the effectiveness of biased sampling in discovering all the clusters in the dataset, when the density of the clusters varies by a factor of 10. We use 2 to 5-dimensional synthetic datasets containing 100000 and 10 clusters. Figure 5(a)(b)(c) shows some of the results.

We run biased sampling with $a = -0.25$ and $a = -0.5$, and therefore we sample dense areas proportionally less than sparser areas. However, since we run for values of a that are between -1 and 0, we still expect the larger, denser clusters to be found, and in addition we make the smaller clusters proportionally larger in the sample. The lower the overall level of noise, the smaller the value of a should be to obtain high accuracy. For example, for 10% noise, the best results are achieved for $a = -0.5$, for 20% noise, $a = -0.25$. This result is intuitive, because when a is small, as the noise level increases, proportionally more noise points are added to the sample, than when a is larger. Interestingly, BIRCH is not affected as much by the noise, as it is affected by the relative size of the clusters. In all cases, BIRCH discovers between 6 and 8 out of the 10 clusters, and misses most small clusters all together.

We also run experiments using the biased sampling technique of [22] (using the code that is available in the web). This technique has been designed for this situation, and samples sparse regions at a higher rate in order to identify small clusters. It uses a grid to partition the data space. Since the number of cells can be very large, grid cells are hashed into a hash table. In the experiments we allow up to 5MB of memory to store the hash table. To make a fair comparison, we use this technique to obtain a sample, and then we use the hierarchical algorithm to find the clus-

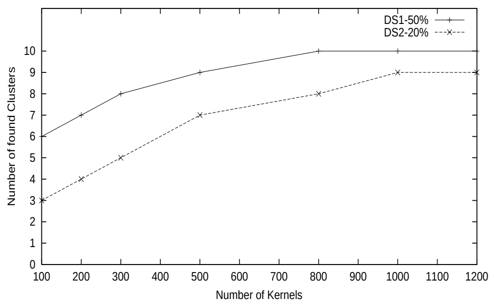


Figure 7. Varying the number of Kernels

ters in the sample. This grid based technique works well in lower dimensions and no noise, but it is not very accurate at higher dimensions and when there is noise in the dataset (Figure 5(c)) even when we allow 5MB of memory to be used. In 5 dimensions and 10% noise its performance is better than uniform random sampling but worse that our technique which estimates the density more accurately.

Varying The Sample Size In both previous experiments we also varied the size of the sample. We observe that both density-biased and uniform sampling do not improve much when the size of the sample is increased beyond a certain value. An interesting observation is that this occurs much earlier when using density-biased sampling than uniform sampling, an observation that is consistent with our analytical expectations. In our experiments, this value was generally 1K points for density-biased sampling, and 2K points for uniform sampling.

Varying The Number of Kernels In another set of experiments we varied the number of kernels and therefore the quality of the density approximation. In the Figure 7 we present results for two datasets. The dataset $DS1 - 50\%$ contains 100k points with 10 clusters of the same size and an additional 50% noise. The second dataset, $DS2 - 20\%$, consists of 100k points of 10 clusters with very different sizes plus 20% noise. We used biased sampling with a equal to 1.0 and -0.25 respectively and 500 sample points. Notice that in the second case the accuracy of the density estimation is more important because there is a very large variation in the densities of the clusters. In both cases when increasing the number of kernels from 100 to 1200 the quality of the clustering results initially improves considerably but the rate of the improvement is reduced continuously.

Real Datasets We also used two real life datasets to test the proposed algorithm. In *NorthEast* Dataset we were able to indentify three clusters that correspond to the three largest metropolitan areas, New York, Philadelphia, and Boston. Random sampling fails to identify these high density areas because there is also a lot of noise, in the form of widely distributed rural areas and smaller population centers Similarly, density-biased sample is more effective in identifying large clusters in the California dataset as well. The graphs are omitted for brevity and are available in [15].

4.4 Practitioner's Guide

We summarize the results of our experimental evaluation, and we offer some suggestions on how to use density-biased sampling for cluster detection in practice.

- For noisy datasets, setting $a = 1$ allows reliable detection of dense clusters.
- If the dataset is not expected to include noise, setting $a = -0.5$ allows detection of very small or very sparse clusters.
- Setting the number of kernels in the density estimation $\beta = 1000$ allows accurate estimation of the dataset density.
- Setting the size of the density-biased sample at 1% of the dataset size offers very accurate results, and is a good balance between accuracy and performance.

4.5 Outlier detection experiments

We implemented the algorithm described in section 3.2 and ran experiments to indentify distance based outliers in synthetic and real datasets. However due to space constraints we do not report our results here but they are available in [15]. Briefly we can say that in almost all cases the algorithm finds all the outliers with at most two dataset passes plus the dataset pass that is required to compute the density estimator.

5 Conclusions

We have presented a biased sampling technique based on density to expedite the process of cluster and outlier detection. We have analytically demonstrated the potential benefits of biased sampling approaches, and we have presented an efficient algorithm to extract a biased sample from a large dataset capable of factoring in user requirements. Moreover we have proposed efficient algorithms to couple our sampling approach with existing algorithms for cluster and outlier detection. Finally, we have experimentally evaluated our techniques using "off the shelf" clustering and outlier detection algorithms and we have experimentally demonstrated the benefits of our approach.

This work raises several issues for further study. Although we demonstrated that biased sampling can be an asset in two specific data mining tasks, it is interesting to explore the use of sampling in other data mining tasks as well. Several other important tasks, like classification, construction of decision trees, or finding association rules, can potentially benefit both in construction time and usability by the application of similar biased sampling techniques suitably adjusted to take the specifics of these mining tasks into account.

6 Acknowledgments

We wish to thank Piotr Indyk, H. V. Jagadish and S. Muthukrishnan for sharing their thoughts and time during the course of this work, and Daniel Barbara for providing the code for hierarchical clustering, V. Ganti for providing the code for BIRCH and C. Palmer for making his code for biased sampling available.

References

[1] S. Acharya, P. Gibbons, V. Poosala, and S. Ramaswamy. Join Synopses For Approximate Query Answering. *In Proc. of SIGMOD*, pages 275–286, June 1999.

[2] S. Acharya, V. Poosala, and S. Ramaswamy. Selectivity Estimation In Spatial Databases. *Proceedings of SIGMOD, Philladelphia PA*, June 1999.

[3] D. Barbara, C. Faloutsos, J. Hellerstein, Y. Ioannidis, H. V. Jagadish, T. Johnson, R. Ng, V. Poosala, K. Ross, and K.C. Sevcik. The new jersey data reduction report. *Data Engineering Bulletin*, September 1996.

[4] B. Blohsfeld, D. Korus, and B. Seeger. A Comparison of Selectivity Estimators for Range Queries on Metric Attributes. *In Proc. of ACM SIGMOD, June 1999*.

[5] S. Chaudhuri, R. Motwani, and V. Narasayya. Random Sampling For Histogram Construction: How Much Is Enough? *In Proc. of ACM SIGMOD, Seattle WA*, pages 436–447, June 1998.

[6] S. Chaudhuri, R. Motwani, and V. Narasayya. On Random Sampling Over Joins. *In Proc. of ACM SIGMOD, Philadelphia, PA*, pages 263–274, June 1999.

[7] P.B. Gibbons and Y. Matias. New Sampling-Based Summary Statistics for Improving Approximate Query Answers. *In Proc. of the 1998 ACM SIGMOD, June 1998*.

[8] S. Guha, R. Rastogi, and K. Shim. CURE: An Efficient Clustering Algorithm for Large Databases. *Proceedings of ACM SIGMOD*, pages 73–84, June 1998.

[9] D. Gunopulos, G. Kollios, V. Tsotras, and C. Domeniconi. Approximating Multi-Dimensional Aggregate Range Queries over Real Attributes. *In Proc. of ACM SIGMOD, Dallas, TX*, May 2000.

[10] P. Haas and A. Swami. Sequantial Sampling Procedures for Query Size Estimation. *Proceedings of ACM SIGMOD, San Diego, CA*, pages 341–350, June 1992.

[11] Piotr Indyk. Sublinear time algorithms for metric space problems. *In Proc. of 31st STOC, 1999*.

[12] Piotr Indyk. A sublinear-time approximation scheme for clustering in metric spaces. *In Proc. of 40th FOCS, 1999*.

[13] E. Knorr and R. Ng. Algorithms for Mining Distance Based Outliers in Large Databases. *Proceedings of VLDB, New York*, pages 392–403, August 1998.

[14] E. Knorr and R. Ng. Finding Intensional Knowledge of Distance Based Outliers. *Proceedings of VLDB, Endiburgh Scotland*, pages 211–222, September 1999.

[15] G. Kollios, D. Gunopulos, N. Koudas, and S. Berchtold. Biased Sampling For Cluster And Outlier Detection. *AT&T Labs Tech Report & UCR Tech Report*.

[16] F. Korn, T. Johnson, and H. Jagadish. Range Selectivity Estimation for Continuous Attributes. *In Proc. of 11th Int. Conf. on SSDBMs, 1999*.

[17] J. Lee, D. Kim, and C. Chung. Multi-dimensional Selectivity Estimation Using Compressed Histogram Information. *In Proc. of the 1999 ACM SIGMOD, June 1999*.

[18] R. J. Lipton, J. F. Naughton, and D. A. Schneider. Practical Selectivity Estimation Through Adaptive Sampling. *Proceedings of ACM SIGMOD, Atlantic City, NJ*, pages 1–11, May 1990.

[19] Y. Matias, J. Scott Vitter, and M. Wang. Wavelet-Based Histograms for Selectivity Estimation. *In Proc. of the 1998 ACM SIGMOD, June 1998*.

[20] Raymond T. Ng and Jiawei Han. Efficient and Effective Clustering Methods for Spatial Data Mining. *Proceedings of VLDB*, pages 144–155, September 1994.

[21] F. Olken and D. Rotem. Random Sampling from database Files:A Survey. *In Proc. of 5th Int. Conf. on SSDBMs, Charlotte, N.C., 1990*.

[22] C. Palmer and C. Faloutsos. Density Biased Sampling: An Improved Method for Data Mining and Clustering. *In Proc. of ACM SIGMOD, Dallas, TX*, May 2000.

[23] V. Poosala and Y. Ioannidis. Selectivity Estimation Without the Attribute Value Independence Assumption. *Proceedings of VLDB, Athens Greece*, pages 486–495, August 1997.

[24] D. Scott. Multivariate density estimation: Theory, practice and visualization. *Wiley and Sons, 1992*.

[25] B.W. Silverman. Density Estimation for Statistics and Data Analysis. *Monographs on Statistics and Applied Probability, Chapman & Hall 1986*.

[26] S.K. Thompson. Sampling. *John Wiley and Sons, New York, 1992*.

[27] S.K. Thompson and G.A.F. Seber. Adaptive Sampling. *John Wiley and Sons, New York, 1996*.

[28] H. Toivonen. Sampling Large Databases For Association Rules. *Proceedings of VLDB, Mumbai India*, pages 134–145, August 1996.

[29] J.S. Vitter. Random sampling with a reservoir. *In ACM Trans. on Mathematical Software, 11(1), 1985*.

[30] J.S. Vitter, M. Wang, and B. R. Iyer. Data Cube Approximation and Histograms via Wavelets. *In Proc.of the 1998 ACM CIKM, Nov 1998*.

[31] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An Efficient Data Clustering Method for Very Large Databases. *Proceedings of ACM SIGMOD, Montreal Canada*, pages 103–114, June 1996.