to appear in:
Proceedings of the 14th SIAM International Conference on Data Mining (SDM), Philadelphia, PA, 2014

Generalized Outlier Detection with Flexible Kernel Density Estimates

Erich Schubert Arthur Zimek Hans-Peter Kriegel

Institut für Informatik, Ludwig-Maximilians-Universität München, Germany
{schube,zimek,kriegel}@dbs.ifi.lmu.de

Abstract

We analyse the interplay of density estimation and outlier detection in density-based outlier detection. By clear and principled decoupling of both steps, we formulate a generalization of density-based outlier detection methods based on kernel density estimation. Embedded in a broader framework for outlier detection, the resulting method can be easily adapted to detect novel types of outliers: while common outlier detection methods are designed for detecting objects in sparse areas of the data set, our method can be modified to also detect unusual local concentrations or trends in the data set if desired. It allows for the integration of domain knowledge and specific requirements. We demonstrate the flexible applicability and scalability of the method on large real world data sets.

1 Introduction

Barnett and Lewis defined an outlier in their classic textbook [6] as: "An observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data." The definitions and measures of "inconsistency", however, vary widely. Traditionally, outliers were detected based on assumptions concerning statistical distributions. In the context of normal distributions, the "$3 \cdot \sigma$ rule" is the classic example of an outlier test [15]. Alas, in the domain of data mining, data are usually too complex and too large to allow for proper statistical modeling, let alone the evaluation of such models. Large databases require fast and efficient, yet flexible tests for outliers without the need to fully model the data first. DB-Outlier [15] is a good example of such a pragmatic approach to detecting outliers: instead of trying to model the data, a simple radius query is sent to the database for each object, and if there are too few neighbor objects, the object is considered to be an outlier. At first, this approach may appear to be overly simplistic but, when put into the context of database analysis, it has the advantage of being highly efficient as the database index structures can accelerate the required neighborhood queries. Many new methods have been proposed since, often only with slight differences, and many are discussed in a recent textbook [3] and in related surveys [9, 25]. A recently very active domain is subspace outlier detection, tackling the challenges of outlier detection in complex and high-dimensional data [34, 10]. Here, we address local density-based outlier detection in low-dimensional data. Meta-methods such as HiCS [14] could allow to use our method in high-dimensional spaces. It can be seen as a statistically refined and cleaned up variant of density-based outlier detection [8, 21, 18, 13], that tries to go back to the statistical roots of the methods, while still keeping true to the domain of database-backed data analysis and advancing the state of the art both by having a clean connection to the statistical background, but also by making the method more flexible than the earlier algorithms.

In the remainder, we survey outlier detection methods and kernel density estimation in Section 2. In Section 3, we examine some of them, how they connect to density estimation, and in which way they use an overly naïve notion of density. Based on this analysis, we elaborate on the construction of a kernel density estimation (KDE)-based outlier detection method that can be fine-tuned to the domain specific requirements. In Section 4, we demonstrate the flexibility, usefulness, and scalability on large real-world data sets and novel problems. Section 5 concludes the paper.

2 Related Work

The distance-based notion of outliers [15] considers an object x being an outlier if more than a given fraction of all objects in the database have a distance larger than some threshold from the object x. Variants of the distance-based notion of outliers are models based on the distances to the k nearest neighbors [22, 4]. Both the idea of counting neighbors within a threshold radius as well as using the k nearest neighbors are simple density estimates. Refined "local" density-based approaches consider ratios between the local density around an object and the local density around its neighboring objects. The local outlier factor (LOF) [8] compares the density estimate for each object o of a database D with the average density estimates for the k nearest neighbors of o. A LOF value of approximately 1 indicates that the corresponding object is located within a region of homogeneous density (i.e. a cluster). If the density in the local neighborhood of o is lower than the density around the k nearest neighbors of o, o gets assigned a higher LOF value. The higher the LOF value of an object o is, the more distinctly is o considered an outlier.

Several extensions and refinements of the basic LOF model have been proposed, e.g. a connectivity-based outlier factor (COF) [29], or using the reverse nearest neighbors additionally to the nearest neighbors and considering a symmetric relationship between both values as a measure of outlierness [13]. Another variant is named Local Outlier Integral (LOCI) [21], based on the concept of a multi-granularity deviation factor. An adaptation of the distance-based notion of outliers to the idea of local outliers results in a local distance-based outlier detection (LDOF) approach [31]. An attempt to stabilize the results of LOF and make it more robust with respect to the choice of the minPts parameter is LoOP [16]. It is based on a standard deviation-based distance estimation and uses the Gaussian error function to normalize the outlier scores to $[0; 1]$ to improve usability. LDF [18] is a naïve merge of LOF concepts and kernel density estimation.

Kernel density estimation (KDE) is a field that has received much attention in statistics since the 1950s. Summaries of the state of the art can be found in various textbooks and publications [30, 26]. Yet, these techniques are underused in many data mining disciplines, often for the reason that these methods do not pay much attention to computational complexity in databases. KDE is commonly accepted as quadratic complexity, and then is often approximated by using grid based binning, which can be computed in $\mathcal{O}(G \log G)$ using fast Fourier transform (FFT) based convolution [28]. For local outlier detection, a grid based density estimation does not work very well, as it loses local differences in density below the grid resolution. However, kernel density estimation in general obviously is a prime candidate to improve the quality of density-based outlier detection methods, which will be the focus of this article.

To summarize, many approaches to outlier detection provide variants of outlier scores that are essentially based on different methods for density estimation. Since these scores are based on quite different assumptions, intuitions, and models they are differently well suited for different data sets and application areas.

3 KDE for Outlier Detection

The method we propose is not the first to use kernel density estimation (KDE). However, it is the first that pays attention to the statistics of density estimation, and then in a second phase uses these density estimates and outlier detection. For example, Local Density Factor (LDF) [18] is a LOF variant that explicitly uses kernel density estimation. However, instead of keeping the statistic knowledge of density estimation, they modify their estimates (in a statistically not well founded way) to resemble the naïve density estimation of LOF. As a consequence, instead of combining the strength of both and separating the different components, they carry over a conceptual weakness of LOF into their modified KDE. We will detail fundamental concepts of LOF and the conceptional deficits of LDF here before we discuss our approach which, as we will demonstrate, is a more general conceptional and practical improvement over LDF.

3.1 State-of-the-Art First, we will analyze existing methods with respect to the kernel they use for density estimation, to show why it may be desirable to use a proper kernel density instead. We will start with one of the best known outlier detection methods in data mining: the Local Outlier Factor (LOF) [8], and its simpler variant Simplified-LOF [25]. Both can be seen as two phase approaches, where the first phase is the density estimation, the second phase compares the density-estimate of each object to the density-estimates of its neighbors.

LOF uses the asymmetric *reachability distance*:

$$\text{reach-dist}_k(o \leftarrow p) = \max\{k\text{-dist}(p), d(o,p)\} \tag{3.1}$$

where k-dist(p) is the distance from p to its kth nearest neighbor, which is the core distance in OPTICS [5] clustering. From these distances, the *local reachability density* is then estimated using the formula:

$$\text{lrd}(o) := 1 \Big/ \frac{\sum_{p \in k\text{NN}(o)} \text{reach-dist}_k(o \leftarrow p)}{|k\text{NN}(o)|} \tag{3.2}$$

where kNN(o) are the k nearest neighbors of o. Simplified-LOF substitutes the regular distance $d(o,p)$ for the reachability distance. Mathematically, Equation 3.2 however is the *harmonic mean* of the neighbor density estimates:

$$\text{lrd}(o) \equiv \underset{p \in k\text{NN}(o)}{\text{harmonic-mean}} \frac{1}{\text{reach-dist}_k(p \leftarrow o)}. \tag{3.3}$$

This formula is closely related to the standard kernel density estimation formula, but using a different mean:

$$\text{KDE}(o) := \frac{1}{n} \sum_{p \in \text{DB}} K_h(d(o,p)) \equiv \underset{p \in \text{DB}}{\text{mean}} K_h(d(o,p))$$

where $K_h(u)$ is the kernel function with bandwidth h. With $K_{\text{lrd}}(u) = 1/|u|$ the KDE becomes Simplified-LOF with a different mean and with the full database DB instead of using only the neighborhood kNN. If we assume that the density contribution for objects outside the neighborhood kNN(o) is negligible, then this reduces to a linear scaling factor k/n. For LOF itself, the similarity can be formalized as:

$$\begin{aligned}\text{lrd}(o) &= \text{harmonic-mean}_{k\text{NN}(o)} K_{\text{lrd}}(\text{reach-dist}_k(p \leftarrow o)) \\ &\approx \text{harmonic-mean}_{o \in \text{DB}} \frac{n}{k} K_{\text{lrd}}(\text{reach-dist}_k(p \leftarrow o)).\end{aligned}$$

We can now identify and review four differences between the density estimate of LOF and KDE:

(1) the use of the harmonic instead of the arithmetic mean;

(2) density estimation is based on the kNN instead of the complete database;

(3) instead of a statistical kernel function, LOF uses $1/|u|$;

(4) LOF (but not Simplified-LOF) uses reachability distance instead of distance.

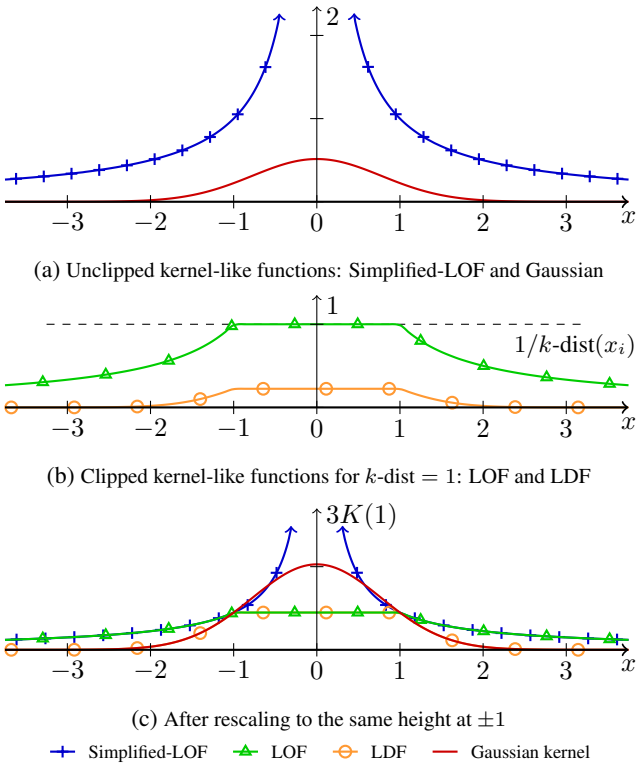
(1) The choice of the harmonic mean (which is most appropriate for values in a frequency domain) is due to the original formulation of LOF, but was not a conscious choice.The inverse quadratic mean used by LoOP [16] is an heuristic hot-fix trying to improve robustness, but is not based on a strong theory. The model of KDE on the other hand is based on additive functions and preserves the unit integral (total density) as desired in density estimation. Therefore, the arithmetic mean is the preferred choice.

(2) Using the kNN only instead of the full database is reasonable in the light of using an indexed database, which reduces overall complexity from $\mathcal{O}(n^2)$ to $\mathcal{O}(n \log n)$. In fact, nearest-neighbor methods are common in Kernel Density Estimation, albeit used slightly differently; compare a review of nearest-neighbor based density estimation [30] and a recent development using the k nearest neighbors [20]. However, kernel density estimators based on theory will always divide the density by the sample size n, not the neighborhood size k.

The points (3) and (4) are two sides of the same coin: how to estimate the density contribution of one neighbor to the sample point. The approach used by LOF is based on the idea of smoothing the results by using the core-distance introduced in OPTICS [5] clustering. The resulting function is, however, not a proper kernel function in the widely accepted definition of a kernel density function, which has the key properties of having a unit integral $\int K(x)\mathrm{d}x = 1$. Not only has the kernel used by LOF an infinite integral, but even when truncated at a finite maximum radius, it does not have a constant integral value. Instead, it gives more weight to objects in dense regions (with a small k-dist), and less weight to objects in sparse regions. The simplified kernel $1/|d(o,p)|$ used by Simplified-LOF (and similarly, the kernel used by LoOP) is even worse here: for $d(o,p) = 0$, the density contribution becomes infinite. The only reason this problem does not surface immediately is due to the computation being done with the inverse values, and the harmonic mean is able to handle infinite values (e.g. $H(.5, \infty) = 1$). The additional clipping by k-dist used by LOF then further stabilizes the results.

Local Density Factor (LDF) [18] is motivated from standard kernel density estimation, using the Gaussian kernel. The widely accepted rotational symmetric version of the Gaussian kernel density estimation in d dimensions is

$$\mathrm{KDE}_{\mathrm{Gauss}}(o) := \frac{1}{n} \sum_{p} \frac{1}{(2\pi)^{d/2} h^d} e^{-\frac{1}{2}\frac{d(o,p)^2}{h^2}}. \tag{3.4}$$



(a) Unclipped kernel-like functions: Simplified-LOF and Gaussian

(b) Clipped kernel-like functions for k-dist $= 1$: LOF and LDF

(c) After rescaling to the same height at ± 1

Figure 1: Simplified-LOF, LOF, and LDF kernel-like functions in comparison to the regular Gaussian kernel

However, the authors of LDF chose not to stick to this popular kernel. Instead, they modified it akin to the LOF reachability density, by substituting the local reachability distance for the plain distance in their density estimate:

$$\mathrm{LDE}_d(o) := \frac{1}{|k\mathrm{NN}(o)|} \sum_{p \in k\mathrm{NN}(o)} \frac{1}{(2\pi)^{d/2} h^d} e^{-\frac{1}{2}\frac{\text{reach-dist}(o,p)^2}{h^2}}.$$

However, by this modification, LDE loses the desired properties of a proper kernel density estimation: instead of having an integral area of 1 for every point, a varying amount of weight is clipped from the kernel in a heuristic fashion. While the resulting kernel has a finite integral, the exact integral value varies based on the k-distances of the neighbors.

In Figure 1, we visualize these kernel-like functions to explain our concerns. In Figure 1a, we plot the unclipped functions, i.e. Simplified-LOF and the Gaussian kernel. Figure 1b visualizes the kernel-like functions after clipping them at a maximum height of $K(1)$. For the LOF function, the clipping is expected to be beneficial, as it avoids the extreme values that would occur close to 0. For LDF however–which uses a modified Gaussian kernel–this argument does not hold. Not only is the Gaussian kernel finite, it also already has a flat top that does not discriminate small distances much anyway. Figure 1c tries to compare the four

kernels with each other by adding a constant scaling factor that does not change relative densities. It can be seen that in the range of $0.75 \dots 1.5$ the kernels are all very similar, whereas close to 0, the Simplified-LOF kernel is substantially different, not necessarily for the better.

In conclusion, while it is rather obvious that LOF and its variants perform some kind of density estimation, the exact properties of this density estimation have not yet been properly investigated or discussed. In this article, we explicitly studied the relationship of LOF to kernel density estimation, and found that LOF and its variants use a heuristic approach not well supported by theory. Even the LDF variant, which tries to combine LOF with kernel density estimation, does so inconsistently and ignores the theoretical foundations of KDE without giving a reason.

3.2 Reconsidering KDE for Outlier Detection The solution to this challenge has the beauty of surprising simplicity. Our proposed method does not divert substantially from the existing methods. We also perform density estimation, then compare the densities within local neighborhoods. As such, it is also an instantiation of the general pattern of *local outlier detection* [25]. However, we propose to use classic kernel density estimation directly instead of experimenting with non-standard kernels without giving a good reason for this.

3.2.1 Density Estimation Step The kernel function to use with our method is best to be considered an input parameter to the algorithm. We suggest to use either the Gaussian or Epanechnikov kernels of bandwidth h and dimensionality d:

$$K_{\text{gauss},h}(u) := \frac{1}{(2\pi)^{d/2}h^d} e^{-\frac{1}{2}\frac{u^2}{h^2}}, \tag{3.5}$$

$$K_{\text{epanechnikov},h}(u) := \frac{3}{4h^d}\left(1 - \frac{u^2}{h^2}\right). \tag{3.6}$$

The radially symmetric versions have the benefit that we only have one bandwidth to estimate, instead of having to estimate full bandwidth matrices for each object, which continues to be a difficult problem [26]. The *balloon estimator* and *sample smoothing estimator* [30] are:

$$\text{KDE}_{\text{balloon},h}(o) := \frac{1}{n}\sum_{p} K_{h(o)}\left(o - p\right), \tag{3.7}$$

$$\text{KDE}_{\text{sample},h}(o) := \frac{1}{n}\sum_{p} K_{h(p)}\left(o - p\right). \tag{3.8}$$

In order to estimate the local kernel bandwidth $h(o)$ respectively $h(p)$, a classic approach is to use the nearest-neighbor distances [19], i.e. $h(o) = k\text{-dist}(o)$. Again, note the similarity between LOF and the sample smoothing estimator. For our method, we will use the balloon estimators, because research in kernel density estimation shows both theoretical and experimental benefits in multivariate KDE [30]. Nevertheless, there are concerns about the bias of this method on the long tail [26]. Sheather and Jones [27] discuss a data-driven method to estimate kernel bandwidths known as the "plug-in bandwidth estimator". For robustness, we use $h(o) = \min\{\text{mean}_{p\in k\text{NN}} d(p,o), \varepsilon\}$ to avoid division by 0 and to be more resistant to outliers in the kNN. But in general, any advanced kernel density estimation method can be used for this step.

Essentially, for density estimation, we recommend to stick to the established and proved methods of density estimation, but additionally to consider runtime. For example, if you have database indexes available to accelerate range queries or k nearest neighbor queries, an approximate density estimation exploiting these indexes is desirable. If the kernel function $K(o - p)$ bears next to no weight beyond the k-nearest neighbor, we do not need to use these for density estimation. We can also drop constant scaling factors, yielding:

$$n \cdot \text{KDE}_{\text{kNN}}(o) := \sum_{p\in k\text{NN}(o)} K_{h(o)}\left(o - p\right). \tag{3.9}$$

Since the parameter k can be hard to choose, we propose to extend the method to cover a range of $k = k_{\text{min}} \dots k_{\text{max}}$ to produce a series of density estimates (one for each k). This approach is similar to LOCI [21], yet it is computationally more efficient and elegant, as the values of k are well-defined steps, while the LOCI model needs to test arbitrary ε-radii. By this extension, the method becomes an ensemble method [2, 33, 32], typically yielding more stable and reliable results.

3.2.2 Density Comparison Step The density comparison function used in LOF and its variants can be written as:

$$\text{LOF}(o) := \underset{p\in k\text{NN}(o)}{\text{mean}} \frac{\text{lrd}(p)}{\text{lrd}(o)} \equiv \frac{\text{mean}_{p\in k\text{NN}(o)}\text{lrd}(p)}{\text{lrd}(o)}.$$

For an object that has an average density, this factor will be close to 1, while for objects with neighborhoods of much higher density than that of the object, this value will be larger. However, there is little control over how large the values become, or when a value is significantly large.

For our approach, we use a slightly different comparison method. We assume that not only the local densities vary, but also the variability itself is sensitive to locality. Therefore, to standardize the deviation from normal density, we apply the well-known z-score transformation: Let μ_X be the mean of the set X and σ_X the standard deviation. The z-score of x then is $z(x, X) := (x - \mu_X)/\sigma_X$ (if $\sigma_X = 0$, then use $z(x, X) := 0$). Alternatively, one could use more robust statistics such as the median absolute deviation from median (MAD) [12]. However, for small sample sizes, the mean often works better than the median. Only for

Algorithm 1: Basic KDE Outlier Algorithm

```
s := array for output scores
S := two dimensional array, o × k_max
// Perform kernel density estimation (KDE):
foreach o in DB do
    N_max = compute k_max-nearest neighbors of o
    foreach k in k_min ... k_max do
        h = compute kernel bandwidth from N_max[1; k]
        foreach n in N_max[1; k] do
            u = distance (o, n)
            S[o][k] = S[o][k] + K(u, h)
        end
    end
end
// Compare densities:
foreach o in DB do
    N_max = compute/get k_max-nearest neighbors of o
    foreach k in k_min ... k_max do
        μ := mean of S[N_max][k]
        σ := standard deviation of S[N_max][k]
        s[o] = s[o] + (μ − S[o][k])/σ
    end
    s[o] = s[o]/(k_max − k_min + 1)
end
// Normalize scores:
foreach o in DB do
    s[o] = 1 − cdf(s[o])
    s[o] = φ · (1 − s[o])/(φ + s[o])
end
return s
```

large values, the median and MAD become more robust to outliers. Intuitively, a z-score of $+3$ indicates a deviation of three standard deviations. When using multiple values of k, we use the average z-score:

$$s(o) := \underset{k_{\min}\dots k_{\max}}{\text{mean}}\ z\left(\text{KDE}(o), \{\text{KDE}(p)\}_{p\in k\text{NN}(o)}\right) \tag{3.10}$$

3.2.3 Score Normalization Step If we now assume the resulting scores to be approximately normally distributed, we can use the normal cumulative density function Φ to normalize the scores to the range $[0; 1]$, and then apply the rescaling introduced in [17],

$$\text{norm}(p, \varphi) := \frac{\varphi \cdot (1 - p)}{\varphi + p}, \tag{3.11}$$

to obtain our proposed outlier score

$$\text{KDEOS}(o, \varphi) := \text{norm}(1 - \Phi(s(o)), \varphi). \tag{3.12}$$

where φ is the expected rate of outliers. This parameter can be intuitively interpreted as a significance threshold.

3.2.4 Algorithm and Complexity Algorithm 1 gives the basic computation of the KDEOS scores for a range of $k = k_{\min} \dots k_{\max}$. But note that we advocate to adapt this code to the particular problem at hand by integrating domain knowledge and specific requirements (we will demonstrate this in the experimental section). The overall code complexity is not high, if the evaluation framework can efficiently provide the neighbor sets. Also notice that the majority of this code is what is called "embarrassingly parallel": if the neighbor sets are precomputed, the main loops can be executed by many mappers in parallel. The aggregation of the $S[o]$ and $s[o]$ values then is the canonical reduce step. The interim data produced is of size $\mathcal{O}(nk^2)$, thus only linear in the input data size. Therefore, except for the neighborhood computation step, this algorithm is easy to implement in a distributed computation framework such as MapReduce.

The complexity of this method is comparable to LOF and many other outlier detection algorithms. In practise, the runtime is dominated by the cost of computing the kNN, which without index support requires $\mathcal{O}(n^2)$ distance computations. Index structures such as the R*-tree [7] for nearest neighbor search can reduce this runtime to $\mathcal{O}(n \log n)$.

Excluding the cost of computing the kNN, the main analysis loop then requires $\mathcal{O}(n \cdot k \cdot \Delta k)$ operations ($\Delta k = k_{\max} - k_{\min} + 1$), usually with $k \ll n$. While there is obviously a computational overhead for the more complex kernel functions, it is small compared to the data management and distance computation costs.

4 Experiments

To study the merits of using flexible KDE for outlier detection, we report results in comparison to standard methods on a well-known data set and two case studies demonstrating the customization for data sets with special requirements. The claims we support with this evaluation are (4.1) the suitability and competitiveness of our method on a well understood outlier task in comparison to several established methods, and (4.2&4.3) the flexibility of our model with adaptations to two different real data scenarios, where the competitors are not applicable in a meaningful way.

4.1 Low-Density Outlier Detection The Amsterdam Library of Object Images (ALOI) [11] is a collection of 110250 images of 1000 small objects, i.e. about 110 images of each object, from different angles and with different light conditions. By downsampling some of these classes to become rare objects, we obtained a data set retaining 75000 images, 717 of which are rare objects (known outliers, up to 4 from the same class).[1] For our experiments, we used the 27-dimensional RGB histogram representation. Overall, this data set and task is a classic setting for density-based outlier detection: the rare objects are expected to be in less dense areas than the members of the clustered images. This data

[1] This data set is available for download at
http://elki.dbs.ifi.lmu.de/wiki/DataSets/MultiView

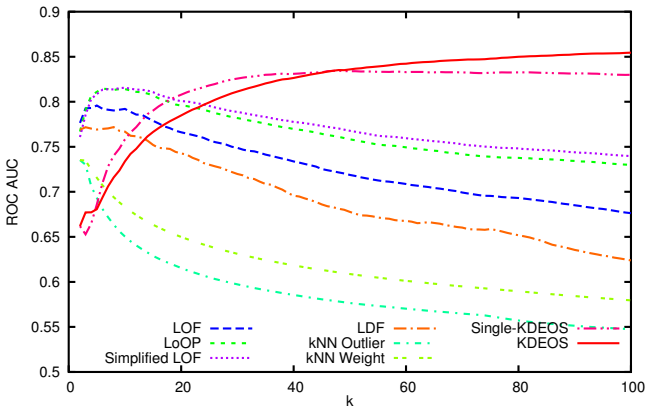


Figure 2: Performance over different values of k.

Table 1: Best performance of different algorithms.

Method	LOF	LoOP	S-LOF	LDF	kNN	kNN-W	Single-KDEOS	KDEOS
Best AUC	.795	.815	.816	.772	.735	.735	.834	**.855**
Best k	5	6	10	8	2	2	$k_{\min} = k_{\max} = 48$	$k_{\max} = 100$

set is non-trivial: besides the labeled rare objects there are also outliers within the other classes coming from rare light situations, and on the other hand, some objects are very similar, and as such some downsampled objects may indeed have another full class that looks alike.

Figure 2 visualizes the performance of various algorithms on this data set in terms of ROC AUC which is typically used for evaluation of outlier detection [23]. As competitors, we use LOF [8], LoOP [16], Simplified-LOF [25], LDF [18], kNN outlier [22], and kNN weight [4] (implementations in ELKI [1]). For Single-KDEOS we set $k_{\min} = k_{\max}$, while for KDEOS we used $k_{\min} = 1$. We report the numbers for the Gaussian kernel, but results using Epanechnikov kernel were almost identical. The best results and the k for the best result are given in Table 1. kNN outlier works best with very small values of k. In ELKI $k = 2$ is the 1NN distance, as the query point is part of the database. For LOF and most of its variants, there is a sweet spot in the small ks. Interestingly, the Simplified-LOF variants work much better than LOF on this data set, probably due to the structure of micro-clusters in this particular data set. Our proposed method produces much more stable results, in particular when choosing a large enough range of ks. Besides offering the best performance of the evaluated algorithms, the parameter k is also much easier to choose – it just needs to be large enough for the kernel density estimation to yield meaningful results. This also is the main limitation: for low values of k, the density estimates are not yet meaningful, and KDEOS thus does not (yet) yield good results, whereas a method such as kNN outlier detection, using a very simple density estimate, can often yield okay results with the 1-nearest neighbor distance.

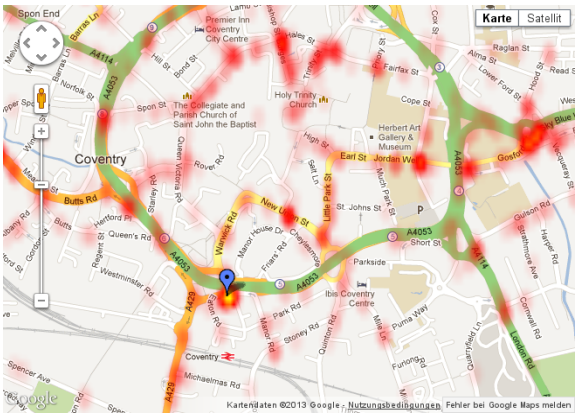
4.2 Case Study: Road Accidents Blackspots To demonstrate the flexibility of our approach, we use a fairly large data set, the road accidents data set from the UK government, spanning the years 2005 to 2011 containing data on 1.2 million road accidents in the UK.[2] On these data, the results of traditional outlier detection methods such as LOF are of little interest to the user: these outliers consist mostly of accidents on low use side roads that only see a single accident every dozen years. Instead, areas of interest are regions with a high concentration of car accidents, and again within these hotspots—many crossroads and roundabouts will show up as hotspots mainly due to the high volume of traffic—only those are interesting that particularly stick out with respect to the usual hotspots in their larger neighborhood.

Our method KDEOS can be easily adapted to the particular needs of this data set. First of all, we modify the density estimation to use a fixed size kernel instead of a dynamic bandwidth. Since cars and roads have the same size across the country, we do not need to adapt the kernel size to local data trends. Instead, we chose a radius of 50 meters along with the Epanechnikov kernel (which drops to 0 beyond the maximal bandwidth). As distance function we choose the great-circle distance and an adapted R*-tree index to accelerate search [24]. In the comparison step, we use a much larger radius of 2 kilometers. Instead of looking for objects of unusually low density, we look for observations of an unusually high accident rate. Observations with a variance of 0—which only happens in remote areas—are not reported as outliers. In order to find the top outliers, we do not need to apply normalization beyond the z-score. In a post-processing step, we extract only the object with the largest density within a radius of 50 meters as representative of the hotspot. Obviously, other accidents at the same location will achieve nearly the same score, and reporting all of these to the user does not yield any benefit.

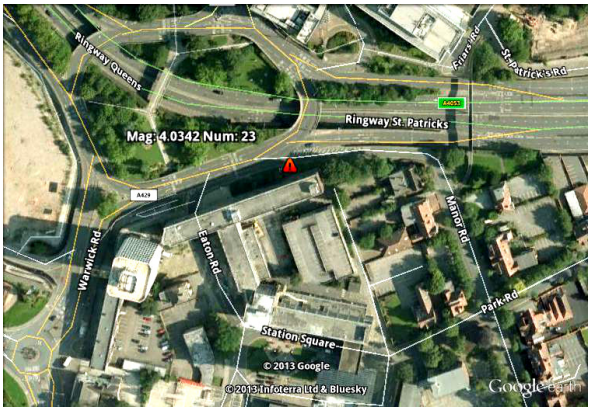
Figure 3 shows an unusual hotspot in Coventry, UK, visualized in Google Maps and Google Earth. In Figure 3a, the traffic accident rate (i.e., our density estimation) is indicated using a heat map. We can see multiple other hotspots in the area, but the detected outlier clearly is an unusually high concentration (bright yellow indicates twice the concentration of bright red). Removing the overlay and zooming in, in Figure 3b, we can see the potential cause for this hotspot: a three way merge with two lanes coming from the roundabout, two lanes coming from the ringway, and a fifth lane coming from the short term parking and car park at Coventry Station. This hotspot is found to have a 4 standard deviations higher accident rate than other accident sites.

A very different hotspot can be seen in Figure 4, near Sunderland in northern UK. Tunstall Hope Road is a known "accident blackspot", "deathtrap", and "one of the most

[2] Publicly available on `http://data.gov.uk/`

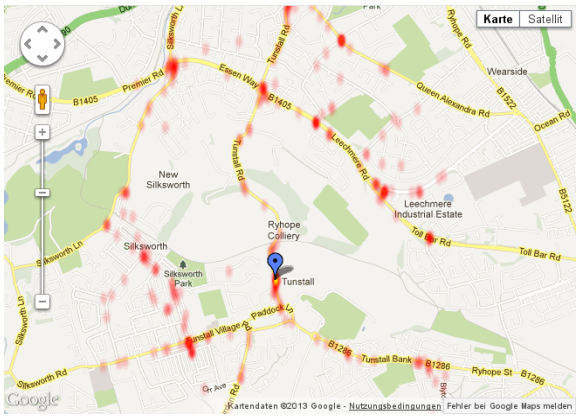


(a) Accident density overlay in Google Maps



(b) 23 accidents at the three-way merge with the entrance to and exit from station square and the car parks there via Manor Road.

Figure 3: Traffic accident hotspot in Coventry, UK.
Background imagery © 2013 Google, Infoterra Ltd & Bluesky



(a) Accident density overlay in Google Maps

(b) Dangerous blind corner with 11 accidents. "Slow" markings were added to the street to warn drivers of the dangers ahead

Figure 4: Traffic accident hotspot in Sunderland, UK.
Background imagery © 2013 Google, Infoterra Ltd & Bluesky

dangerous in Sunderland". Markings on the street now warn drivers to drive slowly. Figure 4a indicates that there is a particularly dangerous spot on this road, with 11 accidents at this particular corner. With the overlay removed in Figure 4b, the place seems to be very usual, but it is easy to imagine that this blind corner, combined with the lack of street lighting and probably slippery foliage can indeed be dangerous.

Yet, only two single spots of the top 50 outliers detected are in Greater London. One of them is seen in Figure 5, a huge roundabout with two multi-lane cut-throughs just north of the M4 motorway and Heathrow airport. This place was a top 10 serious accident site in Greater London, and has since been remodeled, removing one of the cut-throughs in an attempt to reduce car accidents. The reason why so few outliers were detected in London probably lies in the fact that in London there are so many high-accident junctions that none of them sticks out as substantially more serious than the others. Of course, none of the analyzed outliers is a new result. Traffic accident blackspots are usually well known to local police, and can be better analyzed. However, that this fairly general method can be easily adapted to this particular problem shows the flexibility of the approach. At the same time, it shows the following need: instead of looking for an off-the-shelf and parameterless algorithm, a good data mining approach is modular and can this way easily be adapted to the domain knowledge for the desired use case.

4.3 Case Study: Radiation Measurements For the second case study, we use a data set of 6.8 million radiation measurements taken (mostly) in Japan after the Fukushima nuclear disaster.[3] This is a spatio-temporal data set with very different data density due to different sampling rates, automated measurements and a high amount of noise due to different sensors, mobile sensors, and low sensor quality. Again, classic outlier detection methods will be of little use,

[3] Publicly available on `http://safecast.org/`

Figure 5: Traffic accident hotspot north of Heathrow. The east-west cut-through has since been blocked.
Background imagery © 2013 Google, Infoterra Ltd & Bluesky

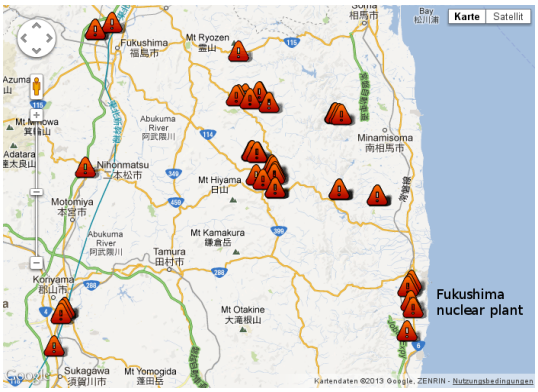


Figure 6: Outliers in radiation measurements, Fukushima prefecture, Japan, in Google Maps.
Background imagery © 2013 Google, ZENRIN

as they do not take spatial and temporal relationship into account. Outliers detected by running LOF on such a data set will not bear useful semantics, as LOF does not treat time, location, and radiation level attributes differently. Therefore, we again need to customize our method for this problem to get meaningful results.

The main customization point here is the definition of neighbors since here the temporal aspect is just as important as the spatial aspect. We define a specialized distance function as follows:

$$d(x \leftarrow y) := \begin{cases} \frac{\text{spatial}(x,y)}{100\text{ meter}} + \frac{\text{temporal}(x,y)}{1\text{ day}} & \text{if } x \text{ before } y \\ \infty & \text{otherwise} \end{cases}$$

This distance function combines both spatial and temporal differences, but it also excludes measurements that come later in time. The reason is that we only want to use a measurement x to estimate the value of a measurement y if it is in the future, in order to detect unexpected changes.

Using this combined (and asymmetric) distance function, we compute the kNN of each object. But instead of estimating the density of observations, we want to estimate the radiation level. For this we use the kernel density as *weight* for averaging the neighboring radiation levels. (An improved method could also take radioactive decay into account.) Mathematically, the method changes only marginally: each object now has a different weight instead of a unit value. This yields two different values for each point: an estimated (or "predicted") radiation level and an actual measurement. However, the raw difference of these two is not very useful yet: in areas where a lot of measurements were taken, we will be having much better predictions, and in areas with higher radiation levels, the natural differences will be much higher. So in order to be able to detect outliers in this data set, we need to put these measurements into a local context. For this we can again use the concepts of local outlier detection as introduced by LOF and discussed throughout this paper. However, the SafeCast data are very noisy: there are measurement errors, varying general radiation levels, and an highly imbalanced number of measurements[4]. Therefore, we need to choose a rather large value of k as reference set, and instead of using mean and variance, we will this time use the median absolute deviation (MAD) between the predicted and the actual values to standardize the deviation. We can then again pinpoint the most "unusually extreme" deviations to the user.

Many of the detected outliers are probably due to measurement errors and badly calibrated sensors. The data provided by SafeCast is in cpm, a unit that is considered to be highly sensor model dependent. Nevertheless, the inspected values all indicate that the outlier detection worked as desired: KDEOS was able to detect outliers of very different magnitudes in different areas of the map.

Some of the most extreme outliers were found along National Route 6, the closest highway to the Fukushima nuclear plants and closed since (in the bottom right of Figure 6). Here, readings of 8000 cpm have been reported in September 2011 and October 2011, while values of 3000 cpm (respectively 700) were predicted. On the online map by SafeCast, peak values of 10000 cpm and averages of 6000 are listed, but these averages do not take temporal aspects into account. On the other hand, there are also a number of outliers reported for Koriyama, consisting of measurements around 200, where 90 are considered normal. A large amount of outliers was detected in the area between Mount Hiyama and Mount Ryozen, an area that was in wind direction when radioactive material was released at Fukushima. Other outliers seem to correlate with highway restaurants, plausibly caused by drivers stopping there and bringing in fresh air and dirt.

[4]Some users contributed up to 50.000 automatic measurements mostly from their back yard, which are extremely redundant.

5 Conclusion

In previous approaches to density-based outlier detection, density estimation and its application to outlier detection were intermixed and combined in a heuristic way only. Here, by theoretical analysis, we provide a clear and principled separation of both components. This results in a combination of existing experience in kernel density estimation with the ideas of local and density-based outlier detection methods, preserving the strengths of both. As a consequence, the new approach, KDEOS, can be seen as a blueprint that can easily be adjusted in various places, to incorporate domain specific knowledge and application-specific requirements, as well as in order to specify the desired type of outliers to be detected. By decoupling the different steps of outlier detection, the new method allows a best-of selection from different domains: established density estimates by KDE in the first step, robust statistics in the second step, and finally a user-oriented score rescaling yield a powerful default combination for outlier detection. Any of these steps can be modified to suit the particular needs of the data set, of the application scenario, and of the user.

References

[1] E. Achtert, H.-P. Kriegel, E. Schubert, and A. Zimek. Interactive data mining with 3D-Parallel-Coordinate-Trees. In *Proc. SIGMOD*, pages 1009–1012, 2013.

[2] C. C. Aggarwal. Outlier ensembles [position paper]. *SIGKDD Explor.*, 14(2):49–58, 2012.

[3] C. C. Aggarwal. *Outlier Analysis*. Springer, 2013.

[4] F. Angiulli and C. Pizzuti. Fast outlier detection in high dimensional spaces. In *Proc. PKDD*, pages 15–26, 2002.

[5] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. OPTICS: Ordering points to identify the clustering structure. In *Proc. SIGMOD*, pages 49–60, 1999.

[6] V. Barnett and T. Lewis. *Outliers in Statistical Data*. John Wiley&Sons, 3rd edition, 1994.

[7] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-Tree: An efficient and robust access method for points and rectangles. In *Proc. SIGMOD*, pages 322–331, 1990.

[8] M. M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander. LOF: Identifying density-based local outliers. In *Proc. SIGMOD*, pages 93–104, 2000.

[9] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM CSUR*, 41(3):Article 15, 1–58, 2009.

[10] X. H. Dang, I. Assent, R. T. Ng, A. Zimek, and E. Schubert. Discriminative features for identifying and interpreting outliers. In *Proc. ICDE*, 2014.

[11] J. M. Geusebroek, G. J. Burghouts, and A. W. M. Smeulders. The Amsterdam Library of Object Images. *Int. J. Computer Vision*, 61(1):103–112, 2005.

[12] F. R. Hampel. The influence curve and its role in robust estimation. *JASA*, 69(346):383–393, 1974.

[13] W. Jin, A. K. H. Tung, J. Han, and W. Wang. Ranking outliers using symmetric neighborhood relationship. In *Proc. PAKDD*, pages 577–593, 2006.

[14] F. Keller, E. Müller, and K. Böhm. HiCS: high contrast subspaces for density-based outlier ranking. In *Proc. ICDE*, 2012.

[15] E. M. Knorr and R. T. Ng. A unified notion of outliers: Properties and computation. In *Proc. KDD*, 1997.

[16] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek. LoOP: local outlier probabilities. In *Proc. CIKM*, 2009.

[17] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek. Outlier detection in arbitrarily oriented subspaces. In *Proc. ICDM*, pages 379–388, 2012.

[18] L. J. Latecki, A. Lazarevic, and D. Pokrajac. Outlier detection with kernel density functions. In *Proc. MLDM*, 2007.

[19] D. O. Loftsgaarden and C. P. Quesenberry. A nonparametric estimate of a multivariate density function. *Annals Math. Stat.*, 36(3):1049–1051, 1965.

[20] P. Mills. Efficient statistical classification of satellite measurements. *Int. J. Remote Sensing*, 32(21):6109–6132, 2011.

[21] S. Papadimitriou, H. Kitagawa, P. Gibbons, and C. Faloutsos. LOCI: Fast outlier detection using the local correlation integral. In *Proc. ICDE*, pages 315–326, 2003.

[22] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *Proc. SIGMOD*, pages 427–438, 2000.

[23] E. Schubert, R. Wojdanowski, A. Zimek, and H.-P. Kriegel. On evaluation of outlier rankings and outlier scores. In *Proc. SDM*, pages 1047–1058, 2012.

[24] E. Schubert, A. Zimek, and H.-P. Kriegel. Geodetic distance queries on R-trees for indexing geographic data. In *Proc. SSTD*, pages 146–164, 2013.

[25] E. Schubert, A. Zimek, and H.-P. Kriegel. Local outlier detection reconsidered: a generalized view on locality with applications to spatial, video, and network outlier detection. *Data Min. Knowl. Disc.*, 28(1):190–237, 2014.

[26] D. Scott and S. Sain. Multi-dimensional density estimation. *Handbook of Statistics*, 24:229–261, 2005.

[27] S. J. Sheather and M. C. Jones. A reliable data-based bandwidth selection method for kernel density estimation. *J. R. Statist. Soc. B*, pages 683–690, 1991.

[28] B. W. Silverman. Algorithm AS 176: Kernel density estimation using the fast Fourier transform. *Appl. Statist.*, 31(1):93–99, 1982.

[29] J. Tang, Z. Chen, A. W.-C. Fu, and D. W. Cheung. Enhancing effectiveness of outlier detections for low density patterns. In *Proc. PAKDD*, pages 535–548, 2002.

[30] G. Terrell and D. Scott. Variable kernel density estimation. *Annals Stat.*, 20(3):1236–1265, 1992.

[31] K. Zhang, M. Hutter, and H. Jin. A new local distance-based outlier detection approach for scattered real-world data. In *Proc. PAKDD*, pages 813–822, 2009.

[32] A. Zimek, R. J. G. B. Campello, and J. Sander. Ensembles for unsupervised outlier detection: Challenges and research questions [position paper]. *SIGKDD Explor.*, 15(1), 2013.

[33] A. Zimek, M. Gaudet, R. J. G. B. Campello, and J. Sander. Subsampling for efficient and effective unsupervised outlier detection ensembles. In *Proc. KDD*, 2013.

[34] A. Zimek, E. Schubert, and H.-P. Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Stat. Anal. Data Min.*, 5(5):363–387, 2012.