Inlier-based Outlier Detection via Direct Density Ratio Estimation

Shohei Hido Yuta Tsuboi Hisashi Kashima
Tokyo Research Laboratory
IBM Research, Japan
{ hido, yutat, hkashima } @jp.ibm.com

Masashi Sugiyama
Department of Computer Science
Tokyo Institute of Technology, Japan
sugi@cs.titech.ac.jp

Takafumi Kanamori
Department of Computer Science and Mathematics Informatics
Nagoya University, Japan
kanamori@is.nagoya-u.ac.jp

Abstract

We propose a new statistical approach to the problem of inlier-based outlier detection, i.e., finding outliers in the test set based on the training set consisting only of inliers. Our key idea is to use the ratio *of training and test data densities as an outlier score; we estimate the ratio directly in a semi-parametric fashion without going through density estimation. Thus our approach is expected to have better performance in high-dimensional problems. Furthermore, the applied algorithm for density ratio estimation is equipped with a natural cross-validation procedure, allowing us to objectively optimize the value of tuning parameters such as the regularization parameter and the kernel width. The algorithm offers a closed-form solution as well as a closed-form formula for the leave-one-out error. Thanks to this, the proposed outlier detection method is computationally very efficient and is scalable to massive datasets. Simulations with benchmark and real-world datasets illustrate the usefulness of the proposed approach.*

Keywords*: outlier detection, density ratio, importance*

1 Introduction

The goal of *outlier detection* (a.k.a. *anomaly detection*, *novelty detection*, or *one-class classification*) is to find uncommon instances ('outliers') in a given dataset. Outlier detection is useful in various applications such as topic detection in news documents [14], intrusion detection in network systems [24], and defect detection from behavior patterns of industrial machines [3, 9]. For this reason, outlier detection has been studied thoroughly in statistics, machine learning, and data mining communities for decades [7].

A standard outlier detection problem falls into the category of *unsupervised learning* due to lack of prior knowledge on the 'anomalous data'. In contrast, the papers [4, 5] addressed a *semi-supervised* outlier detection problem where examples of outlier and inlier are available as a training set. The semi-supervised outlier detection methods could perform better than unsupervised methods thanks to additional label information, but such training samples are not always available in practice. Furthermore, the type of outliers may be diverse and thus the semi-supervised methods—learning from *known* types of outliers—are not necessarily useful in detecting *unknown* types of outliers.

In this paper, we address a problem of *inlier-based* outlier detection where examples of inlier are available. More formally, the inlier-based outlier detection problem is to find outlier instances in the test set based on the training set consisting only of inlier instances. The setting of inlier-based outlier detection would be more practical than the semi-supervised setting since inlier samples are often available abundantly. For example, in defect detection of industrial machines, we already know that there is no outlier (i.e., a defect) in the past since no failure has been observed in the machinery. Therefore, it is reasonable to separate the measurement data into a training set consisting only of inlier samples observed in the past and the test set consisting of recent samples from which we try to find outliers.

As opposed to supervised learning, the outlier detection problem is vague and it is not possible to universally define what the outliers are. In this paper, we consider a statistical framework and regard instances with low probability densities as outliers. In light of inlier-based outlier detection, outliers may be identified via density estimation of inlier samples. However, density estimation is known to be a hard problem particularly in high dimensions, so outlier detection via density estimation may not work well in practice.

To avoid density estimation, we may use *One-class Sup-*

port Vector Machine (OSVM) [19] or *Support Vector Data Description (SVDD)* [23], which finds an inlier region containing a certain fraction of training instances; samples outside the inlier region are regarded as outliers. However, these methods cannot make use of inlier information available in the inlier-based settings. Furthermore, the solutions of OSVM and SVDD depend heavily on the choice of tuning parameters (e.g., the Gaussian kernel width) and there seems to be no reasonable method to appropriately determine the values of the tuning parameters.

To overcome the weakness of the existing methods, we propose a new approach to inlier-based outlier detection. Our key idea is not to directly model the training and test data densities, but only to estimate the *ratio* of training and test data densities in a semi-parametric fashion. Among existing methods of density ratio estimation [1, 8, 10, 16, 21, 22], we adopt an algorithm called *unconstrained Least-Squares Importance Fitting (uLSIF)* [10] for outlier detection. The reason for this choice is that uLSIF is equipped with a variant of cross-validation, so the values of tuning parameters such as the regularization parameter can be objectively determined without subjective trial and error. Furthermore, uLSIF-based outlier detection allows us to compute the outlier score just by solving a system of linear equations—the leave-one-out cross-validation error can also be computed analytically. Thus, the proposed method is computationally very efficient and therefore is scalable to massive datasets. Through experiments using benchmark datasets and a real-world dataset of failure detection in hard disk drives, our approach is shown to compare favorably with existing outlier detection methods and other density ratio estimation methods with higher scalability.

2 Outlier Detection via Direct Importance Estimation

In this section, we propose a new statistical approach to outlier detection.

Suppose we have two sets of samples—training samples $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$ and test samples $\{\boldsymbol{x}_i^{\mathrm{te}}\}_{i=1}^{n_{\mathrm{te}}}$ in a domain $\mathcal{D}$ $(\subset \mathbb{R}^d)$. The training samples $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$ are all inliers, while the test samples $\{\boldsymbol{x}_i^{\mathrm{te}}\}_{i=1}^{n_{\mathrm{te}}}$ can contain some outliers. The goal of outlier detection here is to identify outliers in the test set based on the training set consisting only of inliers. More formally, we want to assign a suitable *inlier score* for the test samples—the smaller the value of the inlier score is, the more plausible the sample is an outlier.

Let us consider a statistical framework of the outlier detection problem: suppose training samples $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$ are independent and identically distributed (i.i.d.) following a training data distribution with density $p_{\mathrm{tr}}(\boldsymbol{x})$ and test samples $\{\boldsymbol{x}_i^{\mathrm{te}}\}_{i=1}^{n_{\mathrm{te}}}$ are i.i.d. following a test data distribution with strictly positive density $p_{\mathrm{te}}(\boldsymbol{x})$. Within this statistical framework, test samples with low training data densities are regarded as outliers. However, $p_{\mathrm{tr}}(\boldsymbol{x})$ is not accessible in practice and density estimation is known to be a hard problem. Therefore, merely using the training data density as an inlier score may not be promising in practice.

In this paper, we propose using the ratio of training and test data densities, called the *importance*, as an inlier score:

$$w(\boldsymbol{x}) = \frac{p_{\mathrm{tr}}(\boldsymbol{x})}{p_{\mathrm{te}}(\boldsymbol{x})}.$$

If there exists no outlier sample in the test set (i.e., the training and test data densities are equivalent), the value of the importance is one. The importance value tends to be small in the regions where the training data density is low and the test data density is high. Thus samples with small importance values are plausible to be outliers.

One may suspect that this importance-based approach is not suitable when there exist only a small number of outliers—since a small number of outliers cannot increase the values of $p_{\mathrm{te}}(\boldsymbol{x})$ significantly. However, outliers are drawn from a region with small $p_{\mathrm{tr}}(\boldsymbol{x})$ and therefore a small change in $p_{\mathrm{te}}(\boldsymbol{x})$ significantly reduces the importance value. For example, let the increase of $p_{\mathrm{te}}(\boldsymbol{x})$ be $\epsilon = 0.01$; then $\frac{1}{1+\epsilon} \approx 1$, but $\frac{0.001}{0.001+\epsilon} \ll 1$. Thus the importance $w(\boldsymbol{x})$ would be a suitable inlier score (see Section 4.3 for illustrative examples).

3 Direct Importance Estimation Methods

The values of the importance are unknown in practice, so we need to estimate them from the data samples. If we estimate the training and test densities from the data samples, it can suffer from the *curse of dimensionality*. So we would like to *directly* estimate the importance values without going through density estimation. In this section, we review such direct importance estimation methods which could be used for outlier detection.

3.1 Kernel Mean Matching (KMM)

The KMM method avoids density estimation and directly gives an estimate of the importance at training points [8].

The basic idea of KMM is to find $\widehat{w}(\boldsymbol{x})$ such that the mean discrepancy between nonlinearly transformed samples drawn from $p_{\mathrm{tr}}(\boldsymbol{x})$ and $p_{\mathrm{te}}(\boldsymbol{x})$ is minimized in a *universal reproducing kernel Hilbert space* [20]. The Gaussian kernel

$$K_\sigma(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2\sigma^2}\right) \qquad (1)$$

is an example of kernels that induce a universal reproducing kernel Hilbert space. It has been shown that the solution of the following optimization problem agrees with the true importance:

$$\min_{w(\boldsymbol{x})} \left\| \int K_\sigma(\boldsymbol{x},\cdot)p_{\mathrm{tr}}(\boldsymbol{x})d\boldsymbol{x} - \int K_\sigma(\boldsymbol{x},\cdot)w(\boldsymbol{x})p_{\mathrm{te}}(\boldsymbol{x})d\boldsymbol{x} \right\|_{\mathcal{F}}^2$$
$$\text{s.t.} \int w(\boldsymbol{x})p_{\mathrm{te}}(\boldsymbol{x})d\boldsymbol{x} = 1 \text{ and } w(\boldsymbol{x}) \geq 0,$$

where $\|\cdot\|_{\mathcal{F}}$ denotes the norm in the Gaussian reproducing kernel Hilbert space.

An empirical version of the above problem is reduced to the following quadratic program:

$$\min_{\{w_i\}_{i=1}^{n_{\mathrm{te}}}} \left[\frac{1}{2} \sum_{i,i'=1}^{n_{\mathrm{te}}} w_i w_{i'} K_\sigma(\boldsymbol{x}_i^{\mathrm{te}}, \boldsymbol{x}_{i'}^{\mathrm{te}}) - \sum_{i=1}^{n_{\mathrm{te}}} w_i \kappa_i \right]$$
$$\text{s.t.} \left| \sum_{i=1}^{n_{\mathrm{te}}} w_i - n_{\mathrm{te}} \right| \leq n_{\mathrm{te}}\epsilon \text{ and } 0 \leq w_1, \ldots, w_{n_{\mathrm{te}}} \leq B,$$

where

$$\kappa_i = \frac{n_{\mathrm{te}}}{n_{\mathrm{tr}}} \sum_{j=1}^{n_{\mathrm{tr}}} K_\sigma(\boldsymbol{x}_i^{\mathrm{te}}, \boldsymbol{x}_j^{\mathrm{tr}}).$$

σ (≥ 0), B (≥ 0), and ϵ (≥ 0) are tuning parameters. The solution $\{\widehat{w}_i\}_{i=1}^{n_{\mathrm{te}}}$ is an estimate of the importance at the test points $\{\boldsymbol{x}_i^{\mathrm{te}}\}_{i=1}^{n_{\mathrm{te}}}$.

Since KMM does not require the density estimates, it is expected to work well. However, the performance of KMM is dependent on the tuning parameters B, ϵ, and σ and they cannot be simply optimized, e.g., by cross-validation since estimates of the importance are available only at the test points.

3.2 Logistic Regression (LogReg)

Another approach to directly estimating the importance is to use a probabilistic classifier. Let us assign a selector variable $\eta = 1$ to training samples and $\eta = -1$ to test samples, i.e., the training and test densities are written as

$$p_{\mathrm{tr}}(\boldsymbol{x}) = p(\boldsymbol{x}|\eta = 1) \text{ and } p_{\mathrm{te}}(\boldsymbol{x}) = p(\boldsymbol{x}|\eta = -1).$$

Application of the Bayes theorem yields that the importance can be expressed in terms of η as follows [1]:

$$w(\boldsymbol{x}) \propto \frac{p(\eta = 1|\boldsymbol{x})}{p(\eta = -1|\boldsymbol{x})}.$$

The conditional probability $p(\eta|\boldsymbol{x})$ could be approximated by discriminating test samples from training samples using a LogReg classifier, where η plays the role of a class variable. Below we briefly explain the LogReg method.

The LogReg classifier employs the following parametric model for expressing the conditional probability $p(\eta|\boldsymbol{x})$:

$$\widehat{p}(\eta|\boldsymbol{x}) = \left\{ 1 + \exp\left(-\eta \sum_{\ell=1}^{m} \zeta_\ell \phi_\ell(\boldsymbol{x}) \right) \right\}^{-1},$$

where m is the number of basis functions and $\{\phi_\ell(\boldsymbol{x})\}_{\ell=1}^m$ are fixed basis functions. The parameter $\boldsymbol{\zeta}$ is learned so that the negative regularized log-likelihood is minimized:

$$\min_{\boldsymbol{\zeta}} \left[\sum_{i=1}^{n_{\mathrm{te}}} \log\left(1 + \exp\left(\sum_{\ell=1}^{m} \zeta_\ell \phi_\ell(\boldsymbol{x}_i^{\mathrm{te}})\right)\right) + \sum_{j=1}^{n_{\mathrm{tr}}} \log\left(1 + \exp\left(-\sum_{\ell=1}^{m} \zeta_\ell \phi_\ell(\boldsymbol{x}^{\mathrm{tr}})\right)\right) + \lambda \boldsymbol{\zeta}^\top \boldsymbol{\zeta} \right].$$

Since the above objective function is convex, the global optimal solution can be obtained by standard nonlinear optimization methods such as Newton's method, conjugate gradient, and the BFGS method. Then the importance estimate is given by

$$\widehat{w}(\boldsymbol{x}) = \exp\left(\sum_{\ell=1}^{m} \zeta_\ell \phi_\ell(\boldsymbol{x})\right).$$

An advantage of the LogReg method is that model selection (i.e., the choice of basis functions $\{\phi_\ell(\boldsymbol{x})\}_{\ell=1}^m$ as well as the regularization parameter λ) is possible by standard cross-validation since the learning problem involved above is a standard supervised classification problem.

3.3 Kullback-Leibler Importance Estimation Procedure (KLIEP)

KLIEP [21, 22] also directly gives an estimate of the importance function without going through density estimation by matching the two distributions in terms of the Kullback-Leibler divergence.

Let us model the importance $w(\boldsymbol{x})$ by the following linear model:

$$\widehat{w}(\boldsymbol{x}) = \sum_{\ell=1}^{b} \alpha_\ell \varphi_\ell(\boldsymbol{x}), \tag{2}$$

where $\{\alpha_\ell\}_{\ell=1}^b$ are parameters and $\{\varphi_\ell(\boldsymbol{x})\}_{\ell=1}^b$ are basis functions such that $\varphi_\ell(\boldsymbol{x}) \geq 0$ for all $\boldsymbol{x} \in \mathcal{D}$ and for $\ell = 1, \ldots, b$. Then an estimate of the training data density $p_{\mathrm{tr}}(\boldsymbol{x})$ is given by

$$\widehat{p}_{\mathrm{tr}}(\boldsymbol{x}) = \widehat{w}(\boldsymbol{x})p_{\mathrm{te}}(\boldsymbol{x}).$$

In KLIEP, the parameters $\boldsymbol{\alpha}$ are determined so that the Kullback-Leibler divergence from $p_{\mathrm{tr}}(\boldsymbol{x})$ to $\widehat{p}_{\mathrm{tr}}(\boldsymbol{x})$ is minimized:

$$\mathrm{KL}[p_{\mathrm{tr}}(\boldsymbol{x})\|\widehat{p}_{\mathrm{tr}}(\boldsymbol{x})] = \int p_{\mathrm{tr}}(\boldsymbol{x}) \log \frac{p_{\mathrm{tr}}(\boldsymbol{x})}{\widehat{w}(\boldsymbol{x})p_{\mathrm{te}}(\boldsymbol{x})} d\boldsymbol{x}$$
$$= \int p_{\mathrm{tr}}(\boldsymbol{x}) \log \frac{p_{\mathrm{tr}}(\boldsymbol{x})}{p_{\mathrm{te}}(\boldsymbol{x})} d\boldsymbol{x} - \int p_{\mathrm{tr}}(\boldsymbol{x}) \log \widehat{w}(\boldsymbol{x}) d\boldsymbol{x}. \quad (3)$$

The first term is a constant, so it can be safely ignored. Since $\widehat{p}_{\mathrm{tr}}(\boldsymbol{x})$ $(= \widehat{w}(\boldsymbol{x})p_{\mathrm{te}}(\boldsymbol{x}))$ is a probability density function, it should satisfy

$$1 = \int \widehat{p}_{\mathrm{tr}}(\boldsymbol{x}) d\boldsymbol{x} = \int \widehat{w}(\boldsymbol{x}) p_{\mathrm{te}}(\boldsymbol{x}) d\boldsymbol{x}. \quad (4)$$

The KLIEP optimization problem is given by replacing the expectations in Eqs. (3) and (4) with empirical averages:

$$\max_{\{\alpha_\ell\}_{\ell=1}^{b}} \left[\sum_{j=1}^{n_{\mathrm{tr}}} \log \left(\sum_{\ell=1}^{b} \alpha_\ell \varphi_\ell(\boldsymbol{x}_j^{\mathrm{tr}}) \right) \right]$$
$$\text{s.t.} \sum_{\ell=1}^{b} \alpha_\ell \left(\sum_{i=1}^{n_{\mathrm{te}}} \varphi_\ell(\boldsymbol{x}_i^{\mathrm{te}}) \right) = n_{\mathrm{te}} \text{ and } \alpha_1, \ldots, \alpha_b \geq 0.$$

This is a convex optimization problem and the global solution—which tends to be sparse—can be obtained, e.g., by simply performing gradient ascent and feasibility satisfaction iteratively. See [16] for the convergence proof.

Model selection of KLIEP is possible by a variant of *likelihood cross-validation* (LCV) [6] as follows. We first divide the training samples $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$ into a training part and a validation part, the model is trained based on the training part, and then its likelihood is verified using the validation part; the model with the largest estimated likelihood is chosen. Note that this LCV procedure corresponds to choosing the model with the smallest $\mathrm{KL}[p_{\mathrm{tr}}(\boldsymbol{x})\|\widehat{p}_{\mathrm{tr}}(\boldsymbol{x})]$.

3.4 Unconstrained Least-Squares Importance Fitting (uLSIF)

In uLSIF, the linear importance model (2) is used and the parameters are determined so that the following objective function is minimized [10]:

$$\frac{1}{2} \int \left(\widehat{w}(\boldsymbol{x}) - \frac{p_{\mathrm{tr}}(\boldsymbol{x})}{p_{\mathrm{te}}(\boldsymbol{x})} \right)^2 p_{\mathrm{te}}(\boldsymbol{x}) d\boldsymbol{x}$$
$$= \frac{1}{2} \int \widehat{w}(\boldsymbol{x})^2 p_{\mathrm{te}}(\boldsymbol{x}) d\boldsymbol{x} - \int \widehat{w}(\boldsymbol{x}) p_{\mathrm{tr}}(\boldsymbol{x}) d\boldsymbol{x} + \frac{1}{2} \int \frac{p_{\mathrm{tr}}(\boldsymbol{x})^2}{p_{\mathrm{te}}(\boldsymbol{x})} d\boldsymbol{x},$$

where the last term in the right-hand side is a constant and therefore can be safely ignored. By the empirical approximation, the following optimization problem is obtained.

$$\widetilde{\boldsymbol{\alpha}} = \operatorname*{argmin}_{\boldsymbol{\alpha}} \left[\frac{1}{2} \boldsymbol{\alpha}^\top \widehat{\boldsymbol{H}} \boldsymbol{\alpha} - \widehat{\boldsymbol{h}}^\top \boldsymbol{\alpha} + \lambda \boldsymbol{\alpha}^\top \boldsymbol{\alpha} \right],$$

where, for $\boldsymbol{\varphi}(\boldsymbol{x}) = (\varphi_1(\boldsymbol{x}), \ldots, \varphi_b(\boldsymbol{x}))^\top$,

$$\widehat{\boldsymbol{H}} = \frac{1}{n_{\mathrm{te}}} \sum_{i=1}^{n_{\mathrm{te}}} \boldsymbol{\varphi}(\boldsymbol{x}_i^{\mathrm{te}}) \boldsymbol{\varphi}(\boldsymbol{x}_i^{\mathrm{te}}) \text{ and } \widehat{\boldsymbol{h}} = \frac{1}{n_{\mathrm{tr}}} \sum_{j=1}^{n_{\mathrm{tr}}} \boldsymbol{\varphi}(\boldsymbol{x}_j^{\mathrm{tr}}).$$

$\lambda \boldsymbol{\alpha}^\top \boldsymbol{\alpha}$ is a regularization term. The solution $\widetilde{\boldsymbol{\alpha}}$ is given *analytically* by

$$\widetilde{\boldsymbol{\alpha}} = (\widehat{\boldsymbol{H}} + \lambda \boldsymbol{I}_b)^{-1} \widehat{\boldsymbol{h}},$$

where $\boldsymbol{I}_b$ is the b-dimensional identity matrix. Since elements of $\widetilde{\boldsymbol{\alpha}}$ could be negative, it is modified as

$$\widehat{\boldsymbol{\alpha}} = \max(\mathbf{0}_b, \widetilde{\boldsymbol{\alpha}}),$$

where $\mathbf{0}_b$ is b-dimensional vector with all zeros. This is the solution of uLSIF, which can be computed analytically.

Let us consider the leave-one-out cross-validation (LOOCV) score of uLSIF:

$$\frac{1}{n_{\mathrm{te}}} \sum_{i=1}^{n_{\mathrm{te}}} \left[\frac{1}{2} \left(\boldsymbol{\varphi}(\boldsymbol{x}_i^{\mathrm{te}})^\top \widehat{\boldsymbol{\alpha}}_\lambda^{(i)} \right)^2 - \boldsymbol{\varphi}(\boldsymbol{x}_i^{\mathrm{tr}})^\top \widehat{\boldsymbol{\alpha}}_\lambda^{(i)} \right],$$

where $\widehat{\boldsymbol{\alpha}}_\lambda^{(i)}$ is a parameter learned without $\boldsymbol{x}_i^{\mathrm{te}}$ and $\boldsymbol{x}_i^{\mathrm{tr}}$. By using the well-known Woodbury inversion formula, $\widehat{\boldsymbol{\alpha}}_\lambda^{(i)}$ can be expressed as

$$\widehat{\boldsymbol{\alpha}}_\lambda^{(i)} = \max \Bigg\{ \mathbf{0}_b, \frac{(n_{\mathrm{te}}-1)n_{\mathrm{tr}}}{n_{\mathrm{te}}(n_{\mathrm{tr}}-1)} \left(\boldsymbol{a} + \frac{\boldsymbol{\varphi}(\boldsymbol{x}_i^{\mathrm{te}})^\top \boldsymbol{a}}{n_{\mathrm{te}} - \boldsymbol{\varphi}(\boldsymbol{x}_i^{\mathrm{te}})^\top \boldsymbol{a}_{\mathrm{te}}} \boldsymbol{a}_{\mathrm{te}} \right)$$
$$- \frac{(n_{\mathrm{te}}-1)}{n_{\mathrm{te}}(n_{\mathrm{tr}}-1)} \left(\boldsymbol{a}_{\mathrm{tr}} + \frac{\boldsymbol{\varphi}(\boldsymbol{x}_i^{\mathrm{te}})^\top \boldsymbol{a}_{\mathrm{tr}}}{n_{\mathrm{te}} - \boldsymbol{\varphi}(\boldsymbol{x}_i^{\mathrm{te}})^\top \boldsymbol{a}_{\mathrm{te}}} \boldsymbol{a}_{\mathrm{te}} \right) \Bigg\},$$

where

$$\boldsymbol{a} = \boldsymbol{A}^{-1} \widehat{\boldsymbol{h}}, \quad \boldsymbol{a}_{\mathrm{te}} = \boldsymbol{A}^{-1} \boldsymbol{\varphi}(\boldsymbol{x}_i^{\mathrm{te}}),$$
$$\boldsymbol{a}_{\mathrm{tr}} = \boldsymbol{A}^{-1} \boldsymbol{\varphi}(\boldsymbol{x}_i^{\mathrm{tr}}), \quad \boldsymbol{A} = \widehat{\boldsymbol{H}} + \frac{(n_{\mathrm{te}}-1)\lambda}{n_{\mathrm{te}}} \boldsymbol{I}_b.$$

This expression implies that the matrix inverse needs to be computed only once (i.e., $\boldsymbol{A}^{-1}$) for computing the LOOCV score. Note that the size of $\boldsymbol{A}^{-1}$ is $b \times b$, which is independent of the numbers of training and test samples. Thus LOOCV can be carried out very efficiently without repeating the hold-out loop.

4 Outlier Detection by uLSIF

In this section, we discuss the characteristics of importance estimation methods reviewed in the previous section and propose a practical outlier detection procedure based on uLSIF.

4.1 Discussions

For KMM, there is no objective model selection method. Therefore, model parameters such as the Gaussian width need to be determined by hand, which is highly subjective in outlier detection. On the other hand, LogReg and KLIEP give an estimate of the entire importance function. Therefore, the importance values at unseen points can be estimated and CV becomes available for model selection. However, LogReg and KLIEP are computationally rather expensive since non-linear optimization problems have to be solved.

uLSIF inherits the preferable properties of LogReg and KLIEP. Furthermore, the solution of uLSIF can be computed analytically through matrix inversion and therefore uLSIF is computationally very efficient. Thanks to the availability of the closed-form solution, the LOOCV score can also be analytically computed without repeating the hold-out loop, which highly contributes to reducing the computation time in the model selection phase.

Based on the above discussion, we decided to use uLSIF in our outlier detection procedure.

4.2 Heuristic of Basis Function Choice

In uLSIF, a good model may be chosen by LOOCV, given that a set of promising model candidates is prepared. Here we propose using a Gaussian kernel model centered at the training points $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$ as model candidates, i.e.,

$$\widehat{w}(\boldsymbol{x}) = \sum_{\ell=1}^{n_{\mathrm{tr}}} \alpha_\ell K_\sigma(\boldsymbol{x}, \boldsymbol{x}_\ell^{\mathrm{tr}}),$$

where $K_\sigma(\boldsymbol{x}, \boldsymbol{x}')$ is the Gaussian kernel (1) with kernel width σ.

The reason why the training points $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$ are chosen as the Gaussian centers, not the test points $\{\boldsymbol{x}_i^{\mathrm{te}}\}_{i=1}^{n_{\mathrm{te}}}$, is as follows. By definition, the importance $w(\boldsymbol{x})$ tends to take large values if the training density $p_{\mathrm{tr}}(\boldsymbol{x})$ is large and the test density $p_{\mathrm{te}}(\boldsymbol{x})$ is small; conversely, $w(\boldsymbol{x})$ tends to be small (i.e., close to zero) if $p_{\mathrm{tr}}(\boldsymbol{x})$ is small and $p_{\mathrm{te}}(\boldsymbol{x})$ is large. When a function is approximated by a Gaussian kernel model, many kernels may be needed in the region where the output of the target function is large; on the other hand, only a small number of kernels would be enough in the region where the output of the target function is close to zero. Following this heuristic, we decided to allocate many kernels at high training density regions, which can be achieved by setting the Gaussian centers at the training points $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$.

Alternatively, we may locate $(n_{\mathrm{tr}} + n_{\mathrm{te}})$ Gaussian kernels at both $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$ and $\{\boldsymbol{x}_i^{\mathrm{te}}\}_{i=1}^{n_{\mathrm{te}}}$. However, in our preliminary experiments, this did not further improve the performance, but just slightly increased the computational cost. Since n_{tr} is typically very large, just using all the training points $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$ as Gaussian centers is already computationally rather demanding. To ease this problem, we practically propose using a subset of $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$ as Gaussian centers for computational efficiency, i.e.,

$$\widehat{w}(\boldsymbol{x}) = \sum_{\ell=1}^{b} \alpha_\ell K_\sigma(\boldsymbol{x}, \boldsymbol{c}_\ell),$$

where $\boldsymbol{c}_\ell$ is a template point randomly chosen from $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$.

We use the above basis functions in LogReg, KLIEP, and uLSIF in the experiments.

4.3 Illustrative Examples

Here, we illustrate how uLSIF behaves in outlier detection.

4.3.1 Toy Dataset

Let the dimension of the data domain be $d = 1$, and let the training density be

(a) $p_{\mathrm{tr}}(x) = \mathcal{N}(x; 0, 1)$,

(b) $p_{\mathrm{tr}}(x) = 0.5\mathcal{N}(x; -5, 1) + 0.5\mathcal{N}(x; 5, 1)$,

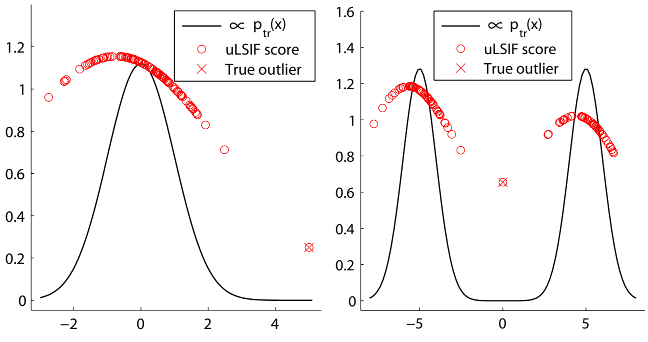
where $\mathcal{N}(x; \mu, \sigma^2)$ denotes the Gaussian density with mean μ and variance σ^2. We draw $n_{\mathrm{tr}} = 300$ training samples and 99 test samples from $p_{\mathrm{tr}}(x)$, and we add an outlier sample at $x = 5$ for the case (a) and at $x = 0$ for the case (b) in the test set; thus the total number of test samples is $n_{\mathrm{te}} = 100$. The number of basis functions in uLSIF is fixed at $b = 100$, and the Gaussian width σ and the regularization parameter λ are chosen from a wide range of values based on LOOCV. The data densities as well as the importance values (i.e., the inlier scores) obtained by uLSIF are depicted in Figure 1. The graphs show that the outlier sample has the smallest inlier score among all samples and therefore the outlier can be successfully detected.

Since the solution of uLSIF tends to be sparse, it may be natural to have a Gaussian-like curve as the inlier score (see Figure 1 again).

4.3.2 USPS Dataset

USPS is a dataset which contains images of hand-written digits provided by U.S. Postal Service. Each image consists of 256 $(= 16 \times 16)$ pixels, each of which takes a value between -1 to $+1$ representing its color in gray-scale. The class labels attached to the images are integers between 0 and 9 denoting the digits the images represent. Here, we try to find irregular samples in the USPS dataset by uLSIF.

To the 256-dimensional image vectors, we append 10 additional dimensions indicating the true class to identify mislabeled images. In uLSIF, we set $b = 100$ and σ and λ are



(a) $\mathcal{N}(x; 0, 1)$ (b) $\frac{1}{2}(\mathcal{N}(x; -5, 1) + \mathcal{N}(x; 5, 1))$

Figure 1. Illustration of uLSIF-based outlier detection.

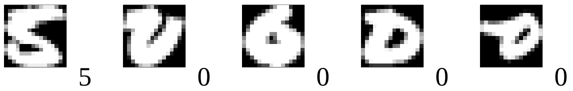


Figure 2. Outliers in the USPS test set.

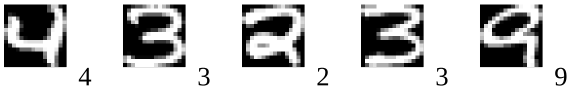


Figure 3. Outliers in the USPS training set.

chosen from a wide range of values based on LOOCV. Figure 2 shows the top 5 outlier samples in the original USPS test set (of size 2007) found by uLSIF where their original labels are attached next to the images. This result clearly shows that the proposed method successfully detects outlier samples that are very hard to recognize even by humans.

We also consider an inverse scenario: we switch the training and test sets and examine the original USPS training set (of size 7291). Figure 3 depicts the top 5 outliers found by uLSIF, showing that they are relatively 'good' samples. This implies that the USPS training set consists only of high-quality samples.

5 Relation to Existing Outlier Detection Methods

In this section, we discuss the relation between the proposed density-ratio based outlier detection approach with existing outlier detection methods.

The outlier detection problem we are addressing in this paper is to find outliers in the test set $\{\boldsymbol{x}_i^{\mathrm{te}}\}_{i=1}^{n_{\mathrm{te}}}$ based on the training set $\{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}}$ consisting only of inliers. On the other hand, the outlier detection problem that the existing methods reviewed here are solving is to find outliers in the test set without the training set. Thus the setting is slightly different. However, the existing methods can also be employed in our setting by simply using the union of training and test samples as a test set: $\{\boldsymbol{x}_k\}_{k=1}^{n} = \{\boldsymbol{x}_j^{\mathrm{tr}}\}_{j=1}^{n_{\mathrm{tr}}} \cup \{\boldsymbol{x}_i^{\mathrm{te}}\}_{i=1}^{n_{\mathrm{te}}}$.

5.1 Kernel Density Estimator (KDE)

KDE is a non-parametric technique to estimate a density $p(\boldsymbol{x})$ from samples $\{\boldsymbol{x}_k\}_{k=1}^{n}$. KDE with the Gaussian kernel is expressed as

$$\widehat{p}(\boldsymbol{x}) = \frac{1}{n_{\mathrm{te}}(2\pi\sigma^2)^{d/2}} \sum_{k=1}^{n} K_\sigma(\boldsymbol{x}, \boldsymbol{x}_k),$$

where $K_\sigma(\boldsymbol{x}, \boldsymbol{x}')$ is the Gaussian kernel (1).

The performance of KDE depends on the choice of the kernel width σ, but its value can be objectively determined based on LCV [6]: a subset of $\{\boldsymbol{x}_k\}_{k=1}^{n}$ is used for density estimation and the rest is used for estimating the likelihood of the held-out samples. Note that this LCV procedure corresponds to choosing σ such that the Kullback-Leibler divergence from $p(\boldsymbol{x})$ to $\widehat{p}(\boldsymbol{x})$ is minimized. The estimated density values could be directly used as an inlier score. A variation of the KDE approach has been studied in the paper [11], where local outliers are detected from multi-modal datasets.

However, density estimation is known to suffer from the *curse of dimensionality*, and therefore the KDE-based outlier detection method may not be reliable in practice[1]. In our experiment, we will use a global KDE-based outlier detection method since we do not assume the multi-modality of the datasets.

5.2 One-class Support Vector Machine (OSVM)

SVM is one of the most successful classification algorithms in machine learning. The core idea of SVM is to separate samples in different classes by the maximum margin hyperplane in a kernel-induced feature space.

OSVM is an extension of SVM to outlier detection [19]. The basic idea of OSVM is to separate data samples $\{\boldsymbol{x}_k\}_{k=1}^{n}$ into outliers and inliers by a hyperplane in a Gaussian reproducing kernel Hilbert space. More specifically, the solution of OSVM is given as the solution of the following quadratic programming problem:

$$\begin{aligned} \min_{\{w_k\}_{k=1}^{n}} \ & \frac{1}{2} \sum_{k,k'=1}^{n} w_k w_{k'} K_\sigma(\boldsymbol{x}_k, \boldsymbol{x}_{k'}) \\ \text{s.t.} \ & \sum_{k=1}^{n} w_k = 1 \text{ and } 0 \leq w_1, \ldots, w_n \leq \frac{1}{\nu n}, \end{aligned}$$

[1] The density ratio can also be estimated by KDE, i.e., first estimating the training and test densities and then taking the ratio. However, the estimation error tends to be accumulated in this two-step process and our preliminary experiments showed that this is not useful.

where ν ($0 \leq \nu \leq 1$) is the maximum fraction of outliers.

OSVM inherits the concept of SVM, so it is expected to work well. However, the OSVM solution is dependent on the outlier ratio ν and the Gaussian kernel width σ; choosing these tuning parameter values could be highly subjective in unsupervised outlier detection. This is a critical limitation in practice. Furthermore, inlier scores cannot be directly obtained by OSVM; the distance from the separating hyperplane may be used as an inlier score, but its statistical meaning is rather unclear.

A similar algorithm named *Support Vector Data Description* (SVDD) [23] is known to be equivalent to OSVM if the Gaussian kernel is used.

5.3 Local Outlier Factor (LOF)

The LOF is an outlier score suitable for detecting local outliers apart from dense regions [2]. The LOF value of an example $\boldsymbol{x}$ is defined using the ratio of the average distance from the nearest neighbors as

$$\mathrm{LOF}_k(\boldsymbol{x}) = \frac{1}{k}\sum_{i=1}^{k}\frac{\mathrm{lrd}_k(\mathrm{nearest}_i(\boldsymbol{x}))}{\mathrm{lrd}_k(\boldsymbol{x})},$$

where $\mathrm{nearest}_i(\boldsymbol{x})$ represents the i-th nearest neighbor of $\boldsymbol{x}$ and $\mathrm{lrd}_k(\boldsymbol{x})$ denotes the inverse of the average distance from the k nearest neighbors of $\boldsymbol{x}$. If $\boldsymbol{x}$ lies around a high density region and its nearest neighbor samples are close to each other in the high density region, $\mathrm{lrd}_k(\boldsymbol{x})$ tends to become much smaller than $\mathrm{lrd}_k(\mathrm{nearest}_i(\boldsymbol{x}))$ for every i. In such cases, $\mathrm{LOF}_k(\boldsymbol{x})$ has a large value and $\boldsymbol{x}$ is regarded as a local outlier.

Although the LOF values seem to be a suitable outlier measure, the performance strongly depends on the choice of the parameter k. To the best of our knowledge, there is no systematic method to select an appropriate value of k. In addition, the computational cost of the LOF scores is expensive since it involves a number of nearest neighbor search procedures.

5.4 Learning from Positive and Unlabeled data

A formulation called *learning from positive and unlabeled data* has been introduced in the paper [13]: given positive and unlabeled datasets, the goal is to detect positive samples contained in the unlabeled dataset. The assumption behind this formulation is that most of the unlabeled samples are negative (outlier) samples, which is different from the current outlier detection setup. In the paper [12], a modified formulation has been addressed in the context of text data analysis—the unlabeled dataset contains only a small number of negative documents. The key idea is to construct a single representative document of the negative (outlier) class based on the difference between the distributions of positive and unlabeled documents. Though the problem setup is similar to ours, the method is specialized in text data, i.e., the *bag-of-words* expression.

Since these above methods do not suit general inlier-based outlier detection scenarios, we will not include them in the experiments in Section 6.

5.5 Discussions

In summary, the proposed density-ratio based approach with direct density-ratio estimation would be more advantageous than KDE since it can avoid solving an unnecessarily difficult problem of density estimation. Compared with OSVM and LOF, the density-ratio based approach with LogReg, KLIEP, and uLSIF would be more useful since it is equipped with a model selection procedure. Furthermore, uLSIF is computationally more efficient than OSVM and LOF thanks to the analytic-form solution.

6 Experiments

In this section, we experimentally compare the performance of the proposed and existing algorithms. For all experiments, we used the standard statistical language environment *R* [17]. We implemented uLSIF, KLIEP, LogReg, KDE, and KMM by ourselves. uLSIF and KLIEP are implemented following the pseudo codes provided in the papers [10, 21, 22]. A package of the *L-BFGS-B* method called the *optim* is used in our LogReg implementation, and a quadratic program solver called the *ipop* contained in the *kernlab* package is used in our KMM implementation. We use the *ksvm* function contained in the *kernlab* package for OSVM and the *lofactor* function included in *dprep* package for LOF.

6.1 Benchmark Datasets

We use 12 datasets available from Rätsch's Benchmark Repository [18]. Note that they are originally binary classification datasets—here we regard the positive samples as inliers and the negative samples as outliers. All the negative samples are removed from the training set, i.e., the training set only contains inlier samples. In contrast, a fraction ρ of the negative samples are retained in the test set, i.e., the test set includes all inlier samples and some outliers.

When evaluating the performance of outlier detection algorithms, it is important to take into account both the *detection rate* (the amount of true outliers an outlier detection algorithm can find) and the *detection accuracy* (the amount of true inliers that an outlier detection algorithm misjudges as outliers). Since there is a trade-off between the detection rate and detection accuracy, we decided to adopt the *Area Under the ROC Curve* (AUC) as our error metric here.

We compare the AUC values of the density-ratio based methods (KMM, LogReg, KLIEP, and uLSIF) and other methods (KDE, OSVM, and LOF). All the tuning parameters included in KDE, LogReg, KLIEP and uLSIF are chosen based on CV from a wide range of values. CV is not available to KMM, OSVM, and LOF; the Gaussian kernel width in KMM and OSVM is set as the median distance between samples, which has been shown to be a useful heuristic[2]. The number of basis functions in uLSIF is fixed at $b = 100$. Note that b can also be optimized by CV, but our preliminary experimental results showed that the performance is not so sensitive to the choice of b and $b = 100$ seems to be a reasonable choice. For LOF, we test 3 different values for the number of nearest neighbors k.

The AUC values as well as the normalized computation time are summarized in Table 1, showing that uLSIF and KLIEP work very well. Though the other methods perform well for some datasets, they also exhibit poor performance in other cases. On the other hand, the performance of uLSIF and KLIEP is relatively stable. In addition, from the viewpoint of computation time, uLSIF is much faster than KLIEP and other methods. Thus, the proposed uLSIF-based method could be a reliable and computationally efficient alternative to existing outlier detection methods.

6.2 Real-world Datasets

Finally, let us consider a real-world failure prediction problem in hard-disk drives equipped with the *Self-Monitoring and Reporting Technology* (SMART). The SMART system monitors individual drives and stores some attributes (e.g., the number of read errors) as time-series data. We use the SMART dataset provided by a manufacturer [15]. The dataset consists of 369 drives, where 178 drives are labeled as good and 191 drives are labeled as failed. Each drive stores up to the last 300 records that are logged almost every 2 hours. Although each record originally includes 59 attributes, we use only 25 variables chosen based on the feature selection test [15]. The sequence of records are converted into data samples in a sliding-window manner with window size ℓ.

In practice, the training set may contain a few "bad" samples. To simulate such realistic situations, we construct the training set by choosing all records of the 178 good drives and adding a small fraction τ of '*before-fail*' examples taken from the 191 failed drives which are more than 300 hours prior to failure. The test set is made of the records of the good drives and the records of the 191 failed drives less than 100 hours prior to failure; the "fail" samples are regarded as outliers in this experiment.

First, we perform experiments for the window size $\ell = 5, 10$ and evaluate the dependence of the feature dimension on the outlier detection performance. The fraction τ of before-fail examples in the training set is fixed to 0.05. Other settings including the fraction ρ of outliers and b are the same as the previous experiments. The results are summarized in Table 2. Next, we change the fraction of before-fail examples in the training set as $\tau = 0.05, 0.1, 0.15, 0.2$ and evaluate the effect of heterogeneousness of the training set on the outlier detection performance. The fraction ρ of outliers in the test set is fixed to 0.05 and the window size ℓ is fixed to 10. The results are summarized in Table 3.

Overall, the density-ratio based methods work very well; among them, uLSIF has the lowest computational cost. The performance of OSVM tends to be degraded as the outlier fraction ρ increases and the performance of KDE rapidly gets worse as the feature dimension ℓ increases. LOF works very well if the number of nearest neighbors k is large. However, a good choice of k may be problem-dependent and there seems no systematic way to determine k appropriately. The computation of LOF is very slow due to extensive nearest neighbor search, and the performance of LOF tends to be degraded if the fraction τ of before-fail examples in the training set is increased.

These results indicate that our algorithm using the density ratio is accurate and computationally efficient in real-world failure prediction tasks—in particular, the use of KLIEP and uLSIF seems promising.

7 Concluding Remarks

We cast the inlier-based outlier detection problem as a problem of estimating the ratio of probability densities (i.e., the *importance*), and proposed a practical outlier detection algorithm based on uLSIF. Our method is equipped with a model selection procedure, which allows us to obtain a purely objective solution. This is a highly valuable feature in ill-defined problems such as outlier detection. Furthermore, the proposed method is computationally very efficient and therefore useful in practice. Through extensive simulations with benchmark and real-world datasets, the usefulness of the proposed approach was demonstrated.

References

[1] S. Bickel, M. Brückner, and T. Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of the 24th International Conference on Machine Learning*, pages 81–88, 2007.

[2] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying density-based local outliers. In *Proceedings of*

[2] We experimentally confirmed that this heuristic works reasonably well in the current experiments.

Table 1. Mean AUC values over 20 trials for the benchmark datasets.

Dataset		uLSIF	KLIEP	LogReg	KMM	OSVM	LOF			KDE
Name	ρ	(CV)	(CV)	(CV)	(med)	(med)	$k = 5$	$k = 30$	$k = 50$	(CV)
banana	0.01	0.851	0.815	0.447	0.578	0.360	0.838	0.915	0.919	0.934
	0.02	0.858	0.824	0.428	0.644	0.412	0.813	0.918	0.920	0.927
	0.05	0.869	0.851	0.435	0.761	0.467	0.786	0.907	0.909	0.923
b-cancer	0.01	0.463	0.480	0.627	0.576	0.508	0.546	0.488	0.463	0.400
	0.02	0.463	0.480	0.627	0.576	0.506	0.521	0.445	0.428	0.400
	0.05	0.463	0.480	0.627	0.576	0.498	0.549	0.480	0.452	0.400
diabetes	0.01	0.558	0.615	0.599	0.574	0.563	0.513	0.403	0.390	0.425
	0.02	0.558	0.615	0.599	0.574	0.563	0.526	0.453	0.434	0.425
	0.05	0.532	0.590	0.636	0.547	0.545	0.536	0.461	0.447	0.435
f-solar	0.01	0.416	0.485	0.438	0.494	0.522	0.480	0.441	0.385	0.378
	0.02	0.426	0.456	0.432	0.480	0.550	0.442	0.406	0.343	0.374
	0.05	0.442	0.479	0.432	0.532	0.576	0.455	0.417	0.370	0.346
german	0.01	0.574	0.572	0.556	0.529	0.535	0.526	0.559	0.552	0.561
	0.02	0.574	0.572	0.556	0.529	0.535	0.553	0.549	0.544	0.561
	0.05	0.564	0.555	0.540	0.532	0.530	0.548	0.571	0.555	0.547
heart	0.01	0.659	0.647	0.833	0.623	0.681	0.407	0.659	0.739	0.638
	0.02	0.659	0.647	0.833	0.623	0.678	0.428	0.668	0.746	0.638
	0.05	0.659	0.647	0.833	0.623	0.681	0.440	0.666	0.749	0.638
satimage	0.01	0.812	0.828	0.600	0.813	0.540	0.909	0.930	0.896	0.916
	0.02	0.829	0.847	0.632	0.861	0.548	0.785	0.919	0.880	0.898
	0.05	0.841	0.858	0.715	0.893	0.536	0.712	0.895	0.868	0.892
splice	0.01	0.713	0.748	0.368	0.541	0.737	0.765	0.778	0.768	0.845
	0.02	0.754	0.765	0.343	0.588	0.744	0.761	0.793	0.783	0.848
	0.05	0.734	0.764	0.377	0.643	0.723	0.764	0.785	0.777	0.849
thyroid	0.01	0.534	0.720	0.745	0.681	0.504	0.259	0.111	0.071	0.256
	0.02	0.534	0.720	0.745	0.681	0.505	0.259	0.111	0.071	0.256
	0.05	0.534	0.720	0.745	0.681	0.485	0.259	0.111	0.071	0.256
titanic	0.01	0.525	0.534	0.602	0.502	0.456	0.520	0.525	0.525	0.461
	0.02	0.496	0.498	0.659	0.513	0.526	0.492	0.503	0.503	0.472
	0.05	0.526	0.521	0.644	0.538	0.505	0.499	0.512	0.512	0.433
twonorm	0.01	0.905	0.902	0.161	0.439	0.846	0.812	0.889	0.897	0.875
	0.02	0.896	0.889	0.197	0.572	0.821	0.803	0.892	0.901	0.858
	0.05	0.905	0.903	0.396	0.754	0.781	0.765	0.858	0.874	0.807
waveform	0.01	0.890	0.881	0.243	0.477	0.861	0.724	0.887	0.889	0.861
	0.02	0.901	0.890	0.181	0.602	0.817	0.690	0.887	0.890	0.861
	0.05	0.885	0.873	0.236	0.757	0.798	0.705	0.847	0.874	0.831
Average		0.661	0.685	0.530	0.608	0.596	0.594	0.629	0.622	0.623
Comp. time		1.00	11.7	5.35	751	12.4	85.5			8.70

the ACM SIGMOD International Conference on Management of Data, pages 93–104, 2000.

[3] R. Fujimaki, T. Yairi, and K. Machida. An approach to spacecraft anomaly detection problem using kernel feature space. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 401–410, 2005.

[4] J. Gao, H. Cheng, and P.-N. Tan. Semi-supervised outlier detection. In *Proceedings of the 2006 ACM symposium on Applied Computing*, pages 635–636, 2006.

[5] J. Gao, H. Chengy, and P.-N. Tan. A novel framework for incorporating labeled examples into anomaly detection. In *Proceedings of the 2006 SIAM International Conference on Data Mining*, pages 593–597, 2006.

[6] W. Härdle, M. Müller, S. Sperlich, and A. Werwatz. Nonparametric and semiparametric models. *Springer Series in Statistics*, 2004.

[7] V. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

Table 2. SMART dataset: mean AUC values when changing the window size ℓ and the outlier ratio ρ

Dataset		uLSIF	KLIEP	LogReg	KMM	OSVM	LOF			KDE
ℓ	ρ	(CV)	(CV)	(CV)	(med)	(med)	$k = 5$	$k = 30$	$k = 50$	(CV)
5	0.01	0.894	0.842	0.851	0.822	0.919	0.854	0.937	0.933	0.918
	0.02	0.870	0.810	0.862	0.813	0.896	0.850	0.934	0.928	0.892
	0.05	0.885	0.858	0.888	0.849	0.864	0.789	0.911	0.923	0.883
10	0.01	0.868	0.805	0.827	0.889	0.812	0.880	0.925	0.920	0.557
	0.02	0.879	0.845	0.852	0.894	0.785	0.860	0.919	0.917	0.546
	0.05	0.889	0.857	0.856	0.898	0.783	0.849	0.915	0.916	0.619
Average		0.881	0.836	0.856	0.861	0.843	0.847	0.924	0.923	0.736
Comp. time		1.00	1.07	3.11	4.36	26.98	65.31			2.19

Table 3. SMART dataset: mean AUC values when changing heterogeneousness τ ($\rho = 0.05$ and $\ell = 10$)

Dataset	uLSIF	KLIEP	LogReg	KMM	OSVM	LOF			KDE
τ	(CV)	(CV)	(CV)	(med)	(med)	$k = 5$	$k = 30$	$k = 50$	(CV)
0.05	0.889	0.857	0.856	0.898	0.783	0.849	0.915	0.916	0.619
0.10	0.885	0.856	0.846	0.890	0.785	0.846	0.841	0.914	0.618
0.15	0.868	0.814	0.785	0.886	0.784	0.831	0.835	0.899	0.536
0.20	0.870	0.815	0.778	0.872	0.749	0.847	0.866	0.838	0.540
Average	0.878	0.836	0.816	0.887	0.775	0.843	0.864	0.892	0.578
Comp. time	1.00	1.19	3.78	5.68	30.83	74.30			2.76

[8] J. Huang, A. Smola, A. Gretton, K. Borgwardt, and B. Schölkopf. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems*, volume 19, 2007.

[9] T. Idé and H. Kashima. Eigenspace-based anomaly detection in computer systems. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 440–449, 2004.

[10] T. Kanamori, S. Hido, and M. Sugiyama. Efficient direct density ratio estimation for non-stationarity adaptation and outlier detection. In *Advances in Neural Information Processing Systems*, 2009, to appear.

[11] L. J. Latecki, A. Lazarevic, and D. Pokrajac. Outlier detection with kernel density functions. In *Proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 61–75, 2007.

[12] X. Li, B. Liu, and S.-K. Ng. Learning to identify unexpected instances in the test set. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2802–2807, 2007.

[13] B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu. Building text classifiers using positive and unlabeled examples. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 179–186, 2003.

[14] L. M. Manevitz and M. Yousef. One-class SVMs for document classification. *Journal of Machine Learning Research*, 2:139–154, 2002.

[15] J. F. Murray, G. F. Hughes, and K. Kreutz-Delgado. Machine learning methods for predicting failures in hard drives: A multiple-instance application. *Journal of Machine Learning Research*, 6:783–816, 2005.

[16] X. Nguyen, M. J. Wainwright, and M. I. Jordan. Estimating divergence functions and the likelihood ratio by penalized convex risk minimization. In *Advances in Neural Information Processing Systems 20*, pages 1089–1096, 2008.

[17] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, 2005.

[18] G. Rätsch, T. Onoda, and K. R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320, 2001.

[19] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[20] I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2001.

[21] M. Sugiyama, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in Neural Information Processing Systems 20*, pages 1433–1440, 2008.

[22] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe. Direct importance estimation for covariate shift adaptation. *Annals of the Institute of Statistical Mathematics*, 60(4), 2008. to appear.

[23] D. M. J. Tax and R. P. W. Duin. Support vector data description. *Machine Learning*, 54(1):45–66, 2004.

[24] K. Yamanishi, J.-I. Takeuchi, G. Williams, and P. Milne. Online unsupervised outlier detection using finite mixtures with discounting learning algorithms. *Data Mining and Knowledge Discovery*, 8(3):275–300, 2004.