A Density-ratio Framework for Statistical Data Processing

Masashi Sugiyama
Tokyo Institute of Technology
2-12-1 O-okayama, Meguro-ku, Tokyo 152-8552, Japan.
sugi@cs.titech.ac.jp http://sugiyama-www.cs.titech.ac.jp/~sugi

Takafumi Kanamori
Nagoya University
Furocho, Chikusaku, Nagoya 464-8603, Japan.

Taiji Suzuki
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

Shohei Hido
IBM Research
1623-14 Shimotsuruma, Yamato-shi, Kanagawa 242-8502, Japan

Jun Sese
Ochanomizu University
2-1-1 Otsuka, Bunkyo, Tokyo 112-8610, Japan

Ichiro Takeuchi
Nagoya Institute of Technology
Gokiso, Syowa-ku, Nagoya, 466-8555, Japan

Liwei Wang
Peking University
Beijing, 100871, P.R.China.

Abstract

In statistical pattern recognition, it is important to avoid density estimation since density estimation is often more difficult than pattern recognition itself. Following this idea—known as Vapnik's principle, a statistical data processing framework that employs the ratio of two probability density functions has been developed recently and is gathering a lot of attention in the machine learning and data mining communities. The purpose of this paper is to introduce to the computer vision community recent advances in density ratio estimation methods and their usage in various statistical data processing tasks such as non-stationarity adaptation, outlier detection, feature selection, and independent component analysis.

1 Introduction

Avoiding density estimation is a key to success in statistical pattern recognition since density estimation is often more difficult than pattern recognition itself. This is sometimes referred to as *Vapnik's principle* [1] and the *support vector machine* would be a successful example of this principle—instead of estimating the data generation model, it directly models the decision boundary which is sufficient for pattern recognition. Following this spirit, a new general framework of statistical data processing has been developed recently in the machine learning and data mining communities [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17]. The purpose of this article is to introduce the new framework to the computer vision community.

The key idea of the new framework is not to estimate probability densities, but to estimate the *ratio* of two probability densities. We first give a comprehensive review of methods for estimating the density ratio, including the kernel density estimator [18], kernel mean matching [4], logistic regression [19, 20, 6], the Kullback-Leibler importance estimation procedure [9, 10], least-squares importance fitting [11], and unconstrained least-squares importance fitting [11]. Note that "importance" refers to the density ratio, derived from *importance sampling* [21]. The use of the kernel density estimator results in a two-step procedure of first estimating the two densities and then plugging them into the density ratio, while recently developed methods are one-shot and directly estimate the density ratio without going through density estimation. Such direct density-ratio estimation methods were shown to be more accurate than the naive two-step approach through extensive experiments [9, 11].

Following the review of the density-ratio estimation methods, we explain how these methods could be used for solving various statistical data processing tasks such as non-stationarity adaptation [22, 2, 3, 5, 14], outlier detection [23, 24, 25, 7], and conditional density estimation [26, 27, 28, 16]. Furthermore, mutual information—which plays a central role in information theory [29]—can also be estimated via density ratio estimation. Since mutual information is a measure of statistical independence between random variables [30, 31], density ratio estimation can be used also for variable selection [32, 8, 12], dimensionality reduction [33], and independent component analysis [34, 13].

These machine learning and data mining tasks would be crucial in computer vision applications. For example, in object recognition, non-stationarity is conceivable since training images are often collected in a controlled environment such as a studio while test images will be gathered in various scenes; outlier images tend to be included due to measurement noise or occlusions. Furthermore, reducing dimensionality of the images would be indispensable for enhancing recognition accuracy.

This paper is an extended version of an earlier conference paper presented at Meeting on Image Recognition and Understanding 2008 (MIRU2008) [35].

2 Density Ratio Estimation

In this section, we formulate the problem of density ratio estimation and give a comprehensive review of density-ratio estimation methods.

2.1 Formulation

Let $\mathcal{D}$ ($\subset \mathbb{R}^d$) be the data domain and suppose we are given independent and identically distributed (i.i.d.) samples $\{\boldsymbol{x}_i\}_{i=1}^n$ from a distribution with density $q(\boldsymbol{x})$ and i.i.d. samples $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$ from another distribution with density $q'(\boldsymbol{x})$. We assume that the first density $q(\boldsymbol{x})$ is strictly positive over the domain $\mathcal{D}$, i.e.,

$$q(\boldsymbol{x}) > 0 \text{ for all } \boldsymbol{x} \in \mathcal{D}.$$

The problem we address in this article is to estimate the density ratio

$$r(\boldsymbol{x}) := \frac{q'(\boldsymbol{x})}{q(\boldsymbol{x})}$$

from samples $\{\boldsymbol{x}_i\}_{i=1}^n$ and $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$.

2.2 Kernel Density Estimator

The *kernel density estimator* (KDE) is a non-parametric technique to estimate a probability density function $p(\boldsymbol{x})$ from its i.i.d. samples $\{\boldsymbol{x}_i\}_{i=1}^n$. For the Gaussian kernel

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

KDE is expressed as

$$\widehat{p}(\boldsymbol{x}) = \frac{1}{n(2\pi\sigma^2)^{d/2}} \sum_{i=1}^n K_\sigma(\boldsymbol{x}, \boldsymbol{x}_i).$$

The performance of KDE depends on the choice of the kernel width σ. The kernel width σ can be optimized by *likelihood cross-validation* (LCV) as follows [18]: First, divide the samples $\{\boldsymbol{x}_i\}_{i=1}^n$ into k disjoint subsets $\{\mathcal{X}_j\}_{j=1}^k$. Then obtain a density estimate $\widehat{p}_{\mathcal{X}_\ell}(\boldsymbol{x})$ from $\{\mathcal{X}_j\}_{j\neq\ell}$ and compute its log-likelihood for $\mathcal{X}_\ell$:

$$\frac{1}{|\mathcal{X}_\ell|} \sum_{\boldsymbol{x}\in\mathcal{X}_\ell} \log \widehat{p}_{\mathcal{X}_\ell}(\boldsymbol{x}).$$

Repeat this procedure for $\ell = 1, 2, \ldots, k$ and choose the value of σ such that the average of the above hold-out log-likelihood over all ℓ is maximized. Note that the average hold-out log-likelihood is an almost unbiased estimate of the negative Kullback-Leibler divergence [36] from $p(\boldsymbol{x})$ to $\widehat{p}(\boldsymbol{x})$, up to an irrelevant constant.

KDE can be used for density ratio estimation by first obtaining density estimators $\widehat{q}(\boldsymbol{x})$ and $\widehat{q}'(\boldsymbol{x})$ separately from $\{\boldsymbol{x}_i\}_{i=1}^n$ and $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$, and then estimating the density ratio by plugging into the ratio as $\widehat{r}(\boldsymbol{x}) = \widehat{q}'(\boldsymbol{x})/\widehat{q}(\boldsymbol{x})$. However, a potential limitation of this naive approach is that KDE is not accurate in high-dimensional problems [1, 18].

2.3 Kernel Mean Matching

Kernel mean matching (KMM) allows us to directly obtain an estimate of the density-ratio values without going through density estimation [4]. The basic idea of KMM is to find $\widehat{r}(\boldsymbol{x})$ such that the mean discrepancy between nonlinearly transformed samples drawn from $q(\boldsymbol{x})$ and $q'(\boldsymbol{x})$ is minimized in a *universal reproducing kernel Hilbert space* [37]. The Gaussian kernel (1) is an example of kernels that induce a universal reproducing kernel Hilbert space and it has been shown [4] that the solution of the following optimization problem agrees with the true density ratio:

$$\min_{r(\boldsymbol{x})} \left\| \int K_\sigma(\boldsymbol{x},\cdot) q'(\boldsymbol{x}) d\boldsymbol{x} - \int K_\sigma(\boldsymbol{x},\cdot) r(\boldsymbol{x}) q(\boldsymbol{x}) d\boldsymbol{x} \right\|_{\mathcal{H}}^2$$
$$\text{subject to } \int r(\boldsymbol{x}) q(\boldsymbol{x}) d\boldsymbol{x} = 1 \text{ and } r(\boldsymbol{x}) \geq 0,$$

where $\|\cdot\|_{\mathcal{H}}$ denotes the norm in the Gaussian reproducing kernel Hilbert space and $K_\sigma(\boldsymbol{x},\boldsymbol{x}')$ is the Gaussian kernel (1).

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

$$\min_{\{r_i\}_{i=1}^n} \left[\frac{1}{2} \sum_{i,i'=1}^n r_i r_{i'} K_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_{i'}) - \sum_{i=1}^n r_i \kappa_i \right]$$
$$\text{subject to } \left| \frac{1}{n} \sum_{i=1}^n r_i - 1 \right| \leq \epsilon \text{ and } 0 \leq r_1, r_2, \ldots, r_n \leq B,$$

where

$$\kappa_i = \frac{n}{n'} \sum_{j=1}^{n'} K_\sigma(\boldsymbol{x}_i, \boldsymbol{x}'_j).$$

B (≥ 0) and ϵ (≥ 0) are tuning parameters that control the regularization effects. The solution $\{\widehat{r}_i\}_{i=1}^n$ is an estimate of the density ratio at $\{\boldsymbol{x}_i\}_{i=1}^n$.

Since KMM does not involve density estimation, it is expected to work well even in high-dimensional cases. However, the performance depends on the choice of the tuning parameters B, ϵ, and σ, and they cannot be simply optimized, e.g., by cross-validation (CV) since estimates of the density ratio are available only at $\{\boldsymbol{x}_i\}_{i=1}^n$. A popular heuristic is to use the median distance between samples as the Gaussian width σ [38, 33], although there seems no strong justification for this heuristic. For the choice of ϵ, a theoretical result given in the reference [4] could be used as guidance. However, it is still hard to determine the best value of ϵ in practice.

2.4 Logistic Regression

Another approach to directly estimating the density ratio is to use a probabilistic classifier. Let us assign a selector variable $\eta = -1$ to samples drawn from $q(\boldsymbol{x})$ and $\eta = 1$ to samples

drawn from $q'(\boldsymbol{x})$, i.e., the two densities are written as

$$q(\boldsymbol{x}) = p(\boldsymbol{x}|\eta = -1),$$
$$q'(\boldsymbol{x}) = p(\boldsymbol{x}|\eta = 1).$$

Note that η is regarded as a random variable.

An application of the Bayes theorem yields that the density ratio can be expressed in terms of η as follows [19, 20, 6]:

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

The ratio $p(\eta = -1)/p(\eta = 1)$ may be simply approximated by the ratio of the numbers of samples:

$$\frac{p(\eta = -1)}{p(\eta = 1)} \approx \frac{n}{n'}.$$

The conditional probability $p(\eta|\boldsymbol{x})$ could be approximated by discriminating $\{\boldsymbol{x}_i\}_{i=1}^n$ from $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$ using a *logistic regression* (LogReg) classifier, where η plays the role of a class variable. Below we briefly explain the LogReg method.

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

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

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:

$$\widehat{\boldsymbol{\zeta}} = \mathop{\mathrm{argmin}}_{\boldsymbol{\zeta}} \left[\sum_{i=1}^n \log\left(1 + \exp\left(\sum_{\ell=1}^m \zeta_\ell \phi_\ell(\boldsymbol{x}_i)\right)\right) + \sum_{j=1}^{n'} \log\left(1 + \exp\left(-\sum_{\ell=1}^m \zeta_\ell \phi_\ell(\boldsymbol{x}'_j)\right)\right) + \lambda \boldsymbol{\zeta}^\top \boldsymbol{\zeta} \right].$$

Since the above objective function is convex, the global optimal solution can be obtained by a standard nonlinear optimization technique such as the gradient method or Newton's method [39]. Then a density-ratio estimator is given by

$$\widehat{r}(\boldsymbol{x}) = \frac{n}{n'} \exp\left(\sum_{\ell=1}^m \widehat{\zeta}_\ell \phi_\ell(\boldsymbol{x})\right). \tag{2}$$

An advantage of the LogReg method is that model selection (i.e., the choice of the 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.

When multi-class logistic-regression classifiers are used, density ratios among multiple densities can be estimated simultaneously [40].

2.5 Kullback-Leibler Importance Estimation Procedure

The *Kullback-Leibler importance estimation procedure* (KLIEP) [9] also directly gives an estimator of the density-ratio function without going through density estimation by matching the two distributions in terms of the Kullback-Leibler divergence [36].

Let us model the density ratio $r(\boldsymbol{x})$ by the following linear model:

$$\widehat{r}(\boldsymbol{x}) = \sum_{\ell=1}^{b} \alpha_\ell \varphi_\ell(\boldsymbol{x}), \tag{3}$$

where $\{\alpha_\ell\}_{\ell=1}^b$ are parameters to be learned from data samples and $\{\varphi_\ell(\boldsymbol{x})\}_{\ell=1}^b$ are basis functions such that

$$\varphi_\ell(\boldsymbol{x}) \geq 0 \text{ for all } \boldsymbol{x} \in \mathcal{D} \text{ and for } \ell = 1, 2, \ldots, b.$$

Note that b and $\{\varphi_\ell(\boldsymbol{x})\}_{\ell=1}^b$ could be dependent on the samples $\{\boldsymbol{x}_i\}_{i=1}^n$ and $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$, so kernel models are also allowed. We explain how the basis functions $\{\varphi_\ell(\boldsymbol{x})\}_{\ell=1}^b$ are designed later.

An estimator of the density $q'(\boldsymbol{x})$ is given by using the density-ratio model $\widehat{r}(\boldsymbol{x})$ as

$$\widehat{q}'(\boldsymbol{x}) = \widehat{r}(\boldsymbol{x}) q(\boldsymbol{x}).$$

In KLIEP, the parameters $\{\alpha_\ell\}_{\ell=1}^b$ are determined so that the Kullback-Leibler divergence from $q'(\boldsymbol{x})$ to $\widehat{q}'(\boldsymbol{x})$ is minimized:

$$\begin{aligned} \mathrm{KL}[q'(\boldsymbol{x}) \| \widehat{q}'(\boldsymbol{x})] &= \int_{\mathcal{D}} q'(\boldsymbol{x}) \log \frac{q'(\boldsymbol{x})}{\widehat{r}(\boldsymbol{x}) q(\boldsymbol{x})} d\boldsymbol{x} \\ &= \int_{\mathcal{D}} q'(\boldsymbol{x}) \log \frac{q'(\boldsymbol{x})}{q(\boldsymbol{x})} d\boldsymbol{x} - \int_{\mathcal{D}} q'(\boldsymbol{x}) \log \widehat{r}(\boldsymbol{x}) d\boldsymbol{x}. \end{aligned} \tag{4}$$

The first term is a constant, so it can be safely ignored. Since $\widehat{q}'(\boldsymbol{x})$ is a probability density function, it should satisfy

$$1 = \int_{\mathcal{D}} \widehat{q}'(\boldsymbol{x}) d\boldsymbol{x} = \int_{\mathcal{D}} \widehat{r}(\boldsymbol{x}) q(\boldsymbol{x}) d\boldsymbol{x}. \tag{5}$$

The KLIEP optimization problem is given by replacing the expectations in Eqs.(4) and (5) with empirical averages:

$$\max_{\{\alpha_\ell\}_{\ell=1}^b} \left[\sum_{j=1}^{n'} \log \left(\sum_{\ell=1}^{b} \alpha_\ell \varphi_\ell(\boldsymbol{x}'_j) \right) \right]$$
$$\text{subject to } \frac{1}{n} \sum_{\ell=1}^{b} \alpha_\ell \sum_{i=1}^{n} \varphi_\ell(\boldsymbol{x}_i) = 1 \text{ and } \alpha_1, \alpha_2, \ldots, \alpha_b \geq 0.$$

Input: $m = \{\varphi_\ell(\boldsymbol{x})\}_{\ell=1}^b$, $\{\boldsymbol{x}_i\}_{i=1}^n$, and $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$
Output: $\widehat{r}(\boldsymbol{x})$

$A_{j,\ell} \longleftarrow \varphi_\ell(\boldsymbol{x}'_j)$ for $j = 1, 2, \ldots, n'$ and $\ell = 1, 2, \ldots, b$;
$\xi_\ell \longleftarrow \frac{1}{n}\sum_{i=1}^n \varphi_\ell(\boldsymbol{x}_i)$ for $\ell = 1, 2, \ldots, b$;
Initialize $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_b)^\top$ $(> \mathbf{0}_b)$ and ε $(0 < \varepsilon \ll 1)$;
Repeat until convergence
    $\boldsymbol{\alpha} \longleftarrow \boldsymbol{\alpha} + \varepsilon \boldsymbol{A}^\top(\mathbf{1}_{n'}./\boldsymbol{A}\boldsymbol{\alpha})$; % Gradient ascent
    $\boldsymbol{\alpha} \longleftarrow \boldsymbol{\alpha} + (1 - \boldsymbol{\xi}^\top\boldsymbol{\alpha})\boldsymbol{\xi}/(\boldsymbol{\xi}^\top\boldsymbol{\xi})$; % Constraint satisfaction
    $\boldsymbol{\alpha} \longleftarrow \max(\mathbf{0}_b, \boldsymbol{\alpha})$; % Constraint satisfaction
    $\boldsymbol{\alpha} \longleftarrow \boldsymbol{\alpha}/(\boldsymbol{\xi}^\top\boldsymbol{\alpha})$; % Constraint satisfaction
end
$\widehat{r}(\boldsymbol{x}) \longleftarrow \sum_{\ell=1}^b \alpha_\ell \varphi_\ell(\boldsymbol{x})$;

Figure 1: Pseudo code of KLIEP. $\mathbf{0}_b$ denotes the b-dimensional vector with all zeros, and $\mathbf{1}_{n'}$ denotes the n'-dimensional vector with all ones. '$./$' indicates the element-wise division and $^\top$ denotes the transpose. Inequalities and the 'max' operation for vectors are applied in the element-wise manner.

Input: $\mathcal{M} = \{m \mid m = \{\varphi_\ell(\boldsymbol{x})\}_{\ell=1}^b\}$, $\{\boldsymbol{x}_i\}_{i=1}^n$, and $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$
Output: $\widehat{r}(\boldsymbol{x})$

Split $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$ into k disjoint subsets $\{\mathcal{X}'_j\}_{j=1}^k$;
for each model $m \in \mathcal{M}$
    for each split $t = 1, 2, \ldots, k$
        $\widehat{r}_t(\boldsymbol{x}) \longleftarrow \mathrm{KLIEP}(m, \{\boldsymbol{x}_i\}_{i=1}^n, \{\mathcal{X}'_j\}_{j\neq t})$;
        $\widehat{L}_t(m) \longleftarrow \frac{1}{|\mathcal{X}'_t|}\sum_{\boldsymbol{x}\in\mathcal{X}'_t} \log \widehat{r}_t(\boldsymbol{x})$;
    end
    $\widehat{L}(m) \longleftarrow \frac{1}{k}\sum_{t=1}^k \widehat{L}_t(m)$;
end
$\widehat{m} \longleftarrow \mathrm{argmax}_{m\in\mathcal{M}} \widehat{L}(m)$;
$\widehat{r}(\boldsymbol{x}) \longleftarrow \mathrm{KLIEP}(\widehat{m}, \{\boldsymbol{x}_i\}_{i=1}^n, \{\boldsymbol{x}'_j\}_{j=1}^{n'})$;

Figure 2: Pseudo code of likelihood cross-validation for KLIEP.

This is a convex optimization problem and the global solution—which tends to be sparse [41]—can be obtained, e.g., by simply performing gradient ascent and feasibility satisfaction iteratively. A pseudo code is summarized in Figure 1.

The performance of KLIEP depends on the choice of the basis functions $\{\varphi_\ell(\boldsymbol{x})\}_{\ell=1}^b$. As explained below, the use of Gaussian basis functions would be reasonable:

$$\widehat{r}(\boldsymbol{x}) = \sum_{\ell=1}^{n'} \alpha_\ell K_\sigma(\boldsymbol{x}, \boldsymbol{x}'_\ell),$$

where $K_\sigma(\boldsymbol{x}, \boldsymbol{x}')$ is the Gaussian kernel with kernel width σ (see Eq.(1)). By definition, the density ratio $r(\boldsymbol{x})$ tends to take large values if $q(\boldsymbol{x})$ is small and $q'(\boldsymbol{x})$ is large; conversely, $r(\boldsymbol{x})$ tends to be small (i.e., close to zero) if $q(\boldsymbol{x})$ is large and $q'(\boldsymbol{x})$ is small. 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, many kernels are allocated in the region where $q'(\boldsymbol{x})$ has large values, which can be achieved by setting the Gaussian centers at $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$.

Model selection of KLIEP is possible based on a variant of likelihood cross-validation. A pseudo code is summarized in Figure 2. A MATLAB® implementation of the entire KLIEP algorithm is available from

`http://sugiyama-www.cs.titech.ac.jp/~sugi/software/KLIEP/`

Properties of KLIEP-type algorithms have been theoretically investigated in the references [42, 19, 20, 10]. Note that the density-ratio model of KLIEP is the linear model (3), while that of LogReg is the log-linear model (2). A variant of KLIEP for log-linear models has been studied in the reference [15].

2.6 Least-squares Importance Fitting

KLIEP employed the Kullback-Leibler divergence for measuring the discrepancy between two densities. *Least-squares importance fitting* (LSIF) [11] uses the squared loss for density-ratio function fitting. The density ratio $r(\boldsymbol{x})$ is again modeled by the linear model (3).

The parameters $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_b)^\top$ in the model $\widehat{r}(\boldsymbol{x})$ are determined so that the following squared error J_0 is minimized:

$$\begin{aligned} J_0(\boldsymbol{\alpha}) &= \frac{1}{2}\int \left(\widehat{r}(\boldsymbol{x}) - r(\boldsymbol{x})\right)^2 q(\boldsymbol{x}) d\boldsymbol{x} \\ &= \frac{1}{2}\int \widehat{r}(\boldsymbol{x})^2 q(\boldsymbol{x}) d\boldsymbol{x} - \int \widehat{r}(\boldsymbol{x}) q'(\boldsymbol{x}) d\boldsymbol{x} + \frac{1}{2}\int r(\boldsymbol{x}) q'(\boldsymbol{x}) d\boldsymbol{x}, \end{aligned}$$

where the last term is a constant and therefore can be safely ignored. Let us denote the first two terms by J:

$$J(\boldsymbol{\alpha}) = \frac{1}{2}\int \widehat{r}(\boldsymbol{x})^2 q(\boldsymbol{x}) d\boldsymbol{x} - \int \widehat{r}(\boldsymbol{x}) q'(\boldsymbol{x}) d\boldsymbol{x}.$$

Approximating the expectations in J by empirical averages, we obtain

$$\begin{aligned}\widehat{J}(\boldsymbol{\alpha}) &= \frac{1}{2n}\sum_{i=1}^{n}\widehat{r}(\boldsymbol{x}_i)^2 - \frac{1}{n'}\sum_{j=1}^{n'}\widehat{r}(\boldsymbol{x}'_j) \\ &= \frac{1}{2}\sum_{\ell,\ell'=1}^{b}\alpha_\ell\alpha_{\ell'}\widehat{H}_{\ell,\ell'} - \sum_{\ell=1}^{b}\alpha_\ell\widehat{h}_\ell, \end{aligned} \tag{6}$$

where

$$\widehat{H}_{\ell,\ell'} := \frac{1}{n}\sum_{i=1}^{n}\varphi_\ell(\boldsymbol{x}_i)\varphi_{\ell'}(\boldsymbol{x}_i), \tag{7}$$

$$\widehat{h}_\ell := \frac{1}{n'}\sum_{j=1}^{n'}\varphi_\ell(\boldsymbol{x}'_j). \tag{8}$$

Taking into account the non-negativity of the density-ratio function $r(\boldsymbol{x})$, the optimization problem is formulated as follows.

$$\begin{aligned}&\min_{\{\alpha_\ell\}_{\ell=1}^{b}}\left[\frac{1}{2}\sum_{\ell,\ell'=1}^{b}\alpha_\ell\alpha_{\ell'}\widehat{H}_{\ell,\ell'} - \sum_{\ell=1}^{b}\alpha_\ell\widehat{h}_\ell + \lambda\sum_{\ell=1}^{b}\alpha_\ell\right] \\ &\qquad \text{subject to } \alpha_\ell \geq 0 \quad \text{for } \ell = 1,2,\ldots,b,\end{aligned} \tag{9}$$

where a penalty term $\lambda\sum_{\ell=1}^{b}\alpha_\ell$ is included for regularization purposes with λ (≥ 0) being a regularization parameter. Eq.(9) is a convex quadratic programming problem and therefore the unique global optimal solution can be computed efficiently by a standard optimization package. Basis functions may be designed in the same way as KLIEP (i.e., Gaussian basis functions; see Section 2.5).

Model selection of the Gaussian width σ and the regularization parameter λ is possible by a variant of cross-validation: First, $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$ are divided into k disjoint subsets $\{\mathcal{X}_i\}_{i=1}^{k}$ and $\{\mathcal{X}'_j\}_{j=1}^{k}$, respectively. Then a density-ratio estimate $\widehat{r}_\ell(\boldsymbol{x})$ is obtained using $\{\mathcal{X}_i\}_{i\neq\ell}$ and $\{\mathcal{X}'_j\}_{j\neq\ell}$, and the cost J is approximated using the hold-out samples $\mathcal{X}_\ell$ and $\mathcal{X}'_\ell$ as

$$\widehat{J}_\ell := \frac{1}{2|\mathcal{X}_\ell|}\sum_{\boldsymbol{x}\in\mathcal{X}_\ell}\widehat{r}_\ell(\boldsymbol{x})^2 - \frac{1}{|\mathcal{X}'_\ell|}\sum_{\boldsymbol{x}'\in\mathcal{X}'_\ell}\widehat{r}_\ell(\boldsymbol{x}').$$

This procedure is repeated for $\ell = 1,2,\ldots,k$ and its average $\widehat{J}$ is used as an estimate of J:

$$\widehat{J} := \frac{1}{k}\sum_{\ell=1}^{k}\widehat{J}_\ell.$$

The LSIF solution $\widehat{\boldsymbol{\alpha}}$ is shown to be piecewise linear with respect to the regularization parameter λ [11]. Therefore, the *regularization path* (i.e., solutions for all λ) can be

Input: $\widehat{\boldsymbol{H}}$ and $\widehat{\boldsymbol{h}}$ % see Eqs.(7) and (8) for the definition
Output: entire regularization path $\widehat{\boldsymbol{\alpha}}(\lambda)$ for $\lambda \geq 0$

$\tau \longleftarrow 0$; $k \longleftarrow \mathrm{argmax}_i\{\widehat{h}_i \mid i = 1, 2, \ldots, b\}$;
$\lambda_\tau \longleftarrow \widehat{h}_k$; $\widehat{\mathcal{A}} \longleftarrow \{1, 2, \ldots, b\} \backslash \{k\}$;
$\widehat{\boldsymbol{\alpha}}(\lambda_\tau) \longleftarrow \mathbf{0}_b$; % the vector with all zeros
While $\lambda_\tau > 0$
  $\widehat{\boldsymbol{E}} \longleftarrow \boldsymbol{O}_{|\widehat{\mathcal{A}}| \times b}$; % the matrix with all zeros
  For $i = 1, 2, \ldots, |\widehat{\mathcal{A}}|$
    $\widehat{E}_{i,j_i} \longleftarrow 1$; % $\widehat{\mathcal{A}} = \{j_1, j_2, \ldots, j_{|\widehat{\mathcal{A}}|} \mid j_1 < j_2 < \cdots < j_{|\widehat{\mathcal{A}}|}\}$
  end
  $\widehat{\boldsymbol{G}} \longleftarrow \begin{pmatrix} \widehat{\boldsymbol{H}} & -\widehat{\boldsymbol{E}}^\top \\ -\widehat{\boldsymbol{E}} & \boldsymbol{O}_{|\widehat{\mathcal{A}}| \times |\widehat{\mathcal{A}}|} \end{pmatrix}$;
  $\boldsymbol{u} \longleftarrow \widehat{\boldsymbol{G}}^{-1} \begin{pmatrix} \widehat{\boldsymbol{h}} \\ \mathbf{0}_{|\widehat{\mathcal{A}}|} \end{pmatrix}$; $\boldsymbol{v} \longleftarrow \widehat{\boldsymbol{G}}^{-1} \begin{pmatrix} \mathbf{1}_b \\ \mathbf{0}_{|\widehat{\mathcal{A}}|} \end{pmatrix}$;
  If $\boldsymbol{v} \leq \mathbf{0}_{b+|\widehat{\mathcal{A}}|}$ % the final interval
    $\lambda_{\tau+1} \longleftarrow 0$; $\widehat{\boldsymbol{\alpha}}(\lambda_{\tau+1}) \longleftarrow (u_1, u_2, \ldots, u_b)^\top$;
  else % an intermediate interval
    $k \longleftarrow \mathrm{argmax}_i\{u_i/v_i \mid v_i > 0,\ i = 1, 2, \ldots, b + |\widehat{\mathcal{A}}|\}$;
    $\lambda_{\tau+1} \longleftarrow \max\{0, u_k/v_k\}$;
    $\widehat{\boldsymbol{\alpha}}(\lambda_{\tau+1}) \longleftarrow (u_1, u_2, \ldots, u_b)^\top - \lambda_{\tau+1}(v_1, v_2, \ldots, v_b)^\top$;
    If $1 \leq k \leq b$
      $\widehat{\mathcal{A}} \longleftarrow \widehat{\mathcal{A}} \cup \{k\}$;
    else
      $\widehat{\mathcal{A}} \longleftarrow \widehat{\mathcal{A}} \backslash \{j_{k-b}\}$;
    end
  end
  $\tau \longleftarrow \tau + 1$;
end
$$\widehat{\boldsymbol{\alpha}}(\lambda) \longleftarrow \begin{cases} \mathbf{0}_b & \text{if } \lambda \geq \lambda_0 \\ \frac{\lambda_{\tau+1} - \lambda}{\lambda_{\tau+1} - \lambda_\tau} \widehat{\boldsymbol{\alpha}}(\lambda_\tau) + \frac{\lambda - \lambda_\tau}{\lambda_{\tau+1} - \lambda_\tau} \widehat{\boldsymbol{\alpha}}(\lambda_{\tau+1}) & \text{if } \lambda_{\tau+1} \leq \lambda \leq \lambda_\tau \end{cases}$$

Figure 3: Pseudo code for computing the entire regularization path of LSIF. The computation of $\widehat{\boldsymbol{G}}^{-1}$ is sometimes unstable. For stabilization purposes, small positive diagonals may be added to $\widehat{\boldsymbol{H}}$.

computed efficiently based on the *parametric optimization technique* [43, 44, 45]. A pseudo code of the regularization path tracking algorithm for LSIF is described in Figure 3. This implies that a quadratic programming solver is no longer needed for obtaining the LSIF solution—just computing matrix inverses is enough. This highly contributes to saving the computation time. Furthermore, the regularization path algorithm is computationally very efficient when the solution is sparse, i.e., most of the elements are zero since the number of change points tends to be small for sparse solutions.

An R implementation of the entire LSIF algorithm is available from

`http://www.math.cm.is.nagoya-u.ac.jp/~kanamori/software/LSIF/`

2.7 Unconstrained Least-squares Importance Fitting

LSIF combined with regularization path tracking is computationally very efficient. However, it sometimes suffers from a numerical problem and therefore is not practically reliable. To cope with this problem, an approximation method called *unconstrained LSIF* (uLSIF) has been introduced [11].

The approximation idea is very simple: the non-negativity constraint in the optimization problem (9) is dropped. This results in the following unconstrained optimization problem.

$$\min_{\{\alpha_\ell\}_{\ell=1}^b} \left[\frac{1}{2} \sum_{\ell,\ell'=1}^{b} \alpha_\ell \alpha_{\ell'} \widehat{H}_{\ell,\ell'} - \sum_{\ell=1}^{b} \alpha_\ell \widehat{h}_\ell + \frac{\lambda}{2} \sum_{\ell=1}^{b} \alpha_\ell^2 \right]. \tag{10}$$

In the above, a quadratic regularization term is included $\lambda \sum_{\ell=1}^b \alpha_\ell^2/2$ instead of the linear one since the linear penalty term does not work as a regularizer without the non-negativity constraint. Eq.(10) is an unconstrained convex quadratic program, so the solution can be analytically computed as

$$\widetilde{\boldsymbol{\alpha}} = (\widetilde{\alpha}_1, \widetilde{\alpha}_2, \ldots, \widetilde{\alpha}_b)^\top = (\widehat{\boldsymbol{H}} + \lambda \boldsymbol{I}_b)^{-1} \widehat{\boldsymbol{h}},$$

where $\boldsymbol{I}_b$ is the b-dimensional identity matrix. Since the non-negativity constraint $\alpha_\ell \geq 0$ is dropped, some of the learned parameters could be negative. To compensate for this approximation error (see the reference [46] for theoretical error analysis), the solution is modified as

$$\widehat{\alpha}_\ell = \max(0, \widetilde{\alpha}_\ell) \quad \text{for } \ell = 1, 2, \ldots, b. \tag{11}$$

An advantage of the above unconstrained formulation is that the solution can be computed just by solving a system of linear equations. Therefore, the computation is fast and stable. See the reference [47] for theoretical analysis of algorithmic stability.

Another, and more significant advantage of uLSIF is that the score of leave-one-out cross-validation (LOOCV) can be computed analytically—thanks to this property, the computational complexity for performing leave-one-out cross-validation is the same order as just computing a single solution, which is explained below. In the current setting,

two sets of samples $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$ are given, which generally have different sample size. For explaining the idea in a simple manner, we assume that $n < n'$ and $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$ $(i = 1, 2, \ldots, n)$ are held out at the same time; $\{\boldsymbol{x}'_j\}_{j=n+1}^{n'}$ are always used for density-ratio estimation.

Let $\widehat{r}^{(i)}(\boldsymbol{x})$ be an estimate of the density ratio obtained without $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$. Then the leave-one-out cross-validation score is expressed as

$$\mathrm{LOOCV} = \frac{1}{n}\sum_{i=1}^{n}\left[\frac{1}{2}(\widehat{r}^{(i)}(\boldsymbol{x}_i))^2 - \widehat{r}^{(i)}(\boldsymbol{x}'_i)\right]. \tag{12}$$

Our approach to efficiently computing the leave-one-out cross-validation score is to use the *Sherman-Woodbury-Morrison* formula [48] for computing matrix inverses. A pseudo code of uLSIF with LOOCV-based model selection is summarized in Figure 4. MATLAB® and R implementations of the entire uLSIF algorithm are available from

`http://sugiyama-www.cs.titech.ac.jp/~sugi/software/uLSIF/`
`http://www.math.cm.is.nagoya-u.ac.jp/~kanamori/software/LSIF/`

2.8 Discussions

Table 1 summarizes properties of the density-ratio estimation methods.

KDE is efficient in computation since no optimization is involved, and model selection is possible by likelihood cross-validation. However, KDE is not accurate in high-dimensional problems [1, 18].

KMM may potentially overcome the weakness of KDE by directly estimating the density ratio. However, there is no objective model selection method. Therefore, model parameters such as the Gaussian width need to be determined by hand, which is highly unreliable unless we have strong prior knowledge. Furthermore, the computation of KMM is rather demanding since a quadratic programming problem has to be solved.

LogReg and KLIEP also do not involve density estimation, but different from KMM, they give estimators the entire density-ratio function, not only the values of the density-ratio function at samples. Therefore, the values of the density ratio at unseen points can be estimated by LogReg and KLIEP. This feature is highly useful since it enables us to employ cross-validation for model selection, which is a significant advantage over KMM. However, LogReg and KLIEP are computationally rather expensive since non-linear optimization problems have to be solved.

LSIF is qualitatively similar to LogReg and KLIEP, i.e., it can avoid density estimation, model selection is possible, and non-linear optimization is involved. LSIF is advantageous over LogReg and KLIEP in that it is equipped with a regularization path tracking algorithm. Thanks to this, model selection of LSIF is computationally much more efficient than LogReg and KLIEP. However, the regularization path tracking algorithm tends to be numerically unstable.

Input: $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$
Output: $\widehat{r}(\boldsymbol{x})$

$b \longleftarrow \min(100, n')$; $\quad \overline{n} \longleftarrow \min(n, n')$;
Randomly choose b centers $\{\boldsymbol{c}_\ell\}_{\ell=1}^{b}$ from $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$ without replacement;
For each candidate of Gaussian width σ

$$\widehat{H}_{\ell,\ell'} \longleftarrow \frac{1}{n}\sum_{i=1}^{n}\exp\left(-\frac{||\boldsymbol{x}_i-\boldsymbol{c}_\ell||^2+||\boldsymbol{x}_i-\boldsymbol{c}_{\ell'}||^2}{2\sigma^2}\right) \text{ for } \ell,\ell'=1,2,\ldots,b;$$

$$\widehat{h}_\ell \longleftarrow \frac{1}{n'}\sum_{j=1}^{n'}\exp\left(-\frac{||\boldsymbol{x}'_j-\boldsymbol{c}_\ell||^2}{2\sigma^2}\right) \text{ for } \ell=1,2,\ldots,b;$$

$$X_{\ell,i} \longleftarrow \exp\left(-\frac{\|\boldsymbol{x}_i-\boldsymbol{c}_\ell\|^2}{2\sigma^2}\right) \text{ for } i=1,2,\ldots,\overline{n} \text{ and } \ell=1,2,\ldots,b;$$

$$X'_{\ell,i} \longleftarrow \exp\left(-\frac{\|\boldsymbol{x}'_i-\boldsymbol{c}_\ell\|^2}{2\sigma^2}\right) \text{ for } i=1,2,\ldots,\overline{n} \text{ and } \ell=1,2,\ldots,b;$$

For each candidate of regularization parameter λ

$$\widehat{\boldsymbol{B}} \longleftarrow \widehat{\boldsymbol{H}} + \frac{\lambda(n-1)}{n}\boldsymbol{I}_b;$$

$$\boldsymbol{B}_0 \longleftarrow \widehat{\boldsymbol{B}}^{-1}\widehat{\boldsymbol{h}}\boldsymbol{1}_{\overline{n}}^\top + \widehat{\boldsymbol{B}}^{-1}\boldsymbol{X}\,\mathrm{diag}\left(\frac{\widehat{\boldsymbol{h}}^\top\widehat{\boldsymbol{B}}^{-1}\boldsymbol{X}}{n\boldsymbol{1}_{\overline{n}}^\top - \boldsymbol{1}_b^\top(\boldsymbol{X}*\widehat{\boldsymbol{B}}^{-1}\boldsymbol{X})}\right);$$

$$\boldsymbol{B}_1 \longleftarrow \widehat{\boldsymbol{B}}^{-1}\boldsymbol{X}' + \widehat{\boldsymbol{B}}^{-1}\boldsymbol{X}\,\mathrm{diag}\left(\frac{\boldsymbol{1}_b^\top(\boldsymbol{X}'*\widehat{\boldsymbol{B}}^{-1}\boldsymbol{X})}{n\boldsymbol{1}_{\overline{n}}^\top - \boldsymbol{1}_b^\top(\boldsymbol{X}*\widehat{\boldsymbol{B}}^{-1}\boldsymbol{X})}\right);$$

$$\boldsymbol{B}_2 \longleftarrow \max\left(\boldsymbol{O}_{b\times\overline{n}}, \frac{n-1}{n(n'-1)}(n'\boldsymbol{B}_0-\boldsymbol{B}_1)\right);$$

$$\boldsymbol{r} \longleftarrow (\boldsymbol{1}_b^\top(\boldsymbol{X}*\boldsymbol{B}_2))^\top; \quad \boldsymbol{r}' \longleftarrow (\boldsymbol{1}_b^\top(\boldsymbol{X}'*\boldsymbol{B}_2))^\top;$$

$$\mathrm{LOOCV}(\sigma,\lambda) \longleftarrow \frac{\boldsymbol{r}^\top\boldsymbol{r}}{2\overline{n}} - \frac{\boldsymbol{1}_{\overline{n}}^\top\boldsymbol{r}'}{\overline{n}};$$

end
end

$(\widehat{\sigma},\widehat{\lambda}) \longleftarrow \mathrm{argmin}_{(\sigma,\lambda)}\,\mathrm{LOOCV}(\sigma,\lambda)$;

$$\widetilde{H}_{\ell,\ell'} \longleftarrow \frac{1}{n}\sum_{i=1}^{n}\exp\left(-\frac{||\boldsymbol{x}_i-\boldsymbol{c}_\ell||^2+||\boldsymbol{x}_i-\boldsymbol{c}_{\ell'}||^2}{2\widehat{\sigma}^2}\right) \text{ for } \ell,\ell'=1,2,\ldots,b;$$

$$\widetilde{h}_\ell \longleftarrow \frac{1}{n'}\sum_{j=1}^{n'}\exp\left(-\frac{||\boldsymbol{x}'_j-\boldsymbol{c}_\ell||^2}{2\widehat{\sigma}^2}\right) \text{ for } \ell=1,2,\ldots,b;$$

$$\widehat{\boldsymbol{\alpha}} \longleftarrow \max(\boldsymbol{0}_b, (\widetilde{\boldsymbol{H}}+\widehat{\lambda}\boldsymbol{I}_b)^{-1}\widetilde{\boldsymbol{h}});$$

$$\widehat{r}(\boldsymbol{x}) \longleftarrow \sum_{\ell=1}^{b}\widehat{\alpha}_\ell\exp\left(-\frac{||\boldsymbol{x}-\boldsymbol{c}_\ell||^2}{2\widehat{\sigma}^2}\right);$$

Figure 4: Pseudo code of uLSIF with leave-one-out cross-validation. $\boldsymbol{B}*\boldsymbol{B}'$ denotes the element-wise multiplication of matrices $\boldsymbol{B}$ and $\boldsymbol{B}'$ of the same size. For n-dimensional vectors $\boldsymbol{b}$ and $\boldsymbol{b}'$, $\mathrm{diag}\left(\frac{\boldsymbol{b}}{\boldsymbol{b}'}\right)$ denotes the $n\times n$ diagonal matrix with the i-th diagonal element b_i/b'_i.

Table 1: Density-ratio estimation methods.

Methods	Density estimation	Model selection	Optimization	Out-of-sample prediction
KDE	Necessary	**Available**	**Analytic**	**Possible**
KMM	**Not necessary**	Not available	Convex quadratic program	Not possible
LogReg	**Not necessary**	**Available**	Convex non-linear	**Possible**
KLIEP	**Not necessary**	**Available**	Convex non-linear	**Possible**
LSIF	**Not necessary**	**Available**	Convex quadratic program	**Possible**
uLSIF	**Not necessary**	**Available**	**Analytic**	**Possible**

uLSIF inherits good properties of other methods, e.g., no density estimation is involved and a built-in model selection method is available. In addition to these preferable properties, the solution of uLSIF can be computed analytically through matrix inversion and therefore uLSIF is computationally very efficient and numerically stable. Furthermore, thanks to the availability of the closed-form solution of uLSIF, the leave-one-out cross-validation score can be analytically computed without repeating hold-out loops, which highly contributes to reducing the computation time in the model selection phase.

As experimentally demonstrated in Section 4, KLIEP, LSIF, and uLSIF are shown to be accurate and uLSIF is computationally advantageous. Thus the use of uLSIF may be recommended for practical use.

3 Statistical Data Processing via Density-ratio Estimation

In this section, we show how the density ratio estimation methods could be employed for solving various statistical data processing tasks.

3.1 Covariate Shift Adaptation

Covariate shift [22] is a situation in supervised learning where the input distributions change between the training and test phases but the conditional distribution of outputs given inputs remains unchanged. Under covariate shift, standard learning techniques such as maximum likelihood estimation are biased; the bias caused by covariate shift can be asymptotically canceled by weighting the loss function according to the importance [22, 2, 3, 5]. The basic idea of covariate shift adaptation is summarized in the following importance sampling identity:

$$\begin{aligned}\mathop{\mathbb{E}}_{q'(\boldsymbol{x})}[g(\boldsymbol{x})] &= \int g(\boldsymbol{x})q'(\boldsymbol{x})d\boldsymbol{x} \\ &= \int g(\boldsymbol{x})r(\boldsymbol{x})q(\boldsymbol{x})d\boldsymbol{x} = \mathop{\mathbb{E}}_{q(\boldsymbol{x})}[g(\boldsymbol{x})r(\boldsymbol{x})].\end{aligned}$$

That is, the expectation of a function $g(\boldsymbol{x})$ over $q'(\boldsymbol{x})$ can be computed by the importance-weighted expectation over $q(\boldsymbol{x})$. Similarly, standard model selection criteria such as cross-validation or Akaike's information criterion lose their unbiasedness due to covariate shift; proper unbiasedness can be recovered by modifying the methods based on importance weighting [22, 2, 3, 5, 4, 14]. Furthermore, the performance of active learning or the experiment design, i.e., the training input distribution is designed by the user to enhance the generalization performance, could also be improved by the use of the importance. [49, 50, 51, 52].

Thus the importance plays a central role in covariate shift adaptation and density-ratio estimation methods could be used for reducing the estimation bias under covariate shift. Examples of successful real-world applications includes brain-computer interface [5], robot control [53], speaker identification [54], and natural language processing [15]. A similar importance-weighting idea also plays a central role in domain adaptation [55] and multi-task learning [40].

A more practical explanation of covariate shift adaptation techniques are summarized in Section 4.2 with experimental results.

3.2 Inlier-based Outlier Detection

Let us consider an outlier detection problem [23, 24] of finding irregular samples in a dataset ("evaluation dataset") based on another dataset ("model dataset") that only contains regular samples. Defining the density ratio over two sets of samples, we can see that the density-ratio values for regular samples are close to one, while those for outliers tend to be significantly deviated from one. Thus the density-ratio values could be used as an index of the degree of outlyingness [7]. Since the evaluation dataset has a wider support than the model dataset, we regard the evaluation dataset as samples corresponding to the denominator in the density ratio and the model dataset as samples corresponding to the numerator in the density ratio. Then outliers tend to have smaller density-ratio values (i.e., close to zero). As such, density-ratio estimation methods could be employed for outlier detection. A similar idea could be used for change-point detection in time-series [56, 57] and two-sample problems in hypothesis testing [58].

Practical advantages of the density-ratio approach in outlier detection scenarios are experimentally investigated in Section 4.3.

3.3 Conditional Density Estimation

Suppose we are given n i.i.d. paired samples $\{(\boldsymbol{x}_k, \boldsymbol{y}_k)\}_{k=1}^n$ drawn from a joint distribution with density $p(\boldsymbol{x}, \boldsymbol{y})$. The goal is to estimate the conditional density $p(\boldsymbol{y}|\boldsymbol{x})$. When the domain of $\boldsymbol{x}$ is continuous, conditional density estimation is not straightforward since a naive empirical approximation cannot be used [26, 59].

In the context of density-ratio estimation, let us regard $\{(\boldsymbol{x}_k, \boldsymbol{y}_k)\}_{k=1}^n$ as samples corresponding to the numerator of the density ratio and $\{\boldsymbol{x}_k\}_{k=1}^n$ as samples corresponding

to the denominator of the density ratio, i.e., we consider the density ratio defined by

$$r(\boldsymbol{x},\boldsymbol{y}) := \frac{p(\boldsymbol{x},\boldsymbol{y})}{p(\boldsymbol{x})},$$

which is equivalent to the conditional density $p(\boldsymbol{y}|\boldsymbol{x})$. Thus a density-ratio estimation method directly gives an estimate of the conditional density. Using a density-ratio estimation method, we can estimate the conditional density as

$$\widehat{p}(\boldsymbol{y}|\boldsymbol{x}) = \widehat{r}(\boldsymbol{x},\boldsymbol{y}).$$

Below, we explain a more technical detail by taking the uLSIF idea as an example. Similar formulations are also possible for LSIF and KLIEP. However, KMM and LogReg may not be used for conditional density estimation since these methods explicitly require that the domains of the denominator and the numerator of the density ratio are common, which is not satisfied here.

For the density-ratio model

$$\widehat{r}(\boldsymbol{x},\boldsymbol{y}) = \sum_{\ell=1}^{b} \alpha_\ell \varphi_\ell(\boldsymbol{x},\boldsymbol{y}),$$

let us consider the following squared fitting error.

$$\begin{aligned}
&\frac{1}{2}\iint \left(\widehat{r}(\boldsymbol{x},\boldsymbol{y}) - r(\boldsymbol{x},\boldsymbol{y})\right)^2 p(\boldsymbol{x}) d\boldsymbol{x} d\boldsymbol{y} \\
&= \frac{1}{2}\iint \left(\sum_{\ell=1}^{b} \alpha_\ell \varphi_\ell(\boldsymbol{x},\boldsymbol{y})\right)^2 p(\boldsymbol{x}) d\boldsymbol{x} d\boldsymbol{y} - \iint \sum_{\ell=1}^{b} \alpha_\ell \varphi_\ell(\boldsymbol{x},\boldsymbol{y}) p(\boldsymbol{x},\boldsymbol{y}) d\boldsymbol{x} d\boldsymbol{y} \\
&\quad + \frac{1}{2}\iint r(\boldsymbol{x},\boldsymbol{y}) p(\boldsymbol{x},\boldsymbol{y}) d\boldsymbol{x} d\boldsymbol{y},
\end{aligned}$$

where the last term is a constant. Ignoring the constant, approximating the integral by sample averages, and adding the ℓ_2-regularizer, we have the following optimization problem:

$$\min_{\{\alpha_\ell\}_{\ell=1}^{b}} \left[\frac{1}{2}\sum_{\ell,\ell'=1}^{b} \alpha_\ell \alpha_{\ell'} \widehat{H}_{\ell,\ell'} - \sum_{\ell=1}^{b} \alpha_\ell \widehat{h}_\ell + \frac{\lambda}{2}\sum_{\ell=1}^{b} \alpha_\ell^2\right],$$

where

$$\begin{aligned}
\widehat{H}_{\ell,\ell'} &:= \frac{1}{n}\sum_{k=1}^{n} \int \varphi_\ell(\boldsymbol{x}_k,\boldsymbol{y}) \varphi_{\ell'}(\boldsymbol{x}_k,\boldsymbol{y}) d\boldsymbol{y}, \\
\widehat{h}_\ell &:= \frac{1}{n}\sum_{k=1}^{n} \varphi_\ell(\boldsymbol{x}_k,\boldsymbol{y}_k).
\end{aligned}$$

This is basically the same formulation as uLSIF. For the Gaussian basis functions

$$\phi_\ell(\boldsymbol{x},\boldsymbol{y}) = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{u}_\ell\|^2}{2\sigma^2}\right)\exp\left(-\frac{\|\boldsymbol{y}-\boldsymbol{v}_\ell\|^2}{2\sigma^2}\right),$$

the integral in the definition of $\widehat{H}_{\ell,\ell'}$ above can be computed analytically as

$$\int \phi_\ell(\boldsymbol{x},\boldsymbol{y})\phi_{\ell'}(\boldsymbol{x},\boldsymbol{y})d\boldsymbol{y} = (\sqrt{\pi}\sigma)^{d_{\mathrm{Y}}}\exp\left(-\frac{2\|\boldsymbol{x}-\boldsymbol{u}_\ell\|^2+2\|\boldsymbol{x}-\boldsymbol{u}_{\ell'}\|^2+\|\boldsymbol{v}_\ell-\boldsymbol{v}_{\ell'}\|^2}{4\sigma^2}\right),$$

where d_{Y} denotes the dimension of $\boldsymbol{y}$.

For the KLIEP-type loss function, we may formulate the optimization criterion as

$$\max_{\{\alpha_\ell\}_{\ell=1}^b}\left[\sum_{k=1}^n \log\left(\sum_{\ell=1}^b \alpha_\ell\varphi_\ell(\boldsymbol{x}_k,\boldsymbol{y}_k)\right)\right]$$
$$\text{subject to } \frac{1}{n}\sum_{\ell=1}^b \alpha_\ell \sum_{k=1}^n \int \varphi_\ell(\boldsymbol{x}_k,\boldsymbol{y})d\boldsymbol{y} = 1 \quad \text{and} \quad \alpha_1,\alpha_2,\ldots,\alpha_b \geq 0.$$

Practical advantages of the density-ratio approach in conditional density estimation scenarios are experimentally investigated in Section 4.4. See the paper [16] for more detail.

3.4 Mutual Information Estimation

Suppose that we are given n i.i.d. paired samples $\{(\boldsymbol{x}_k,\boldsymbol{y}_k)\}_{k=1}^n$ drawn from a joint distribution with density $p(\boldsymbol{x},\boldsymbol{y})$. Let us denote the marginal densities of $\boldsymbol{x}_k$ and $\boldsymbol{y}_k$ by $p(\boldsymbol{x})$ and $p(\boldsymbol{y})$, respectively. Mutual information $I(X,Y)$ between random variables X and Y is defined by

$$I(X,Y) := \iint p(\boldsymbol{x},\boldsymbol{y})\log\frac{p(\boldsymbol{x},\boldsymbol{y})}{p(\boldsymbol{x})p(\boldsymbol{y})}d\boldsymbol{x}d\boldsymbol{y}, \tag{13}$$

which plays a central role in information theory [29]. A squared-loss variant of mutual information is defined by

$$I_s(X,Y) := \frac{1}{2}\int\left(\frac{p(\boldsymbol{x},\boldsymbol{y})}{p(\boldsymbol{x})p(\boldsymbol{y})}-1\right)^2 p(\boldsymbol{x})p(\boldsymbol{y})d\boldsymbol{x}d\boldsymbol{y}. \tag{14}$$

This corresponds to the *f-divergence* [60, 61] from $p(\boldsymbol{x},\boldsymbol{y})$ to $p(\boldsymbol{x})p(\boldsymbol{y})$ with the squared loss, while ordinary mutual information (13) corresponds to the f-divergence with the log loss (i.e., the *Kullback-Leibler divergence* [36]).

3.4.1 Estimating Mutual Information using Density-ratio Methods

Let us regard $\{(\boldsymbol{x}_k, \boldsymbol{y}_k)\}_{k=1}^n$ as samples corresponding to the numerator of the density ratio and $\{(\boldsymbol{x}_k, \boldsymbol{y}_{k'})\}_{k,k'=1}^n$ as samples corresponding to the denominator of the density ratio, i.e., we consider the density ratio defined by

$$r(\boldsymbol{x}, \boldsymbol{y}) := \frac{p(\boldsymbol{x}, \boldsymbol{y})}{p(\boldsymbol{x})p(\boldsymbol{y})}.$$

Then mutual information can be estimated using a density-ratio estimator $\widehat{r}(\boldsymbol{x}, \boldsymbol{y})$ as follows [8]:

$$\widehat{I}(X, Y) = \frac{1}{n}\sum_{k=1}^{n} \log \widehat{r}(\boldsymbol{x}_k, \boldsymbol{y}_k).$$

Below, we explain a more concrete formulation by taking the uLSIF idea as an example; see the papers [8, 13, 17] for more detail on mutual information estimation based on density-ratio methods. For the density-ratio model

$$\widehat{r}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\ell=1}^{b} \alpha_\ell \varphi_\ell(\boldsymbol{x}, \boldsymbol{y}),$$

let us consider the following squared fitting error.

$$\begin{aligned}
&\frac{1}{2}\iint \left(\widehat{r}(\boldsymbol{x}, \boldsymbol{y}) - r(\boldsymbol{x}, \boldsymbol{y})\right)^2 p(\boldsymbol{x})p(\boldsymbol{y})d\boldsymbol{x}d\boldsymbol{y} \\
&= \frac{1}{2}\iint \left(\sum_{\ell=1}^{b} \alpha_\ell \varphi_\ell(\boldsymbol{x}, \boldsymbol{y})\right)^2 p(\boldsymbol{x})p(\boldsymbol{y})d\boldsymbol{x}d\boldsymbol{y} - \iint \sum_{\ell=1}^{b} \alpha_\ell \varphi_\ell(\boldsymbol{x}, \boldsymbol{y}) p(\boldsymbol{x}, \boldsymbol{y})d\boldsymbol{x}d\boldsymbol{y} \\
&\quad + \frac{1}{2}\iint r(\boldsymbol{x}, \boldsymbol{y})p(\boldsymbol{x}, \boldsymbol{y})d\boldsymbol{x}d\boldsymbol{y},
\end{aligned}$$

where the last term is a constant. Ignoring the constant, approximating the integral by sample averages, and adding the ℓ_2-regularizer, we have the following optimization problem:

$$\min_{\{\alpha_\ell\}_{\ell=1}^b} \left[\frac{1}{2}\sum_{\ell,\ell'=1}^{b} \alpha_\ell \alpha_{\ell'} \widehat{H}_{\ell,\ell'} - \sum_{\ell=1}^{b} \alpha_\ell \widehat{h}_\ell + \frac{\lambda}{2}\sum_{\ell=1}^{b} \alpha_\ell^2\right], \tag{15}$$

where

$$\widehat{H}_{\ell,\ell'} := \frac{1}{n^2}\sum_{k,k'=1}^{n} \varphi_\ell(\boldsymbol{x}_k, \boldsymbol{y}_{k'})\varphi_{\ell'}(\boldsymbol{x}_k, \boldsymbol{y}_{k'}),$$

$$\widehat{h}_\ell := \frac{1}{n}\sum_{k=1}^{n} \varphi_\ell(\boldsymbol{x}_k, \boldsymbol{y}_k).$$

This is basically the same formulation as uLSIF.

For the KLIEP-type loss function, we may formulate the optimization criterion as

$$\max_{\{\alpha_\ell\}_{\ell=1}^b} \left[\sum_{k=1}^n \log\left(\sum_{\ell=1}^b \alpha_\ell \varphi_\ell(\boldsymbol{x}_k, \boldsymbol{y}_k)\right)\right]$$
$$\text{subject to } \frac{1}{n^2}\sum_{\ell=1}^b \alpha_\ell \sum_{k,k'=1}^n \varphi_\ell(\boldsymbol{x}_k, \boldsymbol{y}_{k'}) = 1 \quad \text{and} \quad \alpha_1, \alpha_2, \ldots, \alpha_b \geq 0.$$

The uLSIF-type algorithm would be suitable for estimating squared-loss mutual information (14), while the KLIEP-type algorithm would be suited for estimating log-loss mutual information (13).

3.4.2 Variable Selection

Mutual information can be used for measuring independence between random variables [30, 31] since it vanishes if and only if $\boldsymbol{x}$ and $\boldsymbol{y}$ are independent. Thus we can use density-ratio estimation methods for variable selection.

Let

$$\boldsymbol{x} = (x^{(1)}, x^{(2)}, \ldots, x^{(d)})^\top.$$

If mutual information between a variable $x^{(m)}$ and an output $\boldsymbol{y}$ is small, the variable $x^{(m)}$ is irrelevant to the prediction of $\boldsymbol{y}$; on the other hand, a variable with large mutual information is relevant for prediction. Thus, variable selection can be carried out, e.g., by ranking all the variables according to estimated mutual information values and choosing the leading variables for subsequent learning procedures.

Note that the range of application of this method is not limited to selecting a single variable, but this method can also be used for selecting a subset of variables based on mutual information between variable subsets and the output value. See the paper [12] for applications in bioinformatics.

3.4.3 Feature Extraction

Variable selection is restricted to choosing a subset of variables. In contrast, feature extraction tries to construct new features from original variables, typically in the form of a linear combination of original variables [32, 33].

From the viewpoint of independence between random variables, the most informative feature $\boldsymbol{\xi}^{(1)}$ ($\in \mathbb{R}^d$) for predicting the output value $\boldsymbol{y}$ is given by

$$\boldsymbol{\xi}^{(1)} := \operatorname*{argmax}_{\boldsymbol{\xi}} I(X_{\boldsymbol{\xi}}, Y) \quad \text{subject to } ||\boldsymbol{\xi}|| = 1,$$

where $X_{\boldsymbol{\xi}}$ is the orthogonal projection of the input vector X onto $\boldsymbol{\xi}$. The next informative feature may be sought iteratively in the orthogonal complement of the features found so

far. More specifically, the $(m+1)$-th feature is obtained as

$$\boldsymbol{\xi}^{(m+1)} := \operatorname*{argmax}_{\boldsymbol{\xi}} I(X_{\boldsymbol{\xi}}, Y)$$
$$\text{subject to } ||\boldsymbol{\xi}|| = 1, \quad (\boldsymbol{\xi}^{(1)}, \boldsymbol{\xi}^{(2)}, \ldots, \boldsymbol{\xi}^{(m)})^\top \boldsymbol{\xi} = \mathbf{0}_m.$$

Then the final M-dimensional feature vector of a sample $\boldsymbol{x}$ is constructed as

$$(\boldsymbol{\xi}^{(1)}, \boldsymbol{\xi}^{(2)}, \ldots, \boldsymbol{\xi}^{(M)})^\top \boldsymbol{x}.$$

Thus, density-ratio estimation methods can be used for feature extraction.

See the paper [17] for technical details of density-ratio based feature extraction, where M components are searched in a batch manner in the context of sufficient dimension reduction [62].

3.4.4 Independent Component Analysis

Suppose there is a d-dimensional random signal

$$\boldsymbol{x} = (x^{(1)}, x^{(2)}, \ldots, x^{(d)})^\top$$

drawn from $p(\boldsymbol{x})$, where $x^{(1)}, x^{(2)}, \ldots, x^{(d)}$ are statistically independent of each other, i.e., $p(\boldsymbol{x})$ is factorized as

$$p(\boldsymbol{x}) = p_1(x^{(1)}) p_2(x^{(2)}) \cdots p_d(x^{(d)}).$$

We cannot directly observe the signal $\boldsymbol{x}$, but we are only given a linearly mixed signal $\boldsymbol{y}$:

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x},$$

where $\boldsymbol{A}$ is a $d \times d$ invertible matrix called the *mixing matrix*. The goal of independent component analysis[34] is, given mixed signal samples $\{\boldsymbol{y}_i\}_{i=1}^n$, to obtain a *de-mixing matrix* $\boldsymbol{W}$ that recovers the original signal $\boldsymbol{x}$:

$$\widehat{\boldsymbol{x}} = \boldsymbol{W}\boldsymbol{y}.$$

The ideal solution is given by

$$\boldsymbol{W} = \boldsymbol{A}^{-1},$$

but we do not care about the permutation and scaling of each component of $\widehat{\boldsymbol{x}}$.

A direct approach to independent component analysis is to determine $\boldsymbol{W}$ so that each component of $\widehat{\boldsymbol{x}}$ is as independent as possible. Here we can use mutual information as an independence measure.

$$I(\widehat{X}^{(1)}, \widehat{X}^{(2)}, \ldots, \widehat{X}^{(d)}) := \int p(\widehat{\boldsymbol{x}}) \log \frac{p(\widehat{\boldsymbol{x}})}{\prod_{m=1}^d p(\widehat{x}^{(m)})} d\widehat{\boldsymbol{x}}.$$

Thus density-ratio estimation methods could be used for independent component analysis.

See the paper [13] for technical details.

4 Experiments

In this section, we illustrate the usefulness of the density ratio methods through various experiments.

4.1 Comparison of Density-ratio Estimation Methods

First, we compare the experimental performance of the density-ratio estimation methods reviewed in Section 2.

Let the dimension of the input space be d and

$$q(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; (0, 0, \ldots, 0)^\top, \boldsymbol{I}_d), \tag{16}$$

$$q'(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; (1, 0, \ldots, 0)^\top, \boldsymbol{I}_d), \tag{17}$$

where $\mathcal{N}(\cdot; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the Gaussian density with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ and $\boldsymbol{I}_d$ is the d-dimensional identity matrix. The task is to estimate the density ratio at $\{\boldsymbol{x}_i\}_{i=1}^n$:

$$r_i := r(\boldsymbol{x}_i) = \frac{q'(\boldsymbol{x}_i)}{q(\boldsymbol{x}_i)} \quad \text{for } i = 1, 2, \ldots, n. \tag{18}$$

We compare the following methods:

KDE(CV): The Gaussian kernel (1) is used, where the kernel widths for estimating $q(\boldsymbol{x})$ and $q'(\boldsymbol{x})$ are separately optimized based on 5-fold likelihood cross-validation.

KMM(med): The performance of KMM is dependent on B, ϵ, and σ. We set $B = 1000$ and $\epsilon = (\sqrt{n} - 1)/\sqrt{n}$ following the original paper [4], and the Gaussian width σ is set at the median distance among all pairs of samples in $\{\boldsymbol{x}_i\}_{i=1}^n \cup \{\boldsymbol{x}'_j\}_{j=1}^{n'}$, following the heuristic used in the references [38, 33].

LogReg(CV): Gaussian kernels are used as basis functions, where the kernel width σ and the regularization parameter λ are chosen based on 5-fold cross-validation.

KLIEP(CV): A Gaussian kernel model is used, where the kernel width σ is selected based on 5-fold likelihood cross-validation.

uLSIF(CV): A Gaussian kernel model is used, where the kernel width σ and the regularization parameter λ are determined based on leave-one-out cross-validation.

We did not include LSIF due to its numerical instability. All the methods are implemented using the *MATLAB*® environment, where the *CPLEX*® optimizer is used for solving quadratic programs in KMM and the *LIBLINEAR* implementation is used for LogReg [63].

We fix $n' = 1000$ and consider the following two settings for n and d:

(a) $n = 100$ and $d = 1, 2, \ldots, 20$,

(b) $d = 10$ and $n = 50, 60, \ldots, 150$.

We run the experiments 100 times for each d, each n, and each method, and evaluate the quality of the importance estimates $\{\widehat{r}_i\}_{i=1}^n$ by the *normalized mean squared error* (NMSE):

$$\mathrm{NMSE} := \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\widehat{r}_i}{\sum_{i'=1}^{n}\widehat{r}_{i'}} - \frac{r_i}{\sum_{i'=1}^{n} r_{i'}}\right)^2. \tag{19}$$

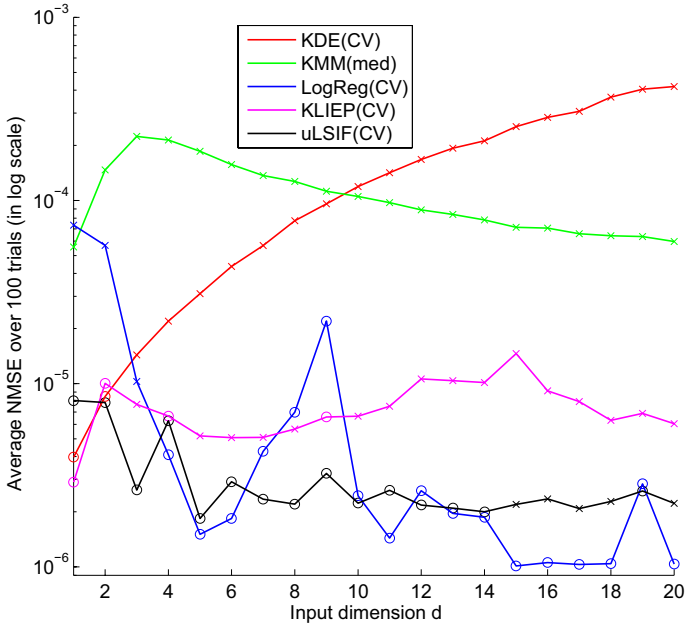
NMSEs averaged over 100 trials (a) as a function of the input dimensionality d and (b) as a function of the sample size n are plotted in log scale in Figure 5. Error bars are omitted for clear visibility—instead, the best method in terms of the mean error and comparable ones based on the *t-test* [58] at the significance level 1% are indicated by '$\circ$'; the methods with significant difference are indicated by '$\times$'.

Figure 5(a) shows that the error of KDE(CV) sharply increases as the input dimension grows, while LogReg, KLIEP, and uLSIF tend to give much smaller errors than KDE. This would be the fruit of directly estimating the density ratio without going through density estimation. KMM tends to perform poorly, which is caused by an inappropriate choice of the Gaussian kernel width. On the other hand, model selection in LogReg, KLIEP, and uLSIF seems to work quite well. Figure 5(b) shows that the errors of all methods tend to decrease as the number of samples grows. Again, LogReg, KLIEP, and uLSIF tend to give much smaller errors than KDE and KMM.

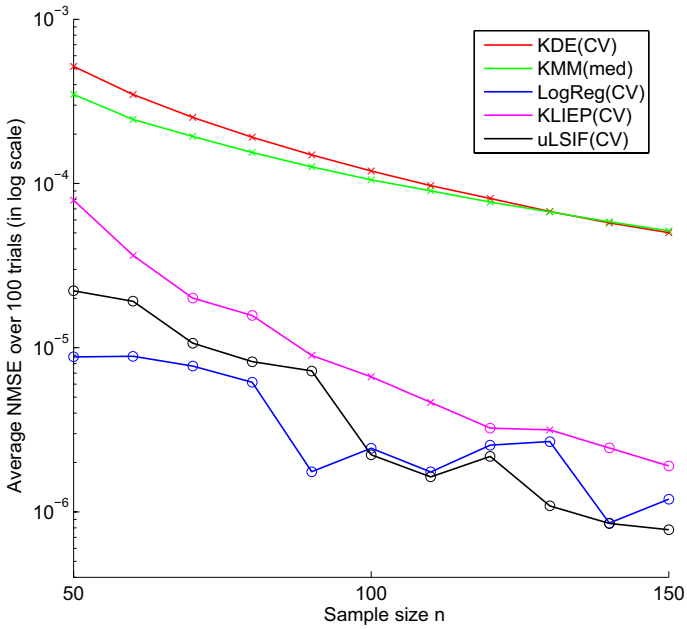
Next we investigate the computation time. Each method has a different model selection strategy, i.e., KMM does not involve model selection, KDE and KLIEP involve cross-validation over the kernel width, and LogReg and uLSIF involve cross-validation over both the kernel width and the regularization parameter. Thus the naive comparison of the total computation time is not so meaningful. For this reason, we first investigate the computation time of each density-ratio estimation method after the model parameters are fixed.

The average CPU computation time over 100 trials are summarized in Figure 6. Figure 6(a) shows that the computation time of KDE, KLIEP, and uLSIF is almost independent of the input dimensionality, while that of KMM and LogReg is highly dependent on the input dimensionality. Among them, uLSIF is one of the fastest methods. Figure 6(b) shows that the computation time of LogReg, KLIEP, and uLSIF is nearly independent of the number of samples, while that of KDE and KMM sharply increase as the number of samples increases.

Both LogReg and uLSIF have very good accuracy and their computation time after model selection is comparable. Finally, we compare the entire computation time of LogReg and uLSIF including cross-validation, which is summarized in Figure 7. We note that the Gaussian width σ and the regularization parameter λ are chosen over the 9×9 equidistant grid in this experiment for both LogReg and uLSIF. Therefore, the comparison of the entire computation time is fair. Figure 7(a) and Figure 7(b) show that uLSIF is approximately 5 times faster than LogReg under the current setup. More practical comparison of computation time will be shown in the next subsections.

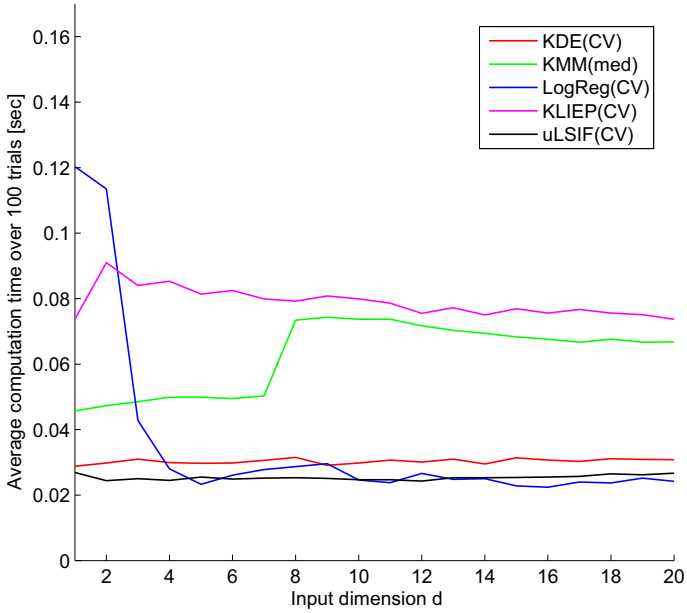


(a) When the input dimensionality d is changed.

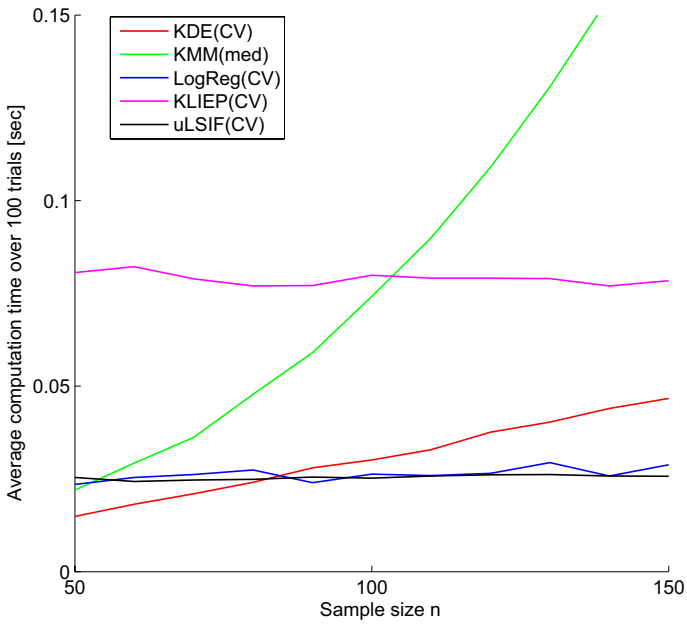


(b) When the sample size n is changed.

Figure 5: NMSEs averaged over 100 trials in log scale. Error bars are omitted for clear visibility. Instead, the best method in terms of the mean error and comparable ones based on the *t-test* at the significance level 1% are indicated by '$\circ$'; the methods with significant difference are indicated by '$\times$'.

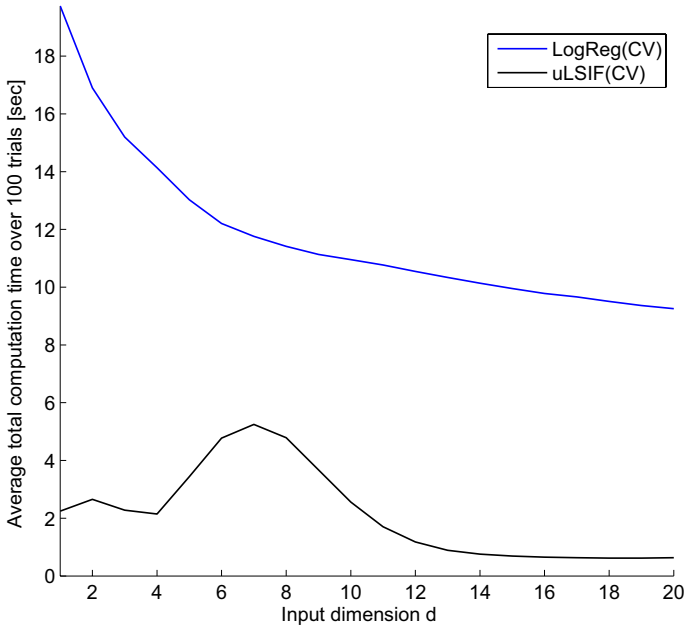


(a) When the input dimensionality d is changed.

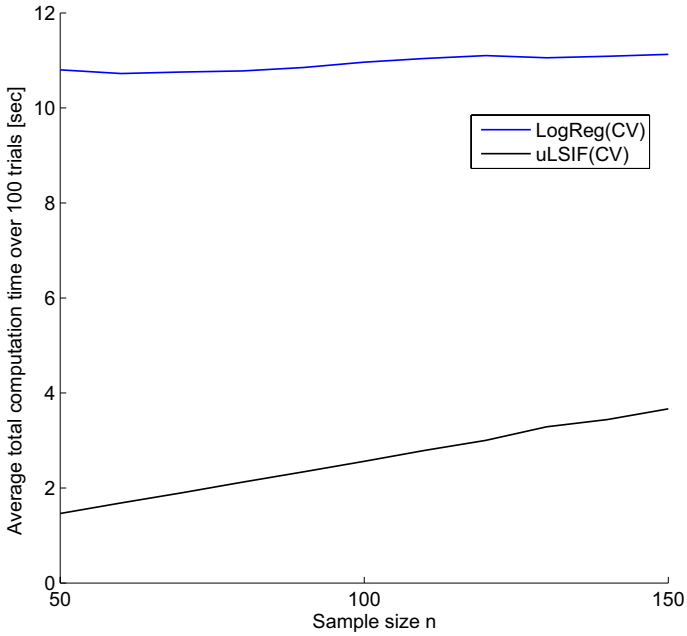


(b) When the sample size n is changed.

Figure 6: Average computation time (after model selection) over 100 trials.



(a) When the input dimensionality d is changed.



(b) When the sample size n is changed.

Figure 7: Average computation time over 100 trials (including model selection of the Gaussian width σ and the regularization parameter λ over the 9×9 grid).

Overall, uLSIF is shown to be the most computationally-efficient method among the group of the most accurate methods.

4.2 Covariate Shift Adaptation in Regression and Classification

Next, we illustrate the performance of the importance estimation methods in covariate shift adaptation.

In addition to training input samples $\{\boldsymbol{x}_i\}_{i=1}^n$ drawn from a training input density $q(\boldsymbol{x})$ and test input samples $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$ drawn from a test input density $q'(\boldsymbol{x})$, suppose that we are given training *output* samples $\{y_i\}_{i=1}^n$ at the training input points $\{\boldsymbol{x}_i\}_{i=1}^n$. The task is to predict the outputs $\{y'_j\}_{j=1}^{n'}$ for the test inputs $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$ based on the input-output training samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$.

We use the following kernel model for function learning:

$$\widehat{f}(\boldsymbol{x};\boldsymbol{\theta}) = \sum_{\ell=1}^{t} \theta_\ell K_h(\boldsymbol{x}, \boldsymbol{m}_\ell),$$

where $K_h(\boldsymbol{x}, \boldsymbol{x}')$ is the Gaussian kernel (1) and $\boldsymbol{m}_\ell$ is a template point randomly chosen from $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$ without replacement. We set the number of kernels at $t = 50$. We learn the parameter $\boldsymbol{\theta}$ by *importance-weighted regularized least-squares* (IWRLS) [64, 3]:

$$\widehat{\boldsymbol{\theta}}_{\mathrm{IWRLS}} := \operatorname*{argmin}_{\boldsymbol{\theta}\in\mathbb{R}^t} \left[\sum_{i=1}^{n} \widehat{r}(\boldsymbol{x}_i) \left(\widehat{f}(\boldsymbol{x}_i;\boldsymbol{\theta}) - y_i \right)^2 + \gamma ||\boldsymbol{\theta}||^2 \right]. \tag{20}$$

It is known that IWRLS is consistent when the true importance $r(\boldsymbol{x}_i)$ is used as weights—unweighted RLS is not consistent due to covariate shift, given that the true learning target function $f(\boldsymbol{x})$ is not realizable by the model $\widehat{f}(\boldsymbol{x})$ [22].

The solution $\widehat{\boldsymbol{\theta}}_{\mathrm{IWRLS}}$ is analytically given by

$$\widehat{\boldsymbol{\theta}}_{\mathrm{IWRLS}} = (\boldsymbol{K}^\top \widehat{\boldsymbol{W}} \boldsymbol{K} + \gamma \boldsymbol{I}_b)^{-1} \boldsymbol{K}^\top \widehat{\boldsymbol{W}} \boldsymbol{y},$$

where

$$\begin{aligned} K_{i,\ell} &:= K_h(\boldsymbol{x}_i, \boldsymbol{m}_\ell), \\ \widehat{\boldsymbol{W}} &:= \mathrm{diag}\left(\widehat{r}(\boldsymbol{x}_1), \widehat{r}(\boldsymbol{x}_2), \ldots, \widehat{r}(\boldsymbol{x}_n)\right), \\ \boldsymbol{y} &:= (y_1, y_2, \ldots, y_n)^\top. \end{aligned}$$

$\mathrm{diag}\,(a, b, \ldots, c)$ denotes the diagonal matrix with the diagonal elements $a, b, \ldots, c$.

The kernel width h and the regularization parameter γ in IWRLS (20) are chosen by *importance-weighted cross-validation* (IWCV) [5]. More specifically, we first divide the training samples $\{\boldsymbol{z}_i \mid \boldsymbol{z}_i = (\boldsymbol{x}_i, y_i)\}_{i=1}^n$ into R disjoint subsets $\{\mathcal{Z}_r\}_{r=1}^R$. Then a function $\widehat{f}_r(\boldsymbol{x})$ is learned using $\{\mathcal{Z}_j\}_{j\neq r}$ by IWRLS and its mean test error for the remaining samples $\mathcal{Z}_r$ is computed:

$$\frac{1}{|\mathcal{Z}_r|} \sum_{(\boldsymbol{x},y)\in\mathcal{Z}_r} \widehat{r}(\boldsymbol{x}) \mathrm{loss}\left(\widehat{f}_r(\boldsymbol{x}), y\right),$$

where

$$\mathrm{loss}\,(\widehat{y}, y) := \begin{cases} (\widehat{y} - y)^2 & \text{(Regression)}, \\ \frac{1}{2}(1 - \mathrm{sign}\{\widehat{y}y\}) & \text{(Classification)}. \end{cases}$$

We repeat this procedure for $r = 1, 2, \ldots, R$ and choose the kernel width h and the regularization parameter γ so that the average of the above mean test error over all r is minimized. We set the number of folds in importance-weighted cross-validation to $R = 5$. Importance-weighted cross-validation is shown to be an (almost) unbiased estimator of the generalization error, while unweighted cross-validation with misspecified models is biased due to covariate shift [2, 5].

The datasets provided by DELVE [65] and IDA [66] are used for performance evaluation. Each dataset consists of input/output samples $\{(\boldsymbol{x}_k, y_k)\}_{k=1}^n$. We normalize all the input samples $\{\boldsymbol{x}_k\}_{k=1}^n$ into $[0,1]^d$ and choose the test samples $\{(\boldsymbol{x}'_j, y'_j)\}_{j=1}^{n'}$ from the pool $\{(\boldsymbol{x}_k, y_k)\}_{k=1}^n$ as follows. We randomly choose one sample $(\boldsymbol{x}_k, y_k)$ from the pool and accept this with probability $\min(1, 4(x_k^{(c)})^2)$, where $x_k^{(c)}$ is the c-th element of $\boldsymbol{x}_k$ and c is randomly determined and fixed in each trial of the experiments. Then we remove $\boldsymbol{x}_k$ from the pool regardless of its rejection or acceptance, and repeat this procedure until n' samples are accepted. We choose the training samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ uniformly from the rest. Thus, in this experiment, the test input density tends to be lower than the training input density when $x_k^{(c)}$ is small. We set the number of samples at $n = 100$ and $n' = 500$ for all datasets. Note that we only use $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ and $\{\boldsymbol{x}'_j\}_{j=1}^{n'}$ for training regressors or classifiers; the test output values $\{y'_j\}_{j=1}^{n'}$ are used only for evaluating the generalization performance.

We run the experiments 100 times for each dataset and evaluate the *mean test error*:

$$\frac{1}{n'} \sum_{j=1}^{n'} \mathrm{loss}\left(\widehat{f}(\boldsymbol{x}'_j), y'_j\right).$$

The results are summarized in Table 2, where 'Uniform' denotes uniform weights (or equivalently, no importance weight). The numbers in the brackets are the standard deviation. All the error values are normalized so that the mean error of Uniform is one. For each dataset, the best method in terms of the mean error and comparable ones based on the *Wilcoxon signed rank test* at the significance level 1% are described in bold face. The upper half of the table corresponds to regression datasets taken from DELVE [65], while the lower half correspond to classification datasets taken from IDA [66]. All the methods are implemented using the *MATLAB*® environment, where the *CPLEX*® optimizer is used for solving quadratic programs in KMM and the *LIBLINEAR* implementation is used for LogReg [63].

The table shows that the generalization performance of KLIEP and uLSIF tends to be better than that of Uniform, KDE, KMM, and LogReg. The mean computation time over 100 trials is described in the bottom row of the table, where the value is normalized so that the computation time of uLSIF is one. This shows that the computation time of uLSIF is much shorter than KLIEP. Thus, uLSIF is overall shown to be useful in covariate shift adaptation.

Table 2: Mean test error averaged over 100 trials for covariate shift adaptation in regression and classification. The numbers in the brackets are the standard deviation. All the error values are normalized by that of 'Uniform' (uniform weighting, or equivalently no importance weighting). For each dataset, the best method in terms of the mean error and comparable ones based on the *Wilcoxon signed rank test* at the significance level 1% are described in bold face. The upper half corresponds to regression datasets taken from DELVE [65], while the lower half correspond to classification datasets taken from IDA [66]. All the methods are implemented using the *MATLAB*® environment, where the *CPLEX*® optimizer is used for solving quadratic programs in KMM and the *LIBLINEAR* implementation is used for LogReg [63].

Data	Uniform	KDE (CV)	KMM (med)	LogReg (CV)	KLIEP (CV)	uLSIF (CV)
kin-8fh	1.00(0.34)	1.22(0.52)	1.55(0.39)	1.31(0.39)	**0.95(0.31)**	**1.02(0.33)**
kin-8fm	1.00(0.39)	1.12(0.57)	1.84(0.58)	1.38(0.57)	**0.86(0.35)**	**0.88(0.39)**
kin-8nh	**1.00(0.26)**	1.09(0.20)	1.19(0.29)	1.09(0.19)	**0.99(0.22)**	**1.02(0.18)**
kin-8nm	**1.00(0.30)**	1.14(0.26)	1.20(0.20)	1.12(0.21)	**0.97(0.25)**	1.04(0.25)
abalone	**1.00(0.50)**	1.02(0.41)	**0.91(0.38)**	**0.97(0.49)**	**0.94(0.67)**	**0.96(0.61)**
image	**1.00(0.51)**	0.98(0.45)	1.08(0.54)	**0.98(0.46)**	**0.94(0.44)**	**0.98(0.47)**
ringnorm	1.00(0.04)	0.87(0.04)	**0.87(0.04)**	0.95(0.08)	0.99(0.06)	0.91(0.08)
twonorm	1.00(0.58)	1.16(0.71)	**0.94(0.57)**	**0.91(0.61)**	**0.91(0.52)**	**0.88(0.57)**
waveform	1.00(0.45)	1.05(0.47)	0.98(0.31)	**0.93(0.32)**	**0.93(0.34)**	**0.92(0.32)**
Average	1.00(0.38)	1.07(0.40)	1.17(0.37)	1.07(0.37)	0.94(0.35)	0.96(0.36)
Comp. time	—	0.82	3.50	3.27	2.23	1.00

4.3 Inlier-based Outlier Detection

Next, we apply importance estimation methods to outlier detection.

We again test KMM(med), LogReg(CV), KLIEP(CV), and uLSIF(CV) for importance estimation; in addition, we include native outlier detection methods for comparison purposes. The outlier detection problem that the native methods used below solve is to find outliers in a single dataset $\{\boldsymbol{x}_k\}_{k=1}^n$—the native methods can be employed in the current scenario (i.e., inlier-based outlier detection) just by finding outliers from all samples:

$$\{\boldsymbol{x}_k\}_{k=1}^n = \{\boldsymbol{x}_i\}_{i=1}^n \cup \{\boldsymbol{x}'_j\}_{j=1}^{n'}.$$

One-class support vector machine (OSVM): The *support vector machine* (SVM) [1, 38] 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 [24]. 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 convex quadratic programming problem:

$$\min_{\{r_k\}_{k=1}^n} \frac{1}{2}\sum_{k,k'=1}^{n} r_k r_{k'} K_\sigma(\boldsymbol{x}_k, \boldsymbol{x}_{k'})$$
$$\text{subject to } \sum_{k=1}^{n} r_k = 1 \text{ and } 0 \le r_1, r_2, \ldots, r_n \le \frac{1}{\nu n},$$

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

We use the inverse distance of a sample from the separating hyperplane as an outlier score. The OSVM solution is dependent on the outlier ratio ν and the Gaussian kernel width σ, and there seems to be no systematic method to determine the values of these tuning parameters. Here we use the median distance between samples as the Gaussian width, which is a popular heuristic [38, 33]. The value of ν is fixed to the true output ratio, i.e., the ideal optimal value. Thus the simulation results below should be slightly in favor of OSVM.

Local outlier factor (LOF): LOF is the score to detect a local outlier which lies relatively far from the nearest dense region [23]. For a prefixed natural number k, the LOF value of a sample $\boldsymbol{x}$ is defined by

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

where $\mathrm{nearest}_i(\boldsymbol{x})$ denotes the i-th nearest neighbor of $\boldsymbol{x}$ and $\mathrm{imd}_k(\boldsymbol{x})$ denotes the inverse mean distance from $\boldsymbol{x}$ to its k nearest neighbors:

$$\mathrm{imd}_k(\boldsymbol{x}) = \frac{1}{\frac{1}{k}\sum_{i=1}^{k} \|\boldsymbol{x} - \mathrm{nearest}_i(\boldsymbol{x})\|}.$$

If $\boldsymbol{x}$ alone is apart from a cloud of points, $\mathrm{imd}_k(\boldsymbol{x})$ tends to become smaller than $\mathrm{imd}_k(\mathrm{nearest}_i(\boldsymbol{x}))$ for all i. Then the LOF value gets large and therefore such a point is regarded as an outlier. The performance of LOF depends on the choice of the parameter k and there seems no systematic way to find an appropriate value of k. Here we test several different values of k.

Kernel density estimator (KDE'): A naive density estimation of all data samples $\{\boldsymbol{x}_k\}_{k=1}^n$ can also be used for outlier detection since the density value itself could be regarded as an outlier score. We use KDE with the Gaussian kernel (1) for density estimation, where the kernel width is determined based on 5-fold likelihood cross-validation.

All the methods are implemented using the R environment—we use the *ksvm* routine in the *kernlab* package for OSVM [67] and the *lofactor* routine in the *dprep* package for LOF [68].

The datasets provided by IDA [66] are used for performance evaluation. These datasets are binary classification datasets consisting of positive/negative and training/test samples. We allocate all positive training samples for the "model" set, while all positive test samples and a fraction ρ $(= 0.01, 0.02, 0.05)$ of negative test samples are assigned in the "evaluation" set. Thus, we regard the positive samples as regular and the negative samples as irregular.

When evaluating the performance of outlier detection methods, 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 the detection accuracy, we adopt the area under the ROC curve (AUC) as our error metric [69].

The mean AUC values over 20 trials as well as the computation time in Table 3, where the value is normalized so that the computation time of uLSIF is one. The table shows that uLSIF works fairly well. KLIEP works slightly better than uLSIF, but uLSIF is computationally much more efficient. LogReg overall works reasonably well, but it performs poorly for some datasets and the average AUC performance is not as good as uLSIF or KLIEP. KMM and OSVM are not comparable to uLSIF in AUC and they are computationally inefficient. Note that we also tested KMM and OSVM with several different Gaussian widths and experimentally found that the heuristic of using the median sample distance as the Gaussian kernel width works reasonably well in this experiment. Thus the AUC values of KMM and OSVM are close to optimal. LOF with large k is shown to work well, although it is not clear whether the heuristic of simply using large k is always appropriate or not. The computational cost of LOF is high since nearest neighbor search is computationally expensive. KDE' works reasonably well, but its performance is not as good as uLSIF and KLIEP.

Overall, uLSIF is shown to work well with low computational costs.

4.4 Conditional Density Estimation

We apply the uLSIF idea to conditional density estimation and investigate its practical performance using benchmark datasets.

We compare the uLSIF-based method with the following methods.

ϵ-neighbor Kernel Density Estimation (ϵ-KDE): For estimating the conditional density $p(\boldsymbol{y}|\boldsymbol{x})$, ϵ-neighbor kernel density estimation (ϵ-KDE) employs the standard kernel density estimator using a subset of samples, $\{\boldsymbol{y}_i\}_{i\in\mathcal{I}_{\boldsymbol{x},\epsilon}}$ for some threshold ϵ (≥ 0), where $\mathcal{I}_{\boldsymbol{x},\epsilon}$ is the set of sample indices such that

$$\|\boldsymbol{x}_i - \boldsymbol{x}\| \leq \epsilon.$$

In the case of Gaussian kernels, ϵ-KDE is expressed as

$$\widehat{p}(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{|\mathcal{I}_{\boldsymbol{x},\epsilon}|} \sum_{i\in\mathcal{I}_{\boldsymbol{x},\epsilon}} \mathcal{N}(\boldsymbol{y}; \boldsymbol{y}_i, \sigma^2 \boldsymbol{I}_{d_Y}),$$

Table 3: Mean AUC values for outlier detection over 20 trials for the benchmark datasets. All the methods are implemented using the R environment, where quadratic programs in KMM are solved by the *ipop* optimizer [67], the *ksvm* routine is used for OSVM [67], and the *lofactor* routine is used for LOF [68].

Data		uLSIF	KLIEP	LogReg	KMM	OSVM	LOF			KDE'
Name	ρ	(CV)	(CV)	(CV)	(med)	(med)	$k=5$	$k=30$	$k=50$	(CV)
banana	.01	.851	.815	.447	.578	.360	.838	.915	.919	.934
	.02	.858	.824	.428	.644	.412	.813	.918	.920	.927
	.05	.869	.851	.435	.761	.467	.786	.907	.909	.923
b-cancer	.01	.463	.480	.627	.576	.508	.546	.488	.463	.400
	.02	.463	.480	.627	.576	.506	.521	.445	.428	.400
	.05	.463	.480	.627	.576	.498	.549	.480	.452	.400
diabetes	.01	.558	.615	.599	.574	.563	.513	.403	.390	.425
	.02	.558	.615	.599	.574	.563	.526	.453	.434	.425
	.05	.532	.590	.636	.547	.545	.536	.461	.447	.435
f-solar	.01	.416	.485	.438	.494	.522	.480	.441	.385	.378
	.02	.426	.456	.432	.480	.550	.442	.406	.343	.374
	.05	.442	.479	.432	.532	.576	.455	.417	.370	.346
german	.01	.574	.572	.556	.529	.535	.526	.559	.552	.561
	.02	.574	.572	.556	.529	.535	.553	.549	.544	.561
	.05	.564	.555	.540	.532	.530	.548	.571	.555	.547
heart	.01	.659	.647	.833	.623	.681	.407	.659	.739	.638
	.02	.659	.647	.833	.623	.678	.428	.668	.746	.638
	.05	.659	.647	.833	.623	.681	.440	.666	.749	.638
s-image	.01	.812	.828	.600	.813	.540	.909	.930	.896	.916
	.02	.829	.847	.632	.861	.548	.785	.919	.880	.898
	.05	.841	.858	.715	.893	.536	.712	.895	.868	.892
splice	.01	.713	.748	.368	.541	.737	.765	.778	.768	.845
	.02	.754	.765	.343	.588	.744	.761	.793	.783	.848
	.05	.734	.764	.377	.643	.723	.764	.785	.777	.849
thyroid	.01	.534	.720	.745	.681	.504	.259	.111	.071	.256
	.02	.534	.720	.745	.681	.505	.259	.111	.071	.256
	.05	.534	.720	.745	.681	.485	.259	.111	.071	.256
titanic	.01	.525	.534	.602	.502	.456	.520	.525	.525	.461
	.02	.496	.498	.659	.513	.526	.492	.503	.503	.472
	.05	.526	.521	.644	.538	.505	.499	.512	.512	.433
t-norm	.01	.905	.902	.161	.439	.846	.812	.889	.897	.875
	.02	.896	.889	.197	.572	.821	.803	.892	.901	.858
	.05	.905	.903	.396	.754	.781	.765	.858	.874	.807
w-form	.01	.890	.881	.243	.477	.861	.724	.887	.889	.861
	.02	.901	.890	.181	.602	.817	.690	.887	.890	.861
	.05	.885	.873	.236	.757	.798	.705	.847	.874	.831
Average		.661	.685	.530	.608	.596	.594	.629	.622	.623
Comp. time		1.00	11.7	5.35	751	12.4	85.5			8.70

where $\mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the Gaussian density with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ and d_{Y} denotes the dimension of $\boldsymbol{y}$. The threshold ϵ and the bandwidth σ may be chosen based on likelihood cross-validation. ϵ-KDE is simple and easy-to-use, but is not reliable in high-dimensional problems. Slightly more sophisticated variants have been proposed based on weighted kernel density estimation[70, 71], but they still share the same weakness.

Mixture Density Network (MDN): The mixture density network (MDN) models the conditional density by a mixture of parametric densities[26]. In the case of Gaussian densities, MDN is expressed as

$$\widehat{p}(\boldsymbol{y}|\boldsymbol{x}) = \sum_{\ell=1}^{b} \pi_\ell(\boldsymbol{x}) \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_\ell(\boldsymbol{x}), \sigma_\ell^2(\boldsymbol{x}) \boldsymbol{I}_{d_{\mathrm{Y}}}),$$

where $\pi_\ell(\boldsymbol{x})$ denotes the mixing coefficient such that

$$\sum_{\ell=1}^{b} \pi_\ell(\boldsymbol{x}) = 1 \quad \text{and} \quad 0 \leq \pi_\ell(\boldsymbol{x}) \leq 1 \quad \text{for all } \boldsymbol{x} \in \mathcal{D}_{\mathrm{X}}.$$

All the parameters $\{\pi_\ell(\boldsymbol{x}), \boldsymbol{\mu}_\ell(\boldsymbol{x}), \sigma_\ell^2(\boldsymbol{x})\}_{\ell=1}^{b}$ are learned as a function of $\boldsymbol{x}$ by a neural network with regularized maximum likelihood estimation. The number b of Gaussian components, the number of hidden units in the neural network, and the regularization parameter may be chosen based on likelihood cross-validation. MDN has been shown to work well, although its training is time-consuming and only a local solution may be obtained due to non-convexity of neural network learning.

Kernel Quantile Regression (KQR): Kernel quantile regression (KQR) predicts the 100τ-percentile of the conditional distribution for a given τ ($\in (0, 1)$) when y is one-dimensional[27, 72]. For the Gaussian kernel model

$$\widehat{f}_\tau(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_{i,\tau} \phi_i(\boldsymbol{x}) + b_\tau,$$

the parameters $\{\alpha_{i,\tau}\}_{i=1}^{n}$ and b_τ are learned by

$$\min_{\{\alpha_{i,\tau}\}_{i=1}^{n}, b_\tau} \left[\sum_{i=1}^{n} \psi_\tau(y_i - \widehat{f}_\tau(\boldsymbol{x}_i)) + \lambda \sum_{i,j=1}^{n} \phi_i(\boldsymbol{x}_j) \alpha_{i,\tau} \alpha_{j,\tau} \right],$$

where

$$\psi_\tau(r) = \begin{cases} (1-\tau)|r| & (r \leq 0), \\ \tau |r| & (r > 0). \end{cases}$$

Thus, solving KQR for all $\tau \in (0, 1)$ gives an estimator of the entire conditional distribution. The bandwidth σ and the regularization parameter λ may be chosen based on cross-validation.

Table 4: Experimental results on benchmark datasets. The difference between out-of-sample negative log-likelihoods of uLSIF and the competing methods are described. The positive (negative) values indicate that the performance of uLSIF is better (worse) than the competing method. Mean computation time is described as the ratio to uLSIF (i.e., smaller is faster). d_{X} denotes the dimensionality of the input $\boldsymbol{x}$. The dimensionality of the output y is one in all the datasets.

Dataset	(n, d_{X})	vs. ϵ-KDE	vs. MDN	vs. KQR
caution	(50,2)	0.25	0.40	0.47
ftcollinssnow	(46,1)	−0.02	0.01	0.79
highway	(19,11)	−0.69	2.15	0.05
heights	(687,1)	0.03	0.14	0.35
sniffer	(62,4)	0.64	0.69	0.44
snowgeese	(22,2)	−1.64	3.32	−0.09
ufc	(117,4)	0.42	−0.21	0.16
birthwt	(94,7)	0.05	0.06	0.53
crabs	(100,6)	0.84	−1.28	−0.78
GAGurine	(157,1)	0.08	−0.49	0.10
geyser	(149,1)	0.22	0.46	0.56
gilgais	(182,8)	0.94	−0.41	1.74
topo	(26,2)	0.02	0.85	0.66
BostonHousing	(253,13)	−0.21	−0.39	−0.30
CobarOre	(19,2)	0.27	0.15	6.84
engel	(117,1)	0.99	−0.21	—
mcycle	(66,1)	−0.22	0.08	0.12
BigMac2003	(34,9)	−0.26	0.13	0.51
UN3	(62,6)	0.93	−0.20	0.02
cpus	(104,7)	0.47	0.82	—
Computation time		×0.0036	×37.0	×0.64

A key fact is that the solution of KQR is piecewise linear with respect to τ, so the entire solution path can be computed efficiently[28]. This implies that the conditional cumulative distribution can be computed efficiently. However, solution path tracking tends to be numerically rather unstable and the range of applications of KQR is limited to one-dimensional output y. Furthermore, some heuristic procedure is needed to convert conditional cumulative distributions into conditional densities, which can cause additional estimation errors.

We use the benchmark datasets described in Table 4, which are accompanied with the *R* package [73]. In each dataset, 50% of samples are randomly chosen for conditional density estimation and the rest are used for computing the estimation accuracy based on the out-of-sample log-likelihood. In MDN, the number of Gaussian components is set to

$b = 3$. The results are summarized in Table 4, showing that uLSIF compares favorably with the other methods in accuracy. Note that the solution of KQR was not available for the 'engel' and 'cpus' datasets due to numerical instability.

4.5 Mutual Information Estimation

Finally, we apply the KLIEP idea to estimating mutual information (13) and investigate its practical performance using artificial datasets.

We compare the KLIEP-based method with the following methods.

Kernel Density Estimator (KDE): Based on KDE, mutual information can be approximated using density estimates $\widehat{p}(\boldsymbol{x}, \boldsymbol{y})$, $\widehat{p}(\boldsymbol{x})$, and $\widehat{p}(\boldsymbol{y})$ (obtained from $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$, $\{\boldsymbol{x}_i\}_{i=1}^n$, and $\{\boldsymbol{y}_i\}_{i=1}^n$, respectively) as

$$\widehat{I}(X, Y) = \frac{1}{n} \sum_{i=1}^{n} \log \frac{\widehat{p}(\boldsymbol{x}_i, \boldsymbol{y}_i)}{\widehat{p}(\boldsymbol{x}_i)\widehat{p}(\boldsymbol{y}_i)}.$$

However, density estimation is known to be a hard problem and division by estimated densities may expand the estimation error. For this reason, the KDE-based approach may not be reliable in practice.

A more sophisticated density estimation approach uses histogram-based density estimators with data-dependent partition. In the context of estimating Kullback-Leibler divergence, consistency properties of histogram-based methods have been studied thoroughly [74, 75, 76]. However, these methods are not reliable in high-dimensional problems.

K-nearest Neighbor Method (KNN): Mutual information can be expressed in terms of the entropies as

$$I(X, Y) = H(X) + H(Y) - H(X, Y),$$

where $H(X)$ denotes the entropy of X:

$$H(X) := -\int q(\boldsymbol{x}) \log q(\boldsymbol{x}) d\boldsymbol{x}.$$

Thus mutual information can be approximated if the entropies $H(X)$, $H(Y)$, and $H(X, Y)$ are estimated.

Based on this expression, a mutual information estimator that utilizes the k-nearest neighbor distance (KNN) was developed [30]. Let us define the norm of $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{y})$ by

$$\|\boldsymbol{z}\|_{\mathrm{z}} := \max\{\|\boldsymbol{x}\|, \|\boldsymbol{y}\|\},$$

where $\|\cdot\|$ denotes the Euclidean norm. Let $\mathcal{N}_k(i)$ be the set of k-nearest neighbor samples of $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ with respect to the norm $\|\cdot\|_{\mathrm{z}}$, and let

$$\begin{aligned}
\epsilon_{\mathrm{x}}(i) &:= \max\{\|\boldsymbol{x}_i - \boldsymbol{x}_{i'}\| \mid (\boldsymbol{x}_{i'}, \boldsymbol{y}_{i'}) \in \mathcal{N}_k(i)\},\\
n_{\mathrm{x}}(i) &:= |\{\boldsymbol{z}_{i'} \mid \|\boldsymbol{x}_i - \boldsymbol{x}_{i'}\| \leq \epsilon_{\mathrm{x}}(i)\}|,\\
\epsilon_{\mathrm{y}}(i) &:= \max\{\|\boldsymbol{y}_i - \boldsymbol{y}_{i'}\| \mid (\boldsymbol{x}_{i'}, \boldsymbol{y}_{i'}) \in \mathcal{N}_k(i)\},\\
n_{\mathrm{y}}(i) &:= |\{\boldsymbol{z}_{i'} \mid \|\boldsymbol{y}_i - \boldsymbol{y}_{i'}\| \leq \epsilon_{\mathrm{y}}(i)\}|.
\end{aligned}$$

Then the KNN-based mutual information estimator is given by

$$\widehat{I}(X,Y) = \psi(k) + \psi(n) - \frac{1}{k} - \frac{1}{n}\sum_{i=1}^{n}\Big[\psi(n_x(i)) + \psi(n_y(i))\Big],$$

where ψ is the *digamma* function.

An advantage of the above KNN-based method is that it does not simply replace entropies with their estimates, but it is designed to cancel the error of individual entropy estimation. It is reported to perform better than the KDE-based method [77], and its consistency properties have been theoretically studied in the context of general Kullback-Leibler divergence estimation [78]. However, a drawback of the KNN-based approach in practice is that the estimation accuracy depends on the value of k and there seems no systematic strategy to choose the value of k appropriately.

Edgeworth Expansion Method (EDGE): An entropy approximator based on the *Edgeworth expansion* was proposed in the reference [31], where the entropy of a distribution is approximated by that of the normal distribution and some additional higher-order correction terms. More specifically, for a d-dimensional distribution, an estimator $\widehat{H}$ of the entropy H is given by

$$\widehat{H} = H_{\mathrm{normal}} - \frac{1}{12}\sum_{i=1}^{d}\kappa_{i,i,i}^2 - \frac{1}{4}\sum_{i,j=1,i\neq j}^{d}\kappa_{i,i,j}^2 - \frac{1}{72}\sum_{i,j,k=1,i<j<k}^{d}\kappa_{i,j,k}^2,$$

where H_{normal} is the entropy of the normal distribution with covariance matrix equal to the target distribution and $\kappa_{i,j,k}$ $(1 \leq i,j,k \leq d)$ is the standardized third cumulant of the target distribution. In practice, all the cumulants are estimated from samples.

Based on EDGE, mutual information can be approximated using entropy estimators $\widehat{H}(X)$, $\widehat{H}(Y)$, and $\widehat{H}(X,Y)$ as

$$\widehat{I}(X,Y) = \widehat{H}(X) + \widehat{H}(Y) - \widehat{H}(X,Y).$$

If the underlying distribution is close to the normal distribution, the above approximation is accurate and the EDGE method works well. However, if the distributions

are far from the normal distribution, the approximation error becomes large and therefore the EDGE method is unreliable.

In principle, it is possible to include the fourth and even higher cumulants for further reducing the estimation bias. However, this in turn increases the estimation variance; the expansion up to the third cumulants seems to be reasonable.

We use the following four datasets for experiments (see Figure 8):

(a) Linear dependence: y has a linear dependence on x as

$$x \sim \mathcal{N}(x; 0, 0.5) \quad \text{and} \quad y|x \sim \mathcal{N}(y; 3x, 1),$$

where $\mathcal{N}(x; \mu, \sigma^2)$ denotes the normal density with mean μ and variance σ^2.

(b) Non-linear dependence with correlation: y has a quadratic dependence on x as

$$x \sim \mathcal{N}(x; 0, 1) \quad \text{and} \quad y|x \sim \mathcal{N}(y; x^2, 1).$$

(c) Non-linear dependence without correlation: y has a lattice-structured dependence on x as

$$x \sim \mathcal{U}(x; -0.5, 0.5) \quad \text{and} \quad y|x \sim \begin{cases} \mathcal{N}(x; 0, \frac{1}{3}) & \text{if } x \leq |\frac{1}{6}|, \\ \frac{1}{2}(\mathcal{N}(x; 1, \frac{1}{3}) + \mathcal{N}(x; -1, \frac{1}{3})) & \text{otherwise,} \end{cases}$$

where $\mathcal{U}(x; a, b)$ denotes the uniform density on (a, b).

(d) Independence: x and y are independent to each other as

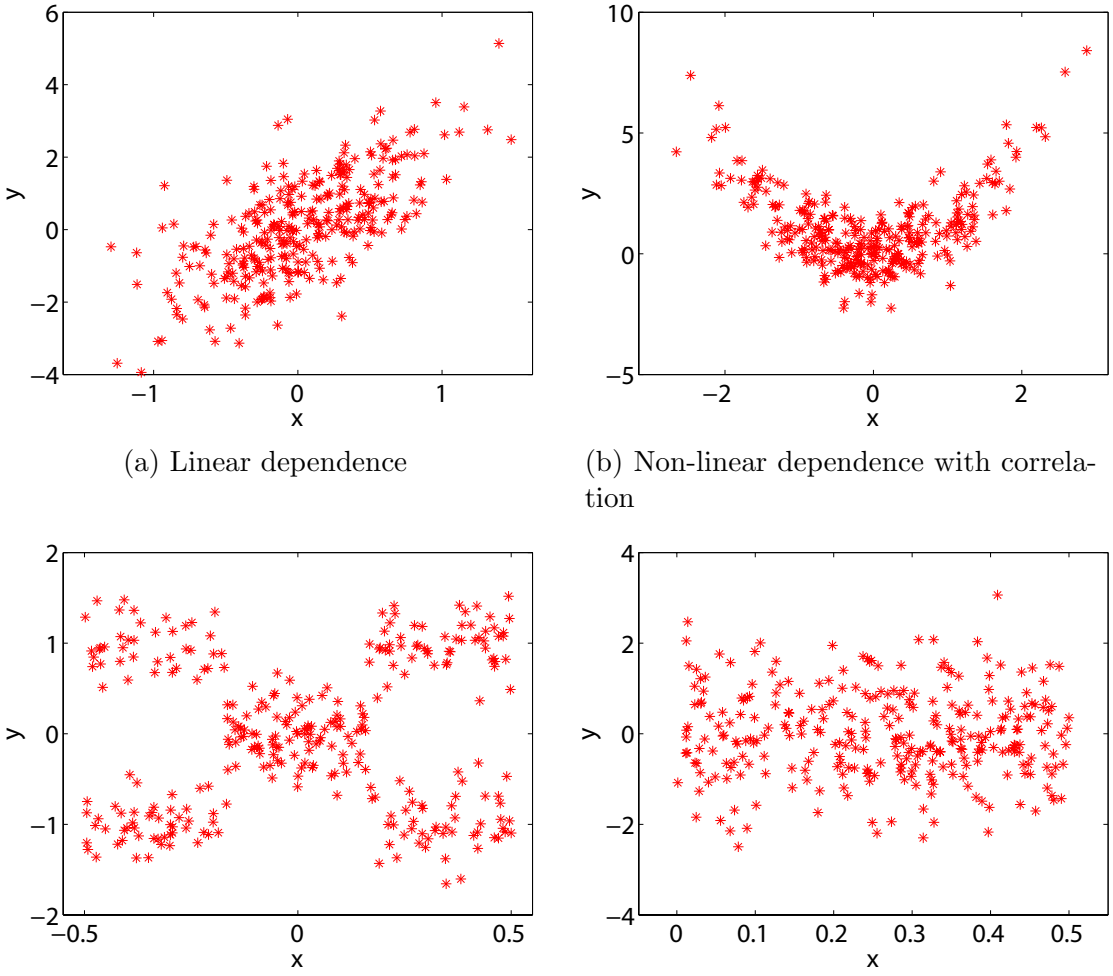
$$x \sim \mathcal{U}(x; 0, 0.5) \quad \text{and} \quad y|x \sim \mathcal{N}(y; 0, 1).$$

The task is to estimate mutual information (13) between x and y. We compare the performance of KLIEP (with cross validation), KDE (with cross validation), KNN (with $k = 1, 5, 15$), and EDGE; the approximation error of a mutual information estimate $\widehat{I}$ is measured by

$$|\widehat{I} - I|.$$

Figure 9 depicts the average approximation error—KLIEP, KDE, KNN with $k = 5$, and EDGE perform well for the dataset (a), KLIEP tends to outperform the other estimators for the dataset (b), KLIEP and KNN with $k = 5$ show the best performance against the other methods for the dataset (c), and KLIEP, EDGE, and KNN with $k = 15$ perform well for the dataset (d).

KDE works moderately well for the datasets (a)–(c), while it performs poorly for the dataset (d). This instability would be ascribed to devision by estimated densities, which tends to magnify the estimation error. KNN seems work well for all four datasets if the value of k is chosen optimally; the best value of k varies depending on the datasets and thus using a prefixed value of k is not appropriate. Therefore, k needs to be chosen *adaptively*



(a) Linear dependence

(b) Non-linear dependence with correlation

(c) Non-linear dependence without correlation

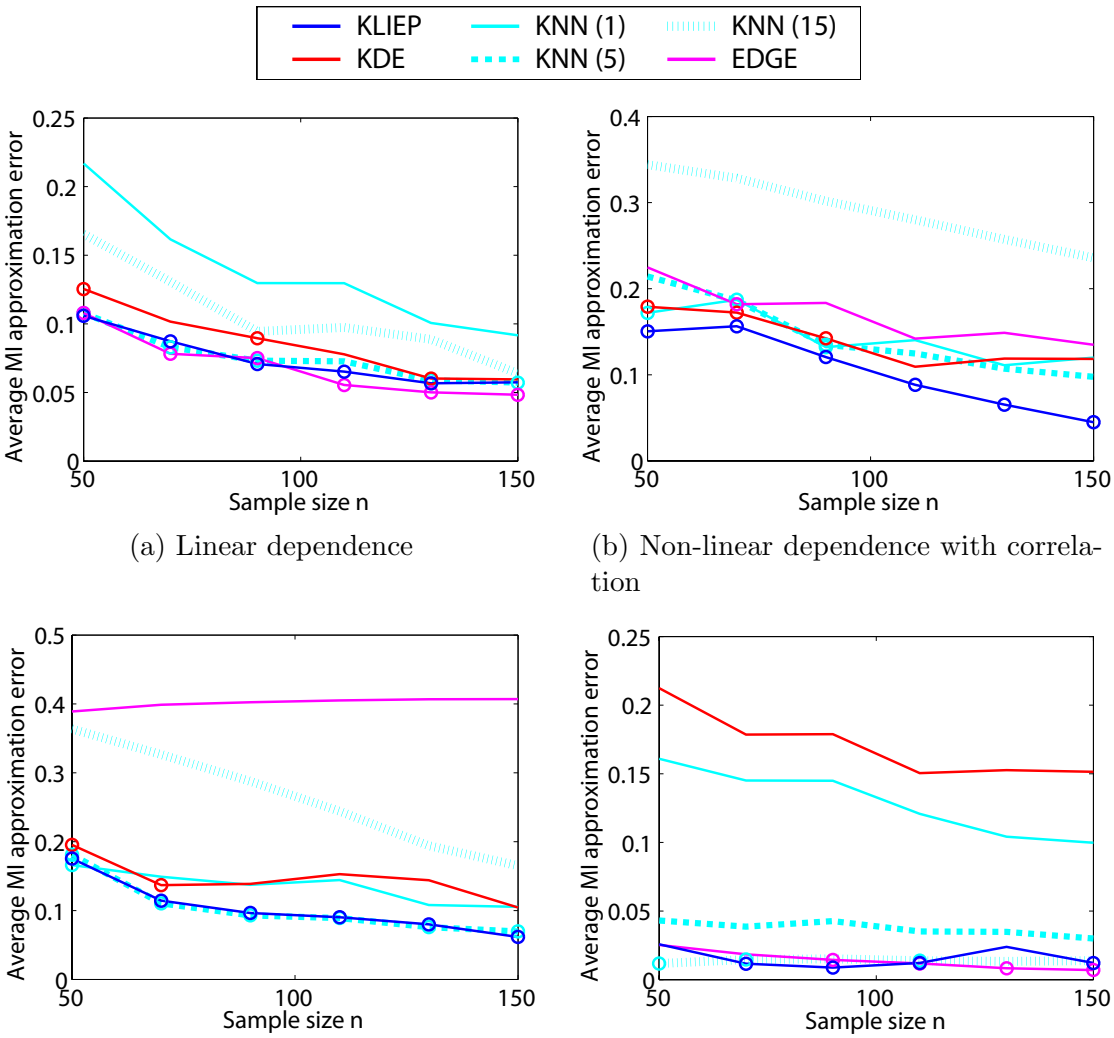
(d) Independence

Figure 8: Datasets used in mutual information estimation experiments.

using the data samples. However, there is no systematic model selection strategy for KNN and therefore KNN would be unreliable in practice. EDGE works well for the datasets (a), (b), and (d), which posses high normality[1]. However, for the dataset (c) where normality of the target distributions is low, the EDGE method performs poorly. In contrast, KLIEP with cross validation performs reasonably well for all four datasets in a stable manner.

These experimental results show that KLIEP nicely compensates for the weaknesses of the existing methods.

[1]Note that although the Edgeworth approximation is exact when the target distributions are precisely normal, the EDGE method still suffers from some estimation error since the cumulants are estimated from samples.



(a) Linear dependence

(b) Non-linear dependence with correlation

(c) Non-linear dependence without correlation

(d) Independence

Figure 9: MI approximation error measured by $|\widehat{I}-I|$ averaged over 100 trials as a function of the sample size n. The symbol '$\circ$' on a line means that the corresponding method is the best in terms of the average error or is judged to be comparable to the best method by the *t-test* at the significance level 1%.

5 Conclusions

Avoiding density estimation is crucial in statistical data processing, as advocated by Vladimir Vapnik. Following this idea, a new data analysis framework based on the ratio of two probability density functions has been developed recently and is gathering a lot of attention in the machine learning and data mining communities. In this paper, we gave a comprehensive review of density-ratio estimation methods and discussed advantages and disadvantages of the methods qualitatively and quantitatively. Then we explained that the density-ratio framework accommodates a wide variety of statistical data processing tasks such as non-stationarity adaptation, outlier detection, conditional density estimation, and mutual information estimation We expect that such a new approach could be useful in various computer vision applications.

So far, parametric and non-parametric convergence rates of density-ratio estimation methods have been investigated thoroughly [42, 4, 10, 8, 11] and algorithmic stability has been studied in the framework of smoothed analysis [47]. Further investigating advantages of direct density-ratio estimation over indirect density estimation approaches is an important future direction to pursue.

Acknowledgements

The authors would like to thank MEXT Grant-in-Aid for Young Scientists (A), 20680007, GCOE Computationism as a Foundation for the Sciences, JFE 21st Century Foundation, Support Center for Advanced Telecommunications Technology Research Foundation, and the Asian Office of Aerospace Research and Development for financial support.

References

[1] Vapnik, V. N.: *Statistical Learning Theory*, Wiley, New York (1998).

[2] Zadrozny, B.: Learning and Evaluating Classifiers under Sample Selection Bias, *Proceedings of the Twenty-First International Conference on Machine Learning*, New York, NY, ACM Press (2004).

[3] Sugiyama, M. and Müller, K.-R.: Input-dependent Estimation of Generalization Error under Covariate Shift, *Statistics & Decisions*, Vol. 23, No. 4, pp. 249–279 (2005).

[4] Huang, J., Smola, A., Gretton, A., Borgwardt, K. M. and Schölkopf, B.: Correcting Sample Selection Bias by Unlabeled Data, *Advances in Neural Information Processing Systems 19* (Schölkopf, B., Platt, J. and Hoffman, T., eds.), MIT Press, Cambridge, MA, pp. 601–608 (2007).

[5] Sugiyama, M., Krauledat, M. and Müller, K.-R.: Covariate Shift Adaptation by Importance Weighted Cross Validation, *Journal of Machine Learning Research*, Vol. 8, pp. 985–1005 (2007).

[6] Bickel, S., Brückner, M. and Scheffer, T.: Discriminative Learning for Differing Training and Test Distributions, *Proceedings of the 24th International Conference on Machine Learning* (2007).

[7] Hido, S., Tsuboi, Y., Kashima, H., Sugiyama, M. and Kanamori, T.: Inlier-based Outlier Detection via Direct Density Ratio Estimation, *Proceedings of IEEE International Conference on Data Mining (ICDM2008)* (Giannotti, F., Gunopulos, D., Turini, F., Zaniolo, C., Ramakrishnan, N. and Wu, X., eds.), Pisa, Italy, pp. 223–232 (2008).

[8] Suzuki, T., Sugiyama, M., Sese, J. and Kanamori, T.: Approximating Mutual Information by Maximum Likelihood Density Ratio Estimation, *JMLR Workshop and Conference Proceedings* (Saeys, Y., Liu, H., Inza, I., Wehenkel, L. and de Peer, Y. V., eds.), New Challenges for Feature Selection in Data Mining and Knowledge Discovery, Vol. 4, pp. 5–20 (2008).

[9] Sugiyama, M., Nakajima, S., Kashima, H., von Bünau, P. and Kawanabe, M.: Direct Importance Estimation with Model Selection and Its Application to Covariate Shift Adaptation, *Advances in Neural Information Processing Systems 20* (Platt, J. C., Koller, D., Singer, Y. and Roweis, S., eds.), Cambridge, MA, MIT Press, pp. 1433–1440 (2008).

[10] Sugiyama, M., Suzuki, T., Nakajima, S., Kashima, H., von Bünau, P. and Kawanabe, M.: Direct Importance Estimation for Covariate Shift Adaptation, *Annals of the Institute of Statistical Mathematics*, Vol. 60, No. 4, pp. 699–746 (2008).

[11] Kanamori, T., Hido, S. and Sugiyama, M.: Efficient Direct Density Ratio Estimation for Non-stationarity Adaptation and Outlier Detection, *Advances in Neural Information Processing Systems 21*, Cambridge, MA, MIT Press (2009). to appear.

[12] Suzuki, T., Sugiyama, M., Kanamori, T. and Sese, J.: Mutual Information Estimation Reveals Global Associations between Stimuli and Biological Processes, *BMC Bioinformatics*, Vol. 10, No. 1, p. S52 (2009).

[13] Suzuki, T. and Sugiyama, M.: Estimating Squared-loss Mutual Information for Independent Component Analysis., *Independent Component Analysis and Blind Signal Separation*, Lecture Notes in Computer Science, Berlin, Springer (2009). to appear.

[14] Quiñonero-Candela, J., Sugiyama, M., Schwaighofer, A. and Lawrence, N.(eds.): *Dataset Shift in Machine Learning*, MIT Press, Cambridge, MA (2009).

[15] Tsuboi, Y., Kashima, H., Hido, S., Bickel, S. and Sugiyama, M.: Direct Density Ratio Estimation for Large-scale Covariate Shift Adaptation, *IPSJ Journal*, Vol. 50, No. 4, pp. 1–19 (2009).

[16] Sugiyama, M., Takeuchi, I., Suzuki, T., Kanamori, T. and Hachiya, H.: Least-squares Conditional Density Estimation, Technical Report TR09-0004, Department of Computer Science, Tokyo Institute of Technology (2009).

[17] Suzuki, T. and Sugiyama, M.: Sufficient Dimension Reduction via Squared-loss Mutual Information Estimation, Technical Report TR09-0005, Department of Computer Science, Tokyo Institute of Technology (2009).

[18] Härdle, W., Müller, M., Sperlich, S. and Werwatz, A.: *Nonparametric and Semiparametric Models*, Springer, Berlin (2004).

[19] Qin, J.: Inferences for Case-control and Semiparametric Two-sample Density Ratio Models, *Biometrika*, Vol. 85, No. 3, pp. 619–639 (1998).

[20] Cheng, K. F. and Chu, C. K.: Semiparametric Density Estimation under a Two-sample Density Ratio Model, *Bernoulli*, Vol. 10, No. 4, pp. 583–604 (2004).

[21] Fishman, G. S.: *Monte Carlo: Concepts, Algorithms, and Applications*, Springer-Verlag, Berlin (1996).

[22] Shimodaira, H.: Improving Predictive Inference under Covariate Shift by Weighting the Log-Likelihood Function, *Journal of Statistical Planning and Inference*, Vol. 90, No. 2, pp. 227–244 (2000).

[23] Breunig, M. M., Kriegel, H.-P., Ng, R. T. and Sander, J.: LOF: Identifying Density-based Local Outliers, *Proceedings of the ACM SIGMOD International Conference on Management of Data* (Chen, W., Naughton, J. F. and Bernstein, P. A., eds.) (2000).

[24] Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J. and Williamson, R. C.: Estimating the Support of a High-Dimensional Distribution, *Neural Computation*, Vol. 13, No. 7, pp. 1443–1471 (2001).

[25] Hodge, V. and Austin, J.: A Survey of Outlier Detection Methodologies, *Artificial Intelligence Review*, Vol. 22, No. 2, pp. 85–126 (2004).

[26] Bishop, C. M.: *Pattern Recognition and Machine Learning*, Springer, New York (2006).

[27] Takeuchi, I., Le, Q. V., Sears, T. D. and Smola, A. J.: Nonparametric Quantile Estimation, *Journal of Machine Learning Research*, Vol. 7, pp. 1231–1264 (2006).

[28] Takeuchi, I., Nomura, K. and Kanamori, T.: Nonparametric Conditional Density Estimation Using Piecewise-linear Solution Path of Kernel Quantile Regression, *Neural Computation*, Vol. 21, No. 2, pp. 533–559 (2009).

[29] Cover, T. M. and Thomas, J. A.: *Elements of Information Theory*, John Wiley & Sons, Inc., New York, NY, USA (1991).

[30] Kraskov, A., Stögbauer, H. and Grassberger, P.: Estimating mutual information, *Physical Review E*, Vol. 69, No. 6, p. 066138 (2004).

[31] Hulle, M. M. V.: Edgeworth Approximation of Multivariate Differential Entropy, *Neural Computation*, Vol. 17, No. 9, pp. 1903–1910 (2005).

[32] Guyon, I. and Elisseeff, A.: An Introduction to Variable and Feature Selection, *Journal of Machine Learning Research*, Vol. 3, No. Mar, pp. 1157–1182 (2003).

[33] Song, L., Smola, A., Gretton, A., Borgwardt, K. M. and Bedo, J.: Supervised Feature Selection via Dependence Estimation, *Proceedings of the 24th International Conference on Machine learning*, New York, NY, USA, ACM, pp. 823–830 (2007).

[34] Hyvärinen, A., Karhunen, J. and Oja, E.: *Independent Component Analysis*, Wiley, New York (2001).

[35] Sugiyama, M., Kanamori, T., Suzuki, T., Hido, S., Sese, J., Takeuchi, I. and Wang, L.: Direct Importance Estimation—A New Versatile Tool for Statistical Pattern Recognition., *Proceedings of Meeting on Image Recognition and Understanding 2008 (MIRU2008)*, Nagano, Japan, pp. 29–36 (2008).

[36] Kullback, S. and Leibler, R. A.: On Information and Sufficiency, *Annals of Mathematical Statistics*, Vol. 22, pp. 79–86 (1951).

[37] Steinwart, I.: On the Influence of the Kernel on the Consistency of Support Vector Machines, *Journal of Machine Learning Research*, Vol. 2, pp. 67–93 (2001).

[38] Schölkopf, B. and Smola, A. J.: *Learning with Kernels*, MIT Press, Cambridge, MA (2002).

[39] Minka, T. P.: A Comparison of Numerical Optimizers for Logistic Regression, Technical report, Microsoft Research (2007).

[40] Bickel, S., Bogojeska, J., Lengauer, T. and Scheffer, T.: Multi-task Learning for HIV Therapy Screening, *Proceedings of 25th Annual International Conference on Machine Learning (ICML2008)* (McCallum, A. and Roweis, S., eds.), Helsinki, Finland, Omnipress, pp. 56–63 (2008).

[41] Boyd, S. and Vandenberghe, L.: *Convex Optimization*, Cambridge University Press, Cambridge (2004).

[42] Nguyen, X., Wainwright, M. and Jordan, M.: Estimating Divergence Functionals and the Likelihood Ratio by Penalized Convex Risk Minimization, *Advances in Neural Information Processing Systems 20* (Platt, J., Koller, D., Singer, Y. and Roweis, S., eds.), MIT Press, Cambridge, MA, pp. 1089–1096 (2008).

[43] Best, M. J.: An Algorithm for the Solution of the Parametric Quadratic Programming Problem, CORR Report 82-24, Faculty of Mathematics, University of Waterloo (1982).

[44] Efron, B., Hastie, T., Tibshirani, R. and Johnstone, I.: Least Angle Regression, *The Annals of Statistics*, Vol. 32, No. 2, pp. 407–499 (2004).

[45] Hastie, T., Rosset, S., Tibshirani, R. and Zhu, J.: The Entire Regularization Path for the Support Vector Machine, *Journal of Machine Learning Research*, Vol. 5, pp. 1391–1415 (2004).

[46] Kanamori, T., Hido, S. and Sugiyama, M.: A Least-squares Approach to Direct Importance Estimation, Technical Report TR08-0003, Department of Computer Science, Tokyo Institute of Technology (2008).

[47] Kanamori, T., Suzuki, T. and Sugiyama, M.: Condition Number Analysis of Kernel-based Density Ratio Estimation, Technical Report TR09-0006, Department of Computer Science, Tokyo Institute of Technology (2009).

[48] Golub, G. H. and Loan, C. F. V.: *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD (1996).

[49] Wiens, D. P.: Robust Weights and Designs for Biased Regression Models: Least Squares and Generalized M-Estimation, *Journal of Statistical Planning and Inference*, Vol. 83, No. 2, pp. 395–412 (2000).

[50] Kanamori, T. and Shimodaira, H.: Active Learning Algorithm Using the Maximum Weighted Log-Likelihood Estimator, *Journal of Statistical Planning and Inference*, Vol. 116, No. 1, pp. 149–162 (2003).

[51] Sugiyama, M.: Active Learning in Approximately Linear Regression Based on Conditional Expectation of Generalization Error, *Journal of Machine Learning Research*, Vol. 7, pp. 141–166 (2006).

[52] Sugiyama, M. and Nakajima, S.: Pool-based Active Learning in Approximate Linear Regression, *Machine Learning* (2009). to appear.

[53] Hachiya, H., Akiyama, T., Sugiyama, M. and Peters, J.: Adaptive Importance Sampling for Value Function Approximation in Off-policy Reinforcement Learning, *Neural Networks* (2009). to appear.

[54] Yamada, M., Sugiyama, M. and Matsui, T.: Covariate Shift Adaptation for Semi-supervised Speaker Identification, *Proceedings of 2009 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP2009)*, Taipei, Taiwan (2009). to appear.

[55] Storkey, A. and Sugiyama, M.: Mixture Regression for Covariate Shift, *Advances in Neural Information Processing Systems 19* (Schölkopf, B., Platt, J. C. and Hoffmann, T., eds.), Cambridge, MA, MIT Press, pp. 1337–1344 (2007).

[56] Brodsky, B. and Darkhovsky, B.: *Nonparametric Methods in Change-Point Problems*, Kluwer Academic Publishers, Dordrecht (1993).

[57] Kawahara, Y. and Sugiyama, M.: Change-point Detection in Time-series Data by Direct Density-ratio Estimation, *Proceedings of 2009 SIAM International Conference on Data Mining (SDM2009)*, Sparks, Nevada, USA (2009). to appear.

[58] Henkel, R. E.: *Tests of Significance*, SAGE Publication, Beverly Hills (1979).

[59] Takeuchi, I., Nomura, K. and Kanamori, T.: The Entire Solution Path of Kernel-based Nonparametric Conditional Quantile Estimator, *Proceedings of the International Joint Conference on Neural Networks*, pp. 153–158 (2006).

[60] Ali, S. M. and Silvey, S. D.: A General Class of Coefficients of Divergence of One Distribution from Another, *Journal of the Royal Statistical Society, Series B*, Vol. 28, No. 1, pp. 131–142 (1966).

[61] Csiszár, I.: Information-type Measures of Difference of Probability Distributions and Indirect Observation, *Studia Scientiarum Mathematicarum Hungarica*, Vol. 2, pp. 229–318 (1967).

[62] Fukumizu, K., Bach, F. R. and Jordan, M. I.: Kernel Dimension Reduction in Regression, *The Annals of Statistics* (2009). to appear.

[63] Lin, C.-J., Weng, R. C. and Keerthi, S. S.: Trust Region Newton Method for Large-Scale Logistic Regression, Technical report, Department of Computer Science, National Taiwan University (2007).

[64] Evgeniou, T., Pontil, M. and Poggio, T.: Regularization Networks and Support Vector Machines, *Advances in Computational Mathematics*, Vol. 13, No. 1, pp. 1–50 (2000).

[65] Rasmussen, C. E., Neal, R. M., Hinton, G. E., van Camp, D., Revow, M., Ghahramani, Z., Kustra, R. and Tibshirani, R.: The DELVE Manual (1996).

[66] Rätsch, G., Onoda, T. and Müller, K.-R.: Soft Margins for AdaBoost, *Machine Learning*, Vol. 42, No. 3, pp. 287–320 (2001).

[67] Karatzoglou, A., Smola, A., Hornik, K. and Zeileis, A.: An S4 Package for Kernel Methods in R, *Journal of Statistical Planning and Inference*, Vol. 11, No. 9, pp. 1–20 (2004).

[68] Fernandez, E. A.: The dprep Package, Technical report, University of Puerto Rico (2005).

[69] Bradley, A. P.: The Use of the Area under the ROC Curve in the Evaluation of Machine Learning Algorithms, *Pattern Recognition*, Vol. 30, No. 7, pp. 1145–1159 (1997).

[70] Fan, J., Yao, Q. and Tong, H.: Estimation of Conditional Densities and Sensitivity Measures in Nonlinear Dynamical Systems, *Biometrika*, Vol. 83, No. 1, pp. 189–206 (1996).

[71] Wolff, R. C. L., Yao, Q. and Hall, P.: Methods for Estimating a Conditional Distribution Function, *Journal of the American Statistical Association*, Vol. 94, No. 445, pp. 154–163 (1999).

[72] Li, Y., Liu, Y. and Zhu, J.: Quantile Regression in Reproducing Kernel Hilbert Spaces, *Journal of the American Statistical Association*, Vol. 102, No. 477, pp. 255–268 (2007).

[73] R Development Core Team: *The R Manuals* (2008). http://www.r-project.org.

[74] Darbellay, G. A. and Vajda, I.: Estimation of the Information by an Adaptive Partitioning of the Observation Space, *IEEE Transactions on Information Theory*, Vol. 45, No. 4, pp. 1315–1321 (1999).

[75] Wang, Q., Kulkarmi, S. R. and Verdú, S.: Divergence Estimation of Contiunous Distributions Based on Data-Dependent Partitions, *IEEE Transactions on Information Theory*, Vol. 51, No. 9, pp. 3064–3074 (2005).

[76] Silva, J. and Narayanan, S.: Universal Consistency of Data-Driven Partitions for Divergence Estimation, *Proceedings of IEEE International Symposium on Information Theory*, Nice, France, pp. 2021–2025 (2007).

[77] Khan, S., Bandyopadhyay, S., Ganguly, A. and Saigal, S.: Relative Performance of Mutual Information Estimation Methods for Quantifying the Dependence among Short and Noisy Data, *Physical Review E*, Vol. 76, p. 026209 (2007).

[78] Pérez-Cruz, F.: Kullback-Leibler Divergence Estimation of Continuous Distributions, *Proceedings of IEEE International Symposium on Information Theory*, Nice, France, pp. 1666–1670 (2008).