An Information-Theoretic Approach to Detecting Changes in Multi-Dimensional Data Streams

Tamraparni Dasu Shankar Krishnan Suresh Venkatasubramanian Ke Yi

AT&T Labs – Research Duke University

Abstract

An important problem in processing large data streams is detecting changes in the underlying distribution that generates the data. The challenge in designing change detection schemes is making them general, scalable, and statistically sound. In this paper, we take a general, information-theoretic approach to the change detection problem, which works for multidimensional as well as categorical data. We use relative entropy, also called the Kullback-Leibler distance, to measure the difference between two given distributions. The KL-distance is known to be related to the optimal error in determining whether the two distributions are the same and draws on fundamental results in hypothesis testing. The KL-distance also generalizes traditional distance measures in statistics, and has invariance properties that make it ideally suited for comparing distributions.

Our scheme is general; it is nonparametric and requires no assumptions on the underlying distributions. It employs a statistical inference procedure based on the theory of bootstrapping, which allows us to determine whether our measurements are statistically significant. The scheme is also quite flexible from a practical perspective; it can be implemented using any spatial partitioning scheme that scales well with dimensionality. In addition to providing change detections, our method generalizes Kulldorff's spatial scan statistic, allowing us to quantitatively identify specific regions in space where large changes have occurred.

We provide a detailed experimental study that demonstrates the generality and efficiency of our approach with different kinds of multidimensional datasets, both synthetic and real.

1 Introduction

We are collecting and storing data in unprecedented quantities and varieties—streams, images, audio, text, metadata descriptions, and even simple numbers. Over time, these data streams change as the underlying processes that generate them change. Some changes are spurious and pertain to glitches in the data. Some are genuine, caused by changes in the underlying distributions. Some changes are gradual and some are more precipitous.

We would like to detect changes in a variety of settings:

Data cleaning: Spurious changes affect the quality of the data. In this context, we might ask, "Given a gold standard D_0, is the dataset at hand, D_1, different? If so, can we isolate the sections that are different?" The gold standard can be something that we have seen in the past or an ideal state that we wish the data to be in. Some examples of spurious changes are missing values, default values erroneously set, discrepancy from an expected stochastic process, *etc.* A detailed discussion of data glitches and the data quality problem in general can be found in [11]. Detecting changes and departures from the benchmark is important because many critical decisions depend on the data being accurate. Corporations spend billions of dollars every year managing and cleaning their business operations databases. Very often, glitches in business and data processes can result in subtle shifts that cannot be detected manually or by aggregate analysis.

Data modeling: Similarly, many corporations, scientific projects and government agencies rely on complex models to summarize, monitor, analyze, predict and forecast. Shifts in underlying probability distributions can cause these models to fail spectacularly. While much effort is spent in building, validating and putting these models in place, there is very little done in terms of detecting changes. Sometimes, the models might be too insensitive to change, reflecting the change only after a big shift in the distributions.

Alarm systems: Some changes are transient, and yet important to detect; network traffic monitoring is one of the best examples of a situation where it is hard to posit realistic underlying models, and yet some anomaly detection approach is needed to detect (in real time) shifts in network behavior along a wide array of dimensions.

1.1 Desiderata

Any change detection mechanism has to satisfy a number of criteria to be viable. Crucial features are:

- **Generality:** Applications for change detection come from a variety of sources, and the notion of "change" varies from setting to setting. Thus, a general approach to defining change is very important.
- **Scalability:** Any approach must be scalable to very large datasets, and be able to adapt to streaming settings as well if necessary. There are sophisticated methods one can apply to small data sets, including methods like logistic regression [26] and proportional hazards [8]; these typically require multiple passes over the data and thus are not suitable for large or streaming datasets. An important aspect of scalability is dealing with multidimensional data. A change detection scheme must be able to work with multidimensional data directly in order to capture spatial relationships and correlations.
- **Statistical soundness:** One of the key problems with a change detection mechanism is determining the significance of an event. By connecting a change detection mechanism to statistically rigorous approaches for significance testing, we ensure that any changes reported by the method can be evaluated objectively, allowing the method to be used for a diverse set of applications. This also reflects the above desire for generality.

A natural approach to detecting change in data is to model the data via a distribution. One can then compare representative statistics like means or fit simple models like linear regression to capture variable interactions. Such approaches aim to capture some simple aspects of the joint distribution (e.g. centrality, relationships between some specific attributes) rather than the entire multivariate distribution.

The parametric approach is very powerful when data is known to come from specific distributions. Parameter estimation is a very well-studied area, and a wide variety of methods can be used to estimate distributions precisely. Moreover, if distributional assumptions hold, parametric methods require very little data in order to work successfully. However, the generality requirement for change detection is violated; data that one typically encounters may not arise from any standard distribution, and thus parametric approaches are not applicable.

Approaches in the database community (where data scarcity is not an issue!) have thus focused on *nonparametric* methods. Loosely, nonparametric methods are those that *make no distributional assumptions on the data*. Statistical tests that have been used in this setting include the Wilcoxon test and the Kolmogorov-Smirnov test, the multinomial test, and variants. Here, as before, the approach used is to compute a *test statistic* (a scalar function of the data), and compare the values computed to determine whether a change has occurred.

1.2 An Information-theoretic Approach

The above tests attempt to capture a notion of *distance* between two distributions. A measure that is one of the most general ways of representing this distance is the *relative entropy* from information theory, also known as the *Kullback-Leibler* (or KL) distance. The KL-distance has many properties that make it ideal for estimating the distance between distributions:

- Given a set of data that we wish to fit to a distribution in a family of distributions, the maximum likelihood estimator is the one that *minimizes the KL-distance to the true distribution.*

- The KL-distance generalizes standard tests of difference like the *t-test*, *chi-square* and the Kulldorff spatial scan statistic; the t-test is equivalent to the KL distance between two normal distributions, the chi-square function is the first term in the Taylor expansion of the KL distance function, and the Kulldorff spatial scan statistic is the KL-distance between a data distribution and an underlying Poisson distribution.

- An optimal classifier that attempts to distinguish between two distributions p and q will have a false positive (or false negative) error proportional (in the limit) to an exponential in the KL-distance from p to q (the exponent is negative, so the error decreases as the distance increases)[7, §12.8].

- It is an example of an *α-divergence*, one among a class of distributional distances that include the Hellinger distance (but not ℓ_1 or ℓ_2) and have various geometric invariance properties.

Intuitively, the KL-distance between distributions behaves like Euclidean distance in $\mathbb{R}^n$; here the "points" are distributions that lie on the simplex, rather than $\mathbb{R}^n$. Using the KL-distance allows us not only to measure the distance between distributions, but attribute a meaning to this value. Further, an information-theoretic distance can be defined independent of the inherent dimensionality of the data, and is even independent of the spatial nature of the data, when one invokes the *theory of types.* Thus, we can isolate the definition of change from the data representation itself, cleanly separating the computational aspects of the problem from the distance estimation itself.

There are advantages to the information-theoretic approach from a computational perspective as well. Tests like the Wilcoxon and Kolmogorov-Smirnov cannot be easily extended to data in more than a single dimension. This is principally because these tests rely on data being ordered (they are rank-based statistics), and thus in two or more dimensions, the lack of a unique ordering renders them ineffective (for further discussion of this issue, see the section in [31, §14.8]). The work by Kifer, Ben-David and Gehrke [21] proposes a modification of the Kolmogorov-Smirnov test that in principle could be extended to higher dimensions, but technical challenges need to be overcome to make this work; indeed, they leave this as an open question in their paper.

The KL-distance has long been utilized to measure distances between distributions. In machine learning and classification, many problems are being viewed through the lens of information theory, and the KL-distance and related measures are being exploited in many situations where the traditional measure of choice was ℓ_2 or some other metric. We discuss these developments in Section 7.

1.3 Bootstrapping and Statistical Significance

The choice of distance function used to determine change is one aspect of the change detection problem. Another is statistical significance. How do we determine whether the measure of change returned is significant or not? A statistical approach poses the question by specifying a null hypothesis (in this case, that change has *not* occurred), and then asking "How likely is it that the measurement could have been obtained under the null hypothesis?". The smaller this value (the so-called "p-value"), the more likely it is that the change is significant[1].

For parametric tests, significance testing is fairly straightforward. For some nonparametric tests also, significance testing can be performed by exploiting certain special properties of the tests used (e.g.,[21]). However, if we wish to determine statistical significance in more general settings, we need a more general approach to determining confidence intervals.

The *bootstrap method*, first developed by Efron, is a *data-centric* approach to determining confidence intervals for inferences on data. Intuitively the bootstrap method determines, by repeated sampling (with or without replacement) from the data, whether a specific measurement on the data is significant or not. The use of the bootstrap is particularly relevant to streaming settings, as one of the key features of bootstrap methods is to make strong inferences from small datasets.

Bootstrap methods satisfy the goal of generality that we stated earlier; they also provide statistical soundness and are well suited for use with nonparametric methods (Ganti *et al.* [15] have used bootstrapping in their data mining system FOCUS; see Section 7 for more details). We will use them extensively in this paper.

1.4 Our Contributions

In this paper, we present a general information theoretic approach to the problem of multi-dimensional change detection. Our specific contributions are:

- The use of the Kullback-Leibler distance as a measure of change in multi-dimensional data, and a demonstration of the statistical guarantees it provides.
- The use of bootstrap methods to establish the statistical significance of the distances we compute.
- An efficient algorithm for change detection on streaming data that scales well with dimension. Our approach is "plug-and-play": any spatial data partitioning scheme that scales well with dimension may be used; we are not dependent on any particular choice of structure, like quad trees or k-d trees.
- An approach for identifying subregions of the data that have the highest changes. This approach is based on the observation that the Kulldorff scan statistic, often used for identifying spatial regions of high density in a population, is a special case of the Kullback-Leibler distance.
- Empirical demonstration (both on real and synthetic data) of the accuracy of our approach.

[1] Other authors refer to the "critical region": essentially the interval of values where the result is deemed significant.

Our methods are framed in terms of the comparison between a reference and a test dataset. This abstraction lends itself naturally to stream-based change detection (where the two sets are two windows on the stream) as well other kinds of dynamic data settings.

We present a birds-eye view of our scheme in Section 2, followed by a more detailed discussion on information-theoretic fundamentals (Section 3) and the theory of bootstrapping (Section 4). Algorithmic details are presented in Section 5, followed by a detailed experimental study in Section 6. Related work is discussed in Section 7.

2 An Overview

We start with a birds-eye view of our change detection scheme, presenting the mechanics of the approach while omitting proofs and justifications.

Let $x_1, x_2, \ldots$ be a stream of objects. For the purpose of this paper, we will assume that each x_i is a point in $\mathbb{R}^d$. A *window* $W_{i,n}$ denotes the sequence of points ending at x_i of size n: $W_{i,n} = (x_{i-n+1}, \ldots, x_i)$. We will drop the subscript n when the context is clear. Distances are measured between distributions constructed from points in two windows W_t and $W_{t'}$.

In general, using different-sized windows allows one to detect changes at different scales, and we can run our scheme with different window sizes in parallel for this purpose. A standard approach that we will use is to choose window sizes that increase exponentially (having sizes $n, 2n, 4n$, and so on). Each window size can be processed independently. Thus in what follows, we will describe a change detection scheme for a fixed window size n, and run one instance of this scheme for each window size. Note that we assume that the time a point arrives is its time stamp; we do not consider streams where data might arrive out of (time) order.

We consider two sliding window models. In the *adjacent windows* model, the two windows that we measure the difference between are W_t and W_{t-n}, where t is the current time. In the *fix-slide windows* model [21], we measure the difference between a fixed window W_n and a sliding window W_t. The first model better captures the notion of "rate of change" at the current moment, while the second one is more suitable for change detection when gradual changes may cumulate over time, and thus the adjacent window model will repeatedly only detect small changes.

Each window W_t defines an empirical distribution F_t, and we compute the distance $d_t = d(F_t, F_{t'})$ from F_t to $F_{t'}$, where t' is either $t - n$ or n depending on the sliding window model we are using. Section 3 describes the computation of d_t in more detail.

This distance is our measure of the difference between the two distributions (and thus the two windows). The next step is to determine whether this measurement is statistically significant. Formally, we assert the null hypothesis

$$H_0 : F_t = F_{t'}$$

and wish to determine the probability of observing the value d_t if H_0 is true.

To determine this, we use bootstrap estimates. Using methods that we describe in more detail in Section 4, we generate a set of k *bootstrap estimates* $\hat{d}_i, i = 1 \ldots k$. These estimates form an empirical distribution from which we construct a critical region (d_{hi}, ∞). If d_t falls into this region, we consider that H_0 is invalidated. Since we test H_0 at every time step, in order to improve robustness, we only signal a change after we have seen γn distances larger than d_{hi} in a row, where γ is a small constant defined by the user. The idea is that a true change should be more persistent than a false alarm, which might be transient, and we term γ the *persistence factor*. If no change has been reported, we update the windows and repeat the procedure. The update algorithm will be discussed in Section 5.

The above algorithm is summarized in Algorithm 2.1.

Algorithm 2.1 Change detection algorithm (for a fixed window size)

$t \leftarrow 2n$;
$t' \leftarrow n$;
Construct windows W_t and $W_{t'}$;
Compute $d_t = d(F_t, F_{t'})$;
Compute bootstrap estimate $\hat{d}_i, i = 1, \ldots, k$ and critical region (d_{hi}, ∞);
$c \leftarrow 0$;
while not at end of stream **do**
  if $d_t > d_{\text{hi}}$ **then**
    $c \leftarrow c + 1$;
    if $c \geq \gamma n$ **then**
      Signal `change`;
      Start over;
    end if
  else
    $c \leftarrow 0$;
  end if
  Slide window W_t (and $W_{t'}$ if required);
  Update d_t;
end while

3 Information-theoretic Distances

The measure we use to compare distributions is the *Kullback-Leibler* distance, also called the *relative entropy* [7, Sec 2.3].

Definition 3.1 *The* relative entropy *or* Kullback-Leibler distance *between two probability mass functions $p(x)$ and $q(x)$ is defined as*[2]

$$D(p\|q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)},$$

where the sum is taken (in the discrete setting) over the atoms of the space of events $\mathcal{X}$.

If the distributions p and q are normally distributed with common variance σ^2 and means μ_p, μ_q, then $D(p\|q) = c \cdot d^2$, where $d = (\mu_p - \mu_q)/\sigma$ is the measure of difference used in Student's *t-test* [32]. Another interesting property of the KL-distance is its relation to the χ^2 statistic [32], where $\chi^2(p, q) = \sum_x (p(x) - q(x))^2/q(x)$. It is not hard to show that χ^2 is twice the first term in the Taylor expansion of $D(p\|q)$, i.e.,

$$D(p\|q) = \frac{1}{2}\chi^2 + \cdots .$$

The relative entropy is a member of a more general class of distances called the *Ali-Silvey* distances (or f-divergences), defined as

$$F(p \mid q) = E_p[f(q/p)],$$

[2] All logarithms are base 2 in this paper.

where f is a strictly convex function. The ratio (q/p) is called the *likelihood ratio*, and its importance derives from the Neyman-Pearson theorem, which shows that a decision region based on the likelihood ratio has optimal error probability when doing hypothesis testing. We describe this connection in more detail in Appendix A. Informally, if we are in a hypothesis testing scenario where we have to determine whether sampled data arises from distributions F or G, then the probability of misclassifying the data is proportional to $2^{-D(F\|G)}$.

Although the relative entropy is referred to as a distance, it is important to note that it is not symmetric, and thus not a metric. This lack of symmetry can be inconvenient in certain settings. As a result, many variants have been proposed that seek to address its lack of symmetry and non-metric nature, while preserving its relation to classifier error rate. Examples (which are also Ali-Silvey distances) include the Chernoff distance, the Bhattacharya distance and the Jensen-Shannon divergence. However, all of the above measures have other disadvantages, and have shown no conclusive superiority in experimental settings. In the rest of this paper, we will focus on the KL-distance, while noting that any of the above measures (since they can be expressed in terms of the KL-distance) can be used in our framework.

3.1 Constructing a Distribution from a Stream

The relative entropy is defined on a pair of probability mass functions. How do we map sequences of points to distributions? The answer lies in the *theory of types*, due to Csiszár and Körner [10]. Let $\mathbf{w} = \{a_1, a_2, \ldots a_n\}$ be a multiset of letters from a finite alphabet $\mathcal{A}$. The *type* $P_{\mathbf{w}}$ of $\mathbf{w}$ is thus vector representing the relative proportion of each element of $\mathcal{A}$ in $\mathbf{w}$

$$P_{\mathbf{w}}(a) = \frac{N(a \mid \mathbf{w})}{n}.$$

Thus each set $\mathbf{w}$ defines a empirical probability distribution $P_{\mathbf{w}}$. For each set, we compute the corresponding empirical distribution, and compute the distance between the two distributions, viewed as mass functions. Gutman[16] has shown that types retain (asymptotically) the same classifier properties as true distributions. Moreover, the empirical distributions are the maximum likelihood estimators of the true distribution.

For d-dimensional data, the "alphabet" will consist of a letter for each leaf of the quad tree used to store the data (to be discussed in details in Section 5), with the count being the number of points in that cell. One further advantage of the use of types is that categorical data can be processed in exactly the same way (with a letter associated with each value in the domain).

One problem with this approach is that the ratio p/q is undefined if $q = 0$. A simple correction suggested by Krichevsky and Trofimov [24] replaces the estimate $P_{\mathbf{w}}(a)$ by the estimate

$$P_{\mathbf{w}}(a) = \frac{N(a|\mathbf{w}) + 0.5}{n + |\mathcal{A}|/2}.$$

To summarize, given two windows W_1, W_2, and their associated multisets of letters $\mathbf{w_1}, \mathbf{w_2}$ constructed from the alphabet defined over quad tree leaf cells, the distance from W_1 to W_2 is

$$D(W_1\|W_2) = \sum_{a\in\mathcal{A}} P_{\mathbf{w_1}}(a) \frac{P_{\mathbf{w_1}}(a)}{P_{\mathbf{w_2}}(a)}.$$

4 Bootstrap Methods and Hypothesis Testing

The *bootstrap method*, first developed by Efron, is a method for determining the significance (or *p-value*) of a test statistic, as well as eliminating bias and improving confidence intervals when doing statistical testing. Our proposed approach for estimating change relies on estimating the relative entropy between two empirical distributions. Empirical likelihood is a nonparametric, data driven method and uses a likelihood based approach for statistical inference. Its relationship to bootstrap along with the advantages and disadvantages of each are discussed in [28]. Viewed as a test statistic, the relative entropy has an (unknown exact) distribution over an (unknown) data distribution. Thus, standard approaches (which require at least one of the unknowns to be known) are ineffective in this context[3]. The bootstrap method can be used to estimate the standard error, bias, and confidence intervals of a test statistic. Its connection to hypothesis testing comes via confidence intervals.

In the hypothesis testing scenario pertaining to data change, the null hypothesis asks whether two distributions F, G are identical

$$H_0 : F = G.$$

Once an observation is made (in this case the calculation of $\hat{d} = D(\hat{F} \| \hat{G})$, where $\hat{F}$ and $\hat{G}$ are the empirical distributions of F and G), the *achievable significance level (ASL)* of the observation is the likelihood that $\hat{d}$ arises naturally under H_0, i.e.,

$$\Pr_{H_0}(\hat{d}^* \geq \hat{d})$$

where $\hat{d}^*$ is a random variable measuring d under H_0.

Notice that $\hat{d}^* = 0$ is an equivalent statement of the null hypothesis. Thus, if we can determine an interval $[0, d_{\text{hi}}]$ such that $\hat{d}$ lies in this range with probability $1 - \alpha$, then this is equivalent to an ASL of α (more details can be found in [14, Chapter 15]). This method is known as the *percentile* method.

The bootstrapping procedure works as follows: given the empirical distributions $\hat{P}$ derived from the counts P (see Section 3), we *sample* k sets $S_1, \ldots S_k$, each of size $2n$. Treating the first n elements S_{i1} as coming from one distribution F, and the remaining n elements $S_{i2} = S_i - S_{i1}$ as coming from the other distribution G, we compute bootstrap estimates $\hat{d}_i = D(S_{i1} \| S_{i2})$.

Once we fix the desired ASL α, we choose the $(1 - \alpha)$-percentile of these bootstrap estimates as d_{hi}. We call (d_{hi}, ∞) the critical region, if $\hat{d} > d_{\text{hi}}$, the measurement is statistically significant and invalidates H_0.

4.1 Sample Sizes

Bootstrap procedures are a form of Monte Carlo sampling over an unknown distribution. Asymptotically, the bootstrap sample distribution approaches the true underlying distribution. The crucial question here is: how fast?

Given a confidence point $\hat{\theta}[\alpha]$ such that $\Pr(\theta \leq \hat{\theta}[\alpha]) = \alpha$, it is said to be *first-order accurate* if

$$\Pr(\theta \leq \hat{\theta}[\alpha]) = \alpha + O(n^{-\frac{1}{2}}),$$

and *second-order accurate* if

$$\Pr(\theta \leq \hat{\theta}[\alpha]) = \alpha + O(n^{-1}),$$

[3]Many of these approaches are based on computing the mean of a (unknown) data distribution. The mean has a (known) Gaussian distribution, enabling significance testing.

where n is the number of samples.

Confidence intervals based on the percentile method are first-order accurate. It is possible to use more complex procedures to determine confidence points that give second-order accurate confidence bounds; we do not employ these methods in this paper, as they do not improve the quality of our results significantly.

The number of bootstrap samples needed in practice depends on the kind of distribution being sampled. Experiments and normal approximation suggest that about 500–1000 samples works well for the percentile method.

5 Data Structures

From now on, we will assume that the data points in the streams lie in a d-dimensional hypercube. In order to maintain the KL-distance between two empirical distributions, we need a way of defining the "types", i.e., a space partitioning scheme that subdivides the space into cells. In principle any space partitioning scheme, for example a quad tree or a k-d-tree, works in our framework, but we would like to use a structure that scales well with the size and dimensionality of the data, and produces "nicely shaped" cells at the same time. The square cells induced by a quad tree are intuitively good, but its 2^d fan-out might hurt its scalability in high dimensions. On the other hand, a k-d-tree scales well with dimensionality, but it might generates very skinny cells. The structure that we propose, called a *kdq*-tree, is combination of these two space partitioning schemes that has the advantages of both structures.

We will describe the structure in two dimensions; generalization to high dimensions will be obvious. A *kdq*-tree is a binary tree, each of whose nodes is associated with a box. The box associated with the root v is the entire unit square, which is then divided into two halves by a vertical cut passing through its center. The two smaller boxes are then associated with the two children of the root v_l and v_r. We then construct the trees rooted at v_l and v_r recursively, and as we go down the tree, the cuts alternate between vertical and horizontal. We stop the recursion if either the number of points in the box is below τ, or all the sides of the box have reached a minimum length δ, where τ and δ are user specified parameters.

The following properties follow from a simple analysis of the structure.

Proposition 5.1 *For a kdq-tree built on n points in d dimensions,*

1. *it has at most $O(dn \log(\frac{1}{\delta})/\tau)$ nodes;*
2. *its height is at most $O(d \log(\frac{1}{\delta}))$;*
3. *it can be constructed in time $O(dn \log(\frac{1}{\delta}))$;*
4. *the aspect ratio of any cell is at most 2.*

We can see that the *kdq*-tree's size scales linearly as the dimensionality and the size of data, at the same time it generates nicely shaped cells. It is also very cheap to maintain the counts associated with the nodes, as the cost is proportional to the height of tree.

We build the *kdq*-tree on the first window W_1, and will use the cells induced by this tree as the types to form the empirical distributions for both W_1 and W_2 until a change has been detected, at which point we rebuild the structure. The same structure is also used to compute the bootstrap estimates.

Maintaining the KL-distance. Let P_v (resp. Q_v) be the number of points from the set W_1 (resp. W_2) that are inside the cell associated with the leaf v of the *kdq*-tree. We would like to maintain the KL-distance between $P = \{P_v\}$ and $Q = \{Q_v\}$:

$$\begin{aligned} D(P\|Q) &= \sum_v \frac{P_v + 1/2}{|W_1| + L/2} \log \frac{(P_v + 1/2)/(|W_1| + L/2)}{(Q_v + 1/2)/(|W_2| + L/2)} \\ &= \log \frac{|W_2| + L/2}{|W_1| + L/2} + \frac{\sum_v (P_v + 1/2) \log \frac{P_v+1/2}{Q_v+1/2}}{|W_1| + L/2}, \end{aligned}$$

where L is the number of leaves in the *kdq*-tree. Since $|W_1|, |W_2|$ and L are readily known, we only need to maintain

$$\tilde{D}(P\|Q) = \sum_v (P_v + 1/2) \log \frac{P_v + 1/2}{Q_v + 1/2}.$$

Since the counts P_v, Q_v can be updated in $O(d \log(\frac{1}{\delta}))$ time per time step, the KL-distance can also be maintained incrementally in the same time bound.

Identifying regions of greatest difference. The *kdq*-tree structure for KL-distance based change detection can also be used to identify the most different regions between the two datasets, once a change has been reported. The idea is to maintain a special case of the KL-distance at each node (internal or leaf) v of the *kdq*-tree. This special case is the *Kulldorff spatial scan statistic* [25], which is defined at a node v as

$$D_K(v) = P_v \log \frac{P_v}{Q_v} + (|W_1| - P_v) \log \frac{|W_1| - P_v}{|W_2| - Q_v} - |W_1| \log \frac{|W_1|}{|W_2|}.$$

Note that it is simply the KL-distance between W_1 and W_2 when there are only two bins: B_v and its complement $\overline{B_v}$. Kulldorff's statistic basically measures how the two datasets differ *only with respect to the region associated with* v. Specifically, it measures the log likelihood ratio of two hypotheses: (1) that the region v has a different density from the rest of space, and (2) that all regions have uniform density. Note that this statistic can be easily maintained as it depends only on P_v and Q_v.

However, one limitation of a such an approach is that we can only compute the Kulldorff statistic in regions that correspond to the tree nodes. It remains an interesting open problem to efficiently maintain the (general) region maximizing the Kulldorff's statistic. The algorithm of Neill and Moore [27] is designed for static data and it does not appear that their scheme can be made dynamic, or even streaming.

6 Experiments

To demonstrate the generality of the KL distance-based change detection scheme, we conducted a series of experiments with various kinds of datasets. The datasets consist primarily of synthetic data, which are generated according to some standard distributions, continuous as well as discrete. An advantage of using these simulated datasets is that we can control change events and see how the change detections mechanisms react to different types of changes. We also included telephone usage data obtained from a major telephone company's network, to see how our scheme performs on a more realistic dataset. In this paper, we focus on experiments using data streams in two or higher dimensions, for which our method appears to be the first streaming change detection scheme in the database literature that has a sound statistical basis. Finally, we also present some results in one dimension comparing our scheme with some of the previously known 1D change detection techniques.

In all the experiments, we use the following default values for some of the parameters, unless specified otherwise.

Parameter	Symbol	Value
Minimum side length of a cell	δ	2^{-10}
Maximum number of points in a cell	τ	100
Persistence factor	γ	5%
Achievable significance level (ASL)	α	1%
Number of bootstrap samples	k	500

Table 1: Default values for some parameters in the experiments.

6.1 Evaluating the Accuracy of the Kullback-Leibler Distance

We know that the Kullback-Leibler distance measures how different one distribution is from another, and equals zero if the two distributions are identical. In order to see how it varies as the amount of difference changes, we did several test runs with simulated data streams generated according to 2D normal distributions.

In these test runs, each data stream consists of $N = 500,000$ points, each of which is generated independently from a 2D normal distribution with certain parameters. These points are divided into groups of 50,000. Within one group, the parameters stay the same, which means that a change happens every 50,000 points. We use the adjacent windows model, where we slide two adjacent windows, both of size $n = 20,000$, through the stream, and measure the KL distances between the two sliding windows. More precisely, we report the distance $D(W_t \| W_{t-n})$ between windows W_t and W_{t-n} for $t = 2n, 2n+1, \ldots, N$. This intuitively measures the "rate of change" of the stream at time t.

For this series of tests, we did not use bootstrapping to detect change, since our goal was merely to demonstrate the behavior of the Kullback-Leibler distance on changing data.

Test 1: Varying the mean μ. In the first data stream, the standard deviation and correlation are fixed at $\sigma_1 = \sigma_2 = 0.2, \rho = 0$, and the mean is set to be $\mu_1 = \mu_2 = 0.2 + 0.06(i-1)$ for the ith group, $i = 1, \ldots, 10$, namely the center of the 2D Gaussian gradually moves from $(0.2, 0.2)$ towards $(0.8, 0.8)$. Figure 1 shows how the Kullback-Leibler distance changes as we slide the two windows. Observe that a change in the mean is usually easily detected, and the amount of difference only depends on the difference between the means. This is the reason why the peaks almost have the same height in Figure 1.

Test 2: Varying σ. In the second stream, we fix the mean and correlation at $\mu_1 = \mu_2 = 0.5, \rho = 0$, but vary the standard deviation with $\sigma_1 = \sigma_2 = 0.1 + 0.02(i-1)$ for group i. The results are shown in Figure 2. Again, observe that a change in the standard deviation is also relatively easy to identify, although the amount of change is not uniform for different σ values (intuitively since as σ increases, the distribution gets more and more concentrated, and so the relative difference is small).

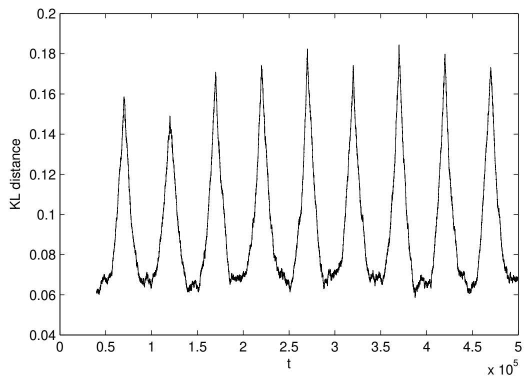

Figure 1: The KL distance between adjacent windows in a stream with varying (μ_1, μ_2). Changes occur every $50,000$ points.

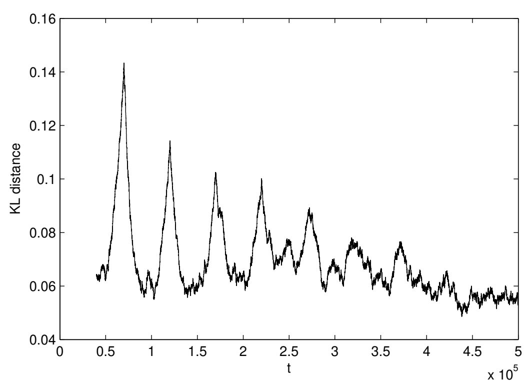

Figure 2: The KL distance between adjacent windows in a stream with varying (σ_1, σ_2). Changes occur every $50,000$ points.

Test 3: Varying the correlation ρ. In the third stream, the mean and variance is fixed at $\mu_1 = \mu_2 = 0.5, \sigma_1 = \sigma_2 = 0.2$, while the correlation is $\rho = 0.08(i-1)$ for group i, i.e., the two coordinates of the data point start with being independent, and then become more and more positively correlated. The results are shown in Figure 3. In this case, changes in the correlation are reflected more subtly in the Kullback-Leibler distance, but we can still detect peaks every $50,000$ points, when the parameter changes. It is worth noting that such changes cannot be detected by projecting the points onto the axes and using any 1D change detection scheme.

Test 4: An empirical case study. Our next experiment is on data obtained from a major telephone company's network. The data represent telecommunication usage in two major urban centers. The data collected include total usage, time-of-day and length-of-usage related attributes. Each urban region includes entities that use landlines and wireless phones, and what one would like to determine is whether the two urban areas differ significantly in terms of land-line vs. wireless line usage. In this particular example, the urban centers chosen are known to have significant differences in their usage profiles.

Each dataset consists of 120,000 three-dimensional data points. We concatenate them into a

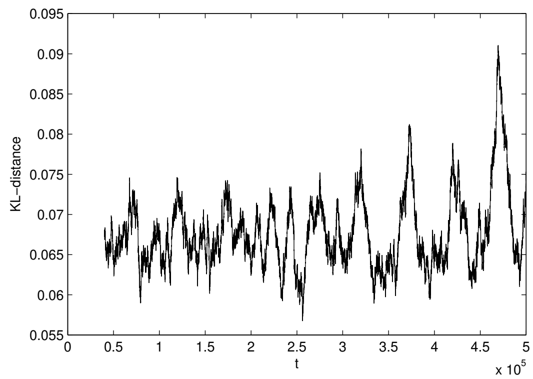

Figure 3: The KL distance between adjacent windows in a the stream with varying ρ. Changes occur every $50,000$ points.

stream of 240,000 points, and slide two adjacent windows of size 20,000 over this 3D data stream in the same way as above. From Figure 4 we can clearly see a large spike as we move from the first dataset to the the second, indicating a significant difference between these two datasets.

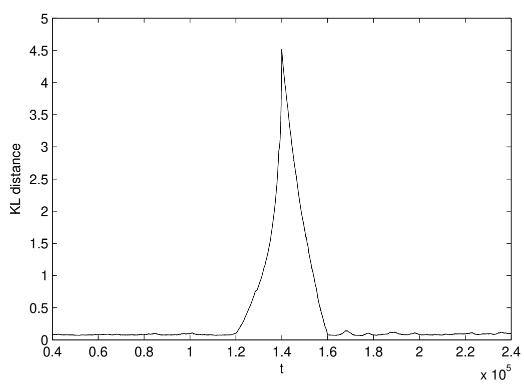

Figure 4: The KL distance between adjacent windows in a 3D data stream obtained from telephone usage in two urban centers. The change between urban centers occurs at $t = 120,000$.

6.2 Evaluating the Change Detection Method

In the tests runs above, we have demonstrated how the Kullback-Leibler distance fluctuates as the underlying distribution changes. In order to determine if a given measurement is statistically significant enough to qualify as a "change", we plug in the bootstrapping method and construct a critical region $(d_{\mathrm{hi}}, \infty)$, as described in Section 4. When the measured distance falls into this region in γn consecutive time steps, we report that a change has happened, as in Algorithm 2.1.

We adopt the fix-slide windows model of Kifer *et al.* [21], where we fix the first window at W_n and slide the second window starting at W_{2n}. There are two reasons for keeping the first window fixed: Firstly, point-to-point changes might be slow, but over time they may eventually accumulate and become significant. Keeping a fixed reference window allows us to capture such gradual changes. Secondly, if we keep both windows moving and pass the point of change, we will miss the change completely! If we keep the first window fixed, we have a chance of capturing the change at a later time. Under the rationale "better late than never", a late detection is still considered to be better than a false negative. More precisely, following [21], we say a detection is late if $t > t_c + 2n$, where t_c is the time of the actual change, and t is the time when the change is reported.

Each simulated data stream consists of $N = 5,000,000$ two-dimensional points, generated from a distribution F with parameters $p_1, \ldots, p_k$. The N data points are divided into groups of $50,000$ points. The distribution F and all of its parameters are fixed at all times except for a particular set of changing parameters $p_{r_1}, p_{r_2}, \ldots$, which vary from group to group. We specify a step size Δ to control the amount of change. When we reach a new group, for each changing parameter p_{r_i}, we choose a random number in $[-\Delta, -\Delta/2] \cup [\Delta/2, \Delta]$ uniformly and add it to p_{r_i}. If p_{r_i} falls out a predefined allowable region $[p_{\min}, p_{\max}]$, we pick another random number and repeat. In this way, we create a total of $5,000,000/50,000 - 1 = 99$ changes. The window size is fixed at $n = 10,000$ unless otherwise specified.

In all our experiments, we report four statistics: the number of changes correctly detected, the number of changes detected late, the number of false positives, and the number of misses changes (false negatives). Note that there is an inherent tradeoff between false positives and false negatives, and that our use of synthetic data is what makes evaluating false negatives possible. In statistical terms, the tradeoff is between the power (1− false positive rate) and the sensitivity (1− false negative rate) of the test.

Data sources. Using the procedure above, we generated three groups of streams from normal distributions. In the first group, we fix the standard deviation $\sigma_1 = \sigma_2 = 0.2$ and correlation $\rho = 0.5$, but vary the mean μ_1 and μ_2. They both start at 0.5, and then perform independent random walks in $[0.2, 0.8]$ with step size Δ as described above. We chose Δ to be $0.01, 0.02$ and 0.05, respectively, to see how our change detection reacts to different amounts of changes. We use $M(\Delta)$ to denote the streams in this group with step size Δ. In the second group of streams, we fix $\mu_1 = \mu_2 = 0.5$ and $\rho = 0.5$, while vary the standard deviation. Both σ_1 and σ_2 start at 0.2, and then perform random walks within $[0, 0.4]$ with step size $\Delta = 0.01, 0.02$, and 0.05, respectively. We use $D(\Delta)$ to denote the streams in this group. Finally, in the third group, we fix $\mu_1 = \mu_2 = 0.5$ and $\sigma_1 = \sigma_2 = 0.2$, but vary ρ, which starts with 0, and then random walks within $[-1, 1]$ with step size $\Delta = 0.05, 0.1$ and 0.2. We call these streams $C(\Delta)$. As observed in Section 6.1, a change in ρ is more difficult to detect than a change in the mean and standard deviation, and thus we have intentionally chosen larger steps sizes for the $C(\Delta)$ streams.

Test 1: Varying Data Sources. In our first experiment, we evaluate the efficacy of the change detection method on the different data stream sources, with the results shown in Table 2. As expected, the performance of our scheme increases as the amount of change gets larger. With $\Delta = 0.05$ for the M and D streams, and $\Delta = 0.2$ for the C streams, we can almost detect all changes. The number of false positives is also quite acceptable, considering there are 5 million points in each of the streams. Note the tradeoff between false positives and negatives; as one decreases, the other increases.

Test 2: Varying the ASL. Next, we investigate how the ASL affects the performance of the change detection scheme. The experiments are performed using the C streams, and we vary α from 5% to 0.2%. The results are summarized in Table 3. Since α determines the sensitivity of the change detection scheme, lower values of α make it more difficult to reject the null hypothesis, leading to low

Stream	Detected	Late	False	Missed
$M(0.01)$	30	17	4	52
$M(0.02)$	70	20	4	9
$M(0.05)$	97	1	4	1
$D(0.01)$	36	20	1	43
$D(0.02)$	95	0	9	4
$D(0.05)$	92	4	7	3
$C(0.1)$	43	18	3	38
$C(0.15)$	83	10	4	6
$C(0.2)$	97	1	4	1

Table 2: Change detection results on different 2D normal data streams.

sensitivity, which in turns means less changes are detected but also less false positives (increasing the power of the test). In practice, one should choose an appropriate value of α depending on whether false positives or false negatives are more tolerable.

Stream	α	Detected	Late	False	Missed
$C(0.1)$	5%	63	15	11	21
$C(0.1)$	1%	43	18	3	38
$C(0.1)$	0.2%	36	13	0	51
$C(0.15)$	5%	88	8	21	3
$C(0.15)$	1%	83	10	4	6
$C(0.15)$	0.2%	76	12	1	11
$C(0.2)$	5%	96	1	26	2
$C(0.2)$	1%	97	1	4	1
$C(0.2)$	0.2%	98	1	3	0

Table 3: Change detection results on the C streams with different ASLs.

Test 3: Varying the window size. We now vary the window size n to see how this affects the performance of our change detection scheme. These experiments were still performed using the $C(\Delta)$ streams. Note that changing the window size does not affect the incremental maintenance cost of our structure, though the space requirement and initial construction cost depends linearly on n. A larger window size gives us a better approximation to the true underlying data distribution, and so we expect our scheme to work better with larger window sizes. This is exactly what we observe from Table 4.

Test 4: Varying the number of bootstrap samples. Finally, we vary k, the number of bootstrap samples, and see how it affects the performance. Once again, we performed our experiments with the $C(\Delta)$ streams, and results are listed in Table 5. Note that k must be at least $1/\alpha$ to be meaningful. What is interesting (and is indicated by Efron and Tibshirani [14]), is that the number of samples does not affect the quality of the results significantly. But if we use too few samples, the robustness of the approach may be impaired, and we feel that $k = 500$ is a reasonable number of samples, as also suggested by some other experimental studies.

Stream	n	Detected	Late	False	Missed
$C(0.1)$	5000	30	25	5	44
	10000	43	18	3	38
	20000	62	7	0	30
$C(0.15)$	5000	68	14	17	17
	10000	83	10	4	6
	20000	91	1	1	7
$C(0.2)$	5000	93	5	15	1
	10000	97	1	4	1
	20000	99	0	0	0

Table 4: Change detection results on the C streams with different window sizes.

Stream	k	Detected	Late	False	Missed
$C(0.1)$	100	51	15	2	33
	500	43	18	3	38
	2000	47	15	2	37
$C(0.15)$	100	85	8	9	6
	500	83	10	4	6
	2000	85	7	8	7
$C(0.2)$	100	97	1	10	1
	500	97	1	4	1
	2000	99	0	2	0

Table 5: Change detection results on the C streams with different number of bootstrap samples.

Test 5: Poisson distributions. Often, a more realistic model for multidimensional data (especially if it arises from natural spatial settings) is a Poisson distribution. We tested our scheme using 2D (discrete) Poisson distributions, with data streams generated according to $(X,Y) \sim Poisson(500(1-\rho), 500(1-\rho), 500\rho)$, where ρ starts at 0.5 and then performs a random walks between 0 and 1 with a step size $\Delta = 0.05, 0.1, 0.2$, in the same manner as before. The goal was to keep the marginal distribution of X and Y the same (with $\lambda = 500$), and change the correlation ρ. We summarize our results in Table 6; once again, larger ρ, corresponding to greater difference, leads to better performance.

Δ	Detected	Late	False	Missed
0.05	60	10	1	29
0.1	67	17	1	15
0.2	98	1	5	0

Table 6: Change detection results on 2D Poisson data streams.

Test 6: Higher dimensions. To test the scalability and performance of our scheme in high dimensions, we take the $C(0.2)$ stream and extend it to d-dimensional streams by adding dimensions in which the data distributions (also Gaussian) do not change. The idea is to see if our method is able to detect the changes in one of the subspaces by working in the full space directly, rather than mapping the points into the 2^d subspaces and do the detection in each subspace individually. The experiment results for different dimensions are listed in Table 7. We can see that as we add more stationary dimensions, it becomes more difficult to detect the real changes, but even with 8 extra dimensions ($d = 10$), our scheme still delivers a reasonable performance.

d	Detected	Late	False	Missed
4	89	1	7	9
6	84	10	8	5
8	83	5	7	11
10	65	12	6	22

Table 7: Change detection results on d-dimensional streams.

Test 7: Efficiency. The cost of our scheme consists of two parts: initial construction of the kdq-tree and incremental updates. Since we reconstruct the kdq-tree only after a change has been reported, and the frequency of changes depend on the data characteristics and the application, it makes sense to measure the two kinds of cost separately. As indicated by Proposition 5.1, incremental updates can be performed very efficiently, in time $O(d\log(\frac{1}{\delta}))$ per time step; the initial construction takes $O(kdn\log(\frac{1}{\delta}))$ time, as we need to repeat for the k bootstrap samples, but it still grows linearly in both d and n. To see how fast our scheme runs in practice, we measure the two kinds of cost on the streams used in Test 6 with varying window sizes, and report the average in Table 8. The other parameters assume their default values in Table 1. These running times were obtained from our unoptimized code on a PC with a 3.0GHz Pentium 4 processor and 1GB memory.

Note that the number listed in the table generally agree with the theoretical bounds, except that the construction cost seems to be growing sub-linearly as dimensions. This is because the bound given in Proposition 5.1 is the worst-case bound, which only happens under contrived examples. The actual cost should be much smaller in most situations.

d	n	Construction (sec)	Update (msec)
4	10000	4.52	0.014
6	10000	5.33	0.022
8	10000	5.46	0.029
10	10000	6.08	0.035
10	20000	13.68	0.036
10	30000	22.09	0.035
10	40000	30.83	0.034

Table 8: Running times with different n's and d's.

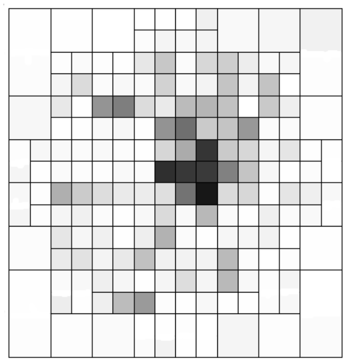

Figure 5: Visualization of the Kulldorff statistic at depth 8 of the *kdq*-tree. The hole is located at $(0.6, 0.6)$ and has radius 0.2.

6.3 Identifying Regions of Greatest Discrepancy

As we described in Section 5, we can augment the nodes in the quad tree with the Kulldorff's spatial scan statistic to further identify the regions where data have changed the most, once a global change has been reported. To illustrate this, we generated a "hole" dataset consisting of 20,000 points. The first 10,000 points are generated from a normal distribution with mean (0.5, 0.5), standard deviation (0.2, 0.2) and correlation 0. The second 10,000 points are generated in the same way, except that for points that fall into a circle centering at (0.6, 0.6) with a radius of 0.2, we reject them with probability 1/2, effectively creating a stochastic "hole". These streams are fed into our scheme with the default parameters. Once we have built the initialized the structure on the two windows W_n and W_{2n}, a change is immediately reported. We then visualize the Kulldorff's statistic associated with the nodes at a certain level of the *kdq*-tree to identify the most different region. In Figure 5, we show a visualization of the *kdq*-tree at depth 8 (with larger values being represented with darker shades of gray), from which the hole can be easily observed.

6.4 Comparison with Prior Work in 1D

Our general change detection scheme works in any dimension, so it is also interesting to compare our scheme with previously known schemes in 1D. In particular, we compare our scheme with the well known Wilcoxon and Kolmogorov-Smirnov (KS) statistics [32], as well as the ϕ and Ξ statistics recently proposed by Kifer *et al.* [21]. These statistics work only in 1D since they crucially rely on the data being ordered. Being nonparametric, they all solely depend on the ranks of the data points. The incremental maintenance costs of these statistics differ greatly. The Wilcoxon and KS statistics can be maintained in $O(\log n)$ time per time step, which is comparable to the update cost of our scheme. The cost to maintain the ϕ and Ξ statistics is $O(n)$ per time step, which is prohibitively high if the stream is coming in at a high rate.

The experimental setup is similar to that in [21]. Each data stream consists of 2 million points, and there is a change every 20,000 points. We created three streams: The first stream is generated from uniform distribution $[0.5 - a, 0.5 + a]$ where a starts with 0.25 and performs a random walk with step size $\Delta = 0.01$. We call this stream U. The second stream is generated from normal distribution with fixed standard deviation $\sigma = 0.2$, while the mean μ starts at 0.5 and performs a random walk between 0.2 and 0.8 with a step size of $\Delta = 0.06$. We call this stream N_μ. The third stream is also generated from normal distribution, where the mean is fixed at $\mu = 0.5$, while σ starts at 0.2 and performs a random walk between 0 and 0.4 with step size $\Delta = 0.06$. This stream is named N_σ. The window size is set to be $n = 1000$. The critical region for the Wilcoxon, KS, ϕ and Ξ statistics are computed corresponding to size(20k, 0.05) according to [21]. This leads to a low false positive rate, so in order to be comparable, we also lower the ASL to $\alpha = 0.02\%$ and use $k = 1000$ bootstrap samples[4].

We list the result in Table 9. Our results generally agree with those obtained in [21]. Specifically, the Wilcoxon statistic performs extremely badly except for changes that affect the median. The KS statistic also primarily looks at changes near the median and does poorly on the N and N_σ streams, but certainly better than Wilcoxon. The ϕ and Ξ statistics perform reasonably well across different kinds of changes, but one should keep in mind its $O(n)$ high maintenance cost, which is exponentially higher than that of Wilcoxon, KL and our scheme. We can see that our KL based approach does not show any obvious weakness, demonstrating its generality and robustness.

Stream	Scheme	Detected	Late	False	Missed
U	Wilcoxon	0	1	1	98
	KS	10	15	3	74
	ϕ	90	6	8	3
	Ξ	81	10	2	8
	KL	72	12	7	15
N_μ	Wilcoxon	90	8	7	1
	KS	89	9	8	1
	ϕ	74	18	4	7
	Ξ	86	13	9	0
	KL	70	23	9	6
N_σ	Wilcoxon	0	3	0	96
	KS	40	19	6	40
	ϕ	58	22	4	19
	Ξ	57	25	4	17
	KL	61	18	9	20

Table 9: Change detection results of different schemes in 1D.

7 Related Work

The Kullback-Leibler distance. The Kullback-Leibler distance is one of the most fundamental measures in information theory and derives its importance from its role in estimating the probability of rare events [7]. It also has a very natural interpretation as a distance function between distributions. As we described earlier, it is a special case of the *Ali-Silvey* distance [9], as well as a special case of the *Bregman* distance [6], in both cases resulting as the unique measure satisfying

[4]There is a subtle issue at play here. The measures proposed by Kifer*et al.* draw on the special structure of the Wilcoxon and KS tests, and thus have sampling distributions that are known and can be approximated *independently* of the input data, which allows them a greater degree of control over the error, at the cost of being limited to these special distributions. Our sampling distributions are defined by the bootstrap samples and are more general, but do have a slightly weaker inherent false positive rate, which we improve by using a stronger ASL.

certain axioms over the unit simplex. The book by Čenkov[2] is a fascinating study of the geometry of the probability simplex; intuitively, the KL-distance appears as a squared geodesic on the manifold defined by probability distributions.

The Kullback-Leibler distance, because of its relation to the log likelihood ratio and Neyman-Pearson classifiers, has been used extensively in classification and model selection problems in machine learning [12, 13, 17, 22, 29, 30, 34].

Most recently, Johnson and Gruner [19] have suggested the idea of using the KL-distance to decode neural signals presented as time series. They also develop the idea of using bootstrap methods to evaluate the significance of their results.

Change detection schemes. A variety of change detection schemes have been studied in the past. Schemes have been proposed for static datasets with specific structure [4, 5], time series data [3, 20], and for detecting "burstiness" in data [23, 35]. There has been a significant amount of work on *anomaly detection* in the networking community; due to space constraints we will defer all discussion of the work in this area.

The definition of change has typically involved fitting a model to the data and determining when the test data deviates from the built model [15, 18]. Other work has used statistical ideas of outliers [35], or has built stochastic likelihood models [23].

In terms of the general formulation that we discussed in Section 1, two approaches are the most directly connected to this paper. The paper by Ganti, Gehrke, Ramakrishnan and Loh [15] uses a family of decision tree models to model data, and defines change in terms of the distance between model parameters that encode both topological and quantitative characteristics of the decision trees. They use bootstrapping to determine the statistical significance of their results.

The paper by Kifer, Ben-David and Gehrke [21] lays out a comprehensive nonparametric framework for change detection in streams. They exploit order statistics of the data, and define generalizations of the Wilcoxon and Kolmogorov-Smirnoff test in order to define distance between two distributions. Their method is effective on one-dimensional data streams; however, as they point out, their approach does not trivially extend to higher dimensions. Aggarwal [1] considers the change detection problem in higher dimensions based on kernel methods; however, his focus is on detecting the "trends" of the data movement, which is different from our goal of detecting statistically significant changes in the underlying distribution. His approach also has a much higher computational cost.

Another approach that focuses on change detection in spatial data was developed by Kulldorff [25]. His work focuses on detecting unusually high disease rates relative to a population. Neill and Moore [27] demonstrated how the Kulldorff distance could be used effectively to find regions of high pattern ("disease") density in a dataset. Their approach is based on a hierarchical data structure that generalizes quad trees, and allows them to determine the maximum density region over all, not just among those defined by the quad tree as in our paper. However, their approach is static, and appears hard to generalize to data streams or dynamic data in general. As we indicated in Section 5, the Kulldorff distance is a special case of the KL-distance.

Computing bootstraps. The bootstrap method was developed by Efron, and the book by Efron and Tibshirani [14] is a definitive reference on bootstrap techniques. KL-distance estimates have bias, and there has been some recent work [33] (for parametric families of distributions) on improving the bootstrap estimates of the KL-distance and removing bias.

8 Conclusions and Further Questions

In summary, this paper presents a general scheme for nonparametric change detection in multidimensional data streams, based on an information-theoretic approach to the data. Our formulation is intrinsically multidimensional, and can even be used to incorporate categorical attributes in data. Experiments indicate that this approach is comparable to more constrained (but powerful) approaches in one dimension, and works efficiently and accurately in higher dimensions. However, many intriguing questions remain. Firstly, it would be interesting to investigate more complex bootstrap methods designed specifically for the KL-distance in order to improve its power and significance, as well as eliminating its inherent bias.

Density estimation kernels could be used to replace the binning scheme we use to compute the KL-distance. We have developed schemes on static data that make use of kernel methods to *smooth* out the bins; however, these are not easily modified for the dynamic/streaming setting. Kernel methods are typically used when an ℓ_2 approximation to an underlying density is acceptable. However, in the information-theoretic setting we need kernels that can approximate the KL-distance well.

Another direction would be to maintain approximate histograms. For ℓ_p metrics, histogram construction is well understood. However, there are no such methods for the KL-distance. It is possible that approximate histograms could be used to improve the performance of our algorithm in higher dimensions.

References

[1] C. C. Aggarwal. A framework for diagnosing changes in evolving data streams. In *SIGMOD*, 2003.

[2] N. N. Čencov. *Statistical Decision Rules and Optimal Inference*, volume 14 of *Translations in Mathematics*. American Mathematical Society, 1982.

[3] S. Chakrabarti, S. Sarawagi, and B. Dom. Mining surprising patterns using temporal description length. In *VLDB*, pages 606–617, 1998.

[4] S. S. Chawathe, S. Abiteboul, and J. Widom. Representing and querying changes in semistructured data. In *ICDE*, pages 4–13, 1998.

[5] S. S. Chawathe and H. Garcia-Molina. Meaningful change detection in structured data. In *SIGMOD*, pages 26–37, 1997.

[6] M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, 48(1/2/3), 2002.

[7] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley and Sons, Inc., 1991.

[8] D. R. Cox and D. Oakes. *Analysis of Survival Data*. Chapman Hall, 1984.

[9] I. Csiszár. Why least squares and maximum entropy? an axiomatic approach to inference for linear inverse problems. *Ann. Statist.*, 19(4):2032–2066, 1991.

[10] I. Csiszár and J. Körner. *Information Theory: Coding Theorems for Discrete Memoryless Systems*. Academic Press, 1981.

[11] T. Dasu and T. Johnson. *Exploratory Data Mining and Data Cleaning*. John Wiley, New York, 2003.

[12] I. S. Dhillon, S. Mallela, and R. Kumar. Enhanced word clustering for hierarchical text classification. In *KDD*, pages 191–200, 2002.

[13] I. S. Dhillon, S. Mallela, and R. Kumar. A divisive information-theoretic feature clustering algorithm for text classification. *J. Machine Learning Research*, 3:1265–1287, 2003.

[14] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman and Hall, 1993.

[15] V. Ganti, J. Gehrke, R. Ramakrishnan, and W.-Y. Loh. A framework for measuring differences in data characteristics. *J. Computer and System Sciences*, 2002.

[16] M. Gutman. Asymptotically optimal classification for multiple tests with empirically observed statistics. *IEEE Trans. Inf. Theory*, 35:401–408, 1989.

[17] T. Hofmann. Probabilistic latent semantic indexing. In *Proc. SIGIR*, pages 50–57. ACM Press, 1999.

[18] G. Hulten, L. Spencer, and P. Domingos. Mining time-changing data streams. In *KDD*, pages 97–106, 2001.

[19] D. Johnson and C. Gruner. Information-theoretic analysis of neural coding. *J. Computational Neuroscience*, 10:47–69, 2001.

[20] E. Keogh, S. Lonardi, and B. Y. Chiu. Finding surprising patterns in a time series database in linear time and space. In *KDD*, pages 550–556, 2002.

[21] D. Kifer, S. Ben-David, and J. Gehrke. Detecting change in data streams. In *VLDB*, pages 180–191, 2004.

[22] J. Kivinen and M. K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132:1–63, 1997.

[23] J. Kleinberg. Bursty and hierarchical structure in streams. *Data Mining and Knowledge Discovery*, 7(4):373–397, 2003.

[24] R. E. Krichevsky and V. K. Trofimov. The performance of universal encoding. *IEEE Trans. Inf. Theory*, 27:199–207, 1981.

[25] M. Kulldorff. A spatial scan statistic. *Communications in Statistics: Theory and Methods*, 26(6):1481–1496, 1997.

[26] P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman Hall, 1989.

[27] D. B. Neill and A. W. Moore. Rapid detection of significant spatial clusters. In *KDD*, 2004.

[28] A. B. Owen. *Empirical Likelihood*. CRC Press, 2001.

[29] F. Pereira, N. Tishby, and L. Lee. Distributional clustering of English words. In *31st Annual Meeting of the ACL*, pages 183–190, 1993.

[30] S. D. Pietra, V. D. Pietra, and J. Lafferty. Inducing features of random fields. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(4), 1997.

[31] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C.* Cambridge University Press, 1992.

[32] D. J. Sheskin. *Handbook of Parametric and nonparametric statistical procedures.* Chapman and Hall/CRC, 3rd edition, 2004.

[33] R. Shibata. Bootstrap estimate of Kullback-Liebler information for model selection. *Statistica Sinica*, 7:375–394, 1997.

[34] N. Tishby, F. Pereira, and W. Bialek. The information bottleneck method. In *Proc. 37th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

[35] Y. Zhu and D. Shasha. Efficient elastic burst detection in data streams. In *KDD*, pages 336–345, 2003.

A Information Theory and Classifiers

In general, let $X_1, \dots X_n$ be independent identically distributed (i.i.d) with respect to distribution Q. Consider the two hypotheses $H_1 : Q = P_1$ and $H_2 : Q = P_2$. A test procedure is some function $g(x_1, \dots x_n)$ such that $g(x_1, \dots x_n) = i$ implies that H_i is chosen. Since there are only two hypotheses, we can also specify g by specifying the set A over which $g = 1$;the complement of A, A^c, is then the set over which $g = 2$.

The two probabilities of error are α, the probability of choosing H_2 when H_1 is true, and β, the converse. These are the so-called *type-1* and *type-2* errors, and can be written as

$$\alpha = Pr(g(x_1, \dots x_n) = 2 \mid H_1 \text{ true}) = P_1^n(A^c)$$

and

$$\beta = Pr(g(x_1, \dots x_n) = 1 \mid H_2 \text{ true}) = P_2^n(A)$$

The goal of hypothesis testing is to design a test procedure that minimizes these two errors, and the Neyman-Pearson theorem tells us that the likelihood ratio defines the best test procedure:

Theorem A.1 (Neyman-Pearson) *For a given threshold T, if we choose as A the region*

$$A(T) = \left\{ \frac{P_1(X_1, \dots X_n)}{P_2(X_1, \dots X_n)} \geq T \right\}$$

with associated error probabilities α^*, β^*, then for any other region B with errors α, β, if $\alpha \leq \alpha^*$, then $\beta \geq \beta^*$.

The relation between the likelihood ratio and the relative entropy can then be expressed via Stein's lemma[7, Sec 12.8].

Lemma A.1 (Stein's Lemma) *For distributions P_1, P_2 as before, with $D(P_1 \| P_2) < \infty$. let A be an acceptance region for hypothesis H_1. Let the associated probabilities of error be α, β, and for $0 < \epsilon < \frac{1}{2}$, let*

$$\beta^\epsilon = \min_{\alpha < \epsilon} \beta$$

Then

$$\lim_{\epsilon \to 0} \lim_{n \to \infty} \frac{1}{n} \log \beta^{\epsilon} = -D(P_1 \| P_2)$$

To understand Stein's Lemma, it is helpful to consider an extreme case. Suppose we have two distributions P_1 and P_2 such that $D(P_1 \| P_2) = 0$ (for discrete distributions, this is equivalent to saying that $P_1 = P_2$). Suppose now that we wish to minimize β, given that $\alpha = 0$. Stein's Lemma tells us that the best bound we can obtain for β is $2^{-nD(P_1 \| P_2)} = 0.5$. How would we achieve this? Since $\alpha = 0$, we can never conclude H_2 if H_1 is true, and so one approach would be to conclude H_1 always. In this case, since the distributions are identical, any input is equally likely to have come from P_1 or P_2, and so we can do no better than a coin toss in determining correctly whether H_2 holds, yielding $\beta = 0.5$.