IEICE Transactions on Information and Systems,
vol.E98-D, no.5, pp.xxx–xxx, 2015.

Direct Density Ratio Estimation with Convolutional Neural Networks with Application in Outlier Detection

Hyunha Nam
Tokyo Institute of Technology
hyunha@sg.cs.titech.ac.jp

Masashi Sugiyama
The University of Tokyo
sugi@k.u-tokyo.ac.jp
http://www.ms.k.u-tokyo.ac.jp

Abstract

Recently, the ratio of probability density functions was demonstrated to be useful in solving various machine learning tasks such as outlier detection, non-stationarity adaptation, feature selection, and clustering. The key idea of this density ratio approach is that the ratio is directly estimated so that difficult density estimation is avoided. So far, parametric and non-parametric direct density ratio estimators with various loss functions have been developed, and the kernel least-squares method was demonstrated to be highly useful both in terms of accuracy and computational efficiency. On the other hand, recent study in pattern recognition exhibited that *deep* architectures such as a *convolutional neural network* can significantly outperform kernel methods. In this paper, we propose to use the convolutional neural network in density ratio estimation, and experimentally show that the proposed method tends to outperform the kernel-based method in outlying image detection.

Keywords

Density ratio estimation, Convolutional neural network, Outlier detection.

1 Introduction

Recently, it was shown [1] that, through the *ratio of probability density functions*, various statistical data analysis paradigms such as outlier detection, non-stationarity adaptation, mutual information estimation, and conditional probability estimation can be efficiently handled in a unified manner. The key idea of this density ratio approach is that, by directly estimating the density ratio, a difficult task of density estimation can be avoided.

So far, various density ratio estimators have been proposed. The simplest approach is to use *logistic regression* to discriminate samples from two distributions [2]. *Kernel mean matching* [3] uses the Hilbert-space embedding of probability distributions to directly approximate the values of the density ratio at data points. The *Kullback-Leibler importance estimation procedure* (KLIEP) fits a density ratio model to data under the log-loss [4, 5]. *Least-squares importance fitting* (LSIF) [6] uses the squared-loss to fit a density ratio model to data. Furthermore, all the above methods can be interpreted as fitting a density ratio model to data under the *Bregman divergence* [7].

Among these direct density ratio estimators, an unconstrained version of LSIF (uLSIF) with a kernel density-ratio model was demonstrated to be highly useful in terms of both accuracy [8] and computational efficiency [9]. For that reason, uLSIF-based machine learning algorithms have been successfully used in solving various machine learning tasks [10, 11, 12].

On the other hand, recent studies in pattern recognition demonstrated that *deep architecture* tends to perform better than kernel models [13]. In particular, a *convolutional neural network* (CNN) is demonstrated to be an excellent model of images [14, 15, 16, 17, 18], which is motivated by a biological brain [19].

The objective of this paper is to use the CNN model in density ratio estimation. To the best of our knowledge, this is the first attempt to apply deep learning to density ratio estimation, and we develop a gradient-based training algorithm under the squared-loss. We them apply the CNN-based density ratio estimator to *inlier-based outlier detection* [20, 21], and demonstrate that the proposed method outperforms existing approaches.

The remainder of this paper is organized as follows. We first review the kernel-based uLSIF method in Section 2. Then, we derive a density ratio estimation algorithm for CNNs in Section 3, and its experimental performance is investigated in Section 4. Finally, we conclude in Section 5.

2 Direct Density Ratio Estimation by uLSIF

In this section, we review the uLSIF method [22].

2.1 Problem Formulation

Suppose we are given independent and identically distributed training samples $\{\mathbf{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ from training distribution with density $p_{\mathrm{tr}}(\mathbf{x})$ and test samples $\{\mathbf{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$ from test distribution with density $p_{\mathrm{te}}(\mathbf{x})$ on some data domain $\mathcal{D} \subset \mathbb{R}^d$. The objective is to estimate the *density ratio*,

$$r(\mathbf{x}) = \frac{p_{\mathrm{tr}}(\mathbf{x})}{p_{\mathrm{te}}(\mathbf{x})},$$

from $\{\mathbf{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ and $\{\mathbf{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$.

A naive approach is to first separately estimate $p_{\mathrm{tr}}(\mathbf{x})$ from $\{\mathbf{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ and $p_{\mathrm{te}}(\mathbf{x})$ from $\{\mathbf{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$, and then compute the ratio of estimated densities. However, such a two-step approach does not perform well because the estimation error incurred in the first density estimation step can be magnified in the second step of computing their ratio [1]. Below, a direct density ratio estimator that does not involve density estimation is reviewed.

2.2 The uLSIF Criterion

Let $r_{\boldsymbol{\alpha}}(\mathbf{x})$ be a model of the density ratio $r(\mathbf{x})$, where $\boldsymbol{\alpha}$ denotes a parameter. The parameter $\boldsymbol{\alpha}$ is determined so that the following squared error is minimized:

$$\begin{aligned}J_0(\boldsymbol{\alpha}) &= \int \Big(r_{\boldsymbol{\alpha}}(\mathbf{x}) - r(\mathbf{x})\Big)^2 p_{\mathrm{te}}(\mathbf{x})\mathrm{d}\mathbf{x}\\ &= \int r_{\boldsymbol{\alpha}}(\mathbf{x})^2 p_{\mathrm{te}}(\mathbf{x})\mathrm{d}\mathbf{x} - 2\int r_{\boldsymbol{\alpha}}(\mathbf{x}) p_{\mathrm{tr}}(\mathbf{x})\mathrm{d}\mathbf{x}\\ &\quad + \int r(\mathbf{x}) p_{\mathrm{tr}}(\mathbf{x})\mathrm{d}\mathbf{x},\end{aligned}$$

where the last term is a constant so can be ignored. The first two terms are denoted by J:

$$J(\boldsymbol{\alpha}) = \int r_{\boldsymbol{\alpha}}(\mathbf{x})^2 p_{\mathrm{te}}(\mathbf{x})\mathrm{d}\mathbf{x} - 2\int r_{\boldsymbol{\alpha}}(\mathbf{x}) p_{\mathrm{tr}}(\mathbf{x})\mathrm{d}\mathbf{x},$$

which is empirically approximated by

$$\frac{1}{n_{\mathrm{te}}}\sum_{j=1}^{n_{\mathrm{te}}} r_{\boldsymbol{\alpha}}(\mathbf{x}_j^{\mathrm{te}})^2 - \frac{2}{n_{\mathrm{tr}}}\sum_{i=1}^{n_{\mathrm{tr}}} r_{\boldsymbol{\alpha}}(\mathbf{x}_i^{\mathrm{tr}}). \tag{1}$$

2.3 uLSIF for Kernel Model

Let us consider the following kernel density ratio model:

$$r_{\boldsymbol{\alpha}}(\mathbf{x}) = \sum_{\ell=1}^{n_{\mathrm{tr}}} \alpha_\ell K(\mathbf{x}, \mathbf{x}_\ell^{\mathrm{tr}}),$$

where $K(\mathbf{x}, \mathbf{x}')$ is a kernel function such as the *Gaussian kernel*:

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right) \quad \text{for } \sigma > 0.$$

Then the uLSIF criterion (1), enhanced with the ℓ_2-regularizer, can be expressed as

$$\widehat{J}(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^\top \widehat{\mathbf{G}}\boldsymbol{\alpha} - 2\widehat{\mathbf{h}}^\top \boldsymbol{\alpha} + \lambda\|\boldsymbol{\alpha}\|^2,$$

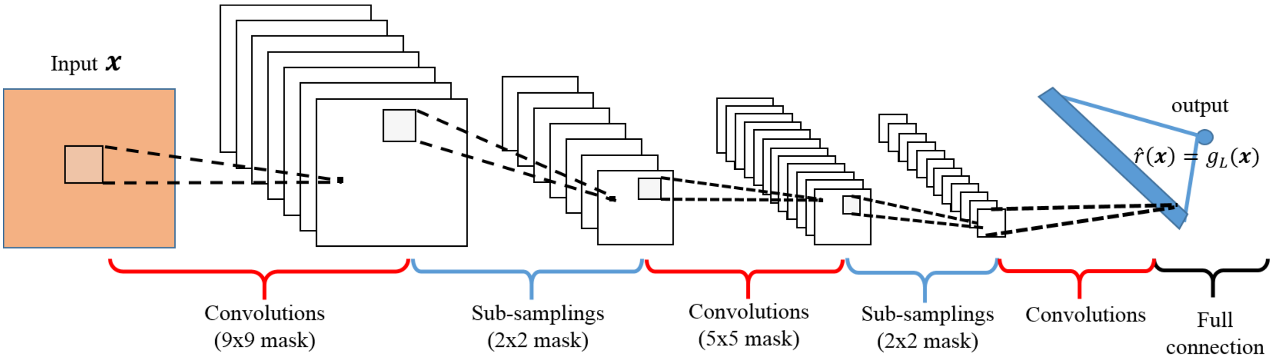

Figure 1: CNN.

where $\lambda > 0$ is the regularization parameter and

$$\widehat{G}_{\ell,\ell'} = \frac{1}{n_{\mathrm{te}}} \sum_{j=1}^{n_{\mathrm{te}}} K(\mathbf{x}_j^{\mathrm{te}}, \mathbf{x}_\ell^{\mathrm{te}}) K(\mathbf{x}_j^{\mathrm{te}}, \mathbf{x}_{\ell'}^{\mathrm{te}}),$$

$$\widehat{h}_\ell = \frac{1}{n_{\mathrm{tr}}} \sum_{i=1}^{n_{\mathrm{tr}}} K(\mathbf{x}_i^{\mathrm{tr}}, \mathbf{x}_\ell^{\mathrm{te}}).$$

Then the minimizer is given analytically as

$$\operatorname*{arg\,min}_{\boldsymbol{\alpha}} \widehat{J}(\boldsymbol{\alpha}) = (\widehat{\mathbf{G}} + \lambda \mathbf{I})^{-1} \widehat{\mathbf{h}},$$

where $\mathbf{I}$ denotes the identity matrix.

3 uLSIF for CNN

In this section, we apply the uLSIF criterion (1) to a *convolutional neural network* (CNN).

3.1 CNN

A CNN is a model for 2-dimensional images and consists of multiple layers (Figure 1): the input layer, alternate succession of convolution layers and sub-sampling layers, fully connected networks, and the output layer.

3.1.1 Convolution Layer

In the convolution layer, the output of the previous layer is convolved with a mask and put through the activation function to form the output feature map. More specifically, a matrix of an output feature map of the lth convolution layer $\mathbf{g}_l^u(\mathbf{x})$ for input feature maps

$\mathbf{g}_{l-1}^{v}(\mathbf{x})$ is given by

$$\begin{aligned}\mathbf{g}_l^u(\mathbf{x}) &= f(\mathbf{z}_l^u(\mathbf{x})),\\ \mathbf{z}_l^u(\mathbf{x}) &= \sum_{v\in\mathcal{M}_v} \mathbf{g}_{l-1}^{v}(\mathbf{x}) * \mathbf{k}_l^{uv} + b_l^u,\end{aligned}$$

where $\mathcal{M}_v$ is a selection of the input feature maps, f represents the sigmoid function $f(x) = (1+e^{-x})^{-1}$, $*$ denotes the convolution operator, b_l^u is a bias parameter, and $\mathbf{k}_l^{uv}$ denotes a mask.

3.1.2 Sub-Sampling Layer

The sub-sampling layer treats each feature map separately and produces sub-sampled versions of the input maps. More formally,

$$\mathbf{g}_l^u(\mathbf{x}) = \mathrm{down}(\mathbf{g}_{l-1}^{u}(\mathbf{x})) + b_l^u,$$

where “down” denotes a sub-sampling function and b_l^u is the bias parameter. The average value over the neighborhood is computed for each feature map in our method. This reduced-resolution output feature map is robust to variation and noise in the input feature map.

3.1.3 Fully-Connected Layer

The output $g_L(\mathbf{x}_j)$ of the fully-connected layer for input vector $\mathbf{g}_{L-1}(\mathbf{x}_j)$ is represented as

$$\begin{aligned}g_L(\mathbf{x}) &= h(z_L(\mathbf{x})),\\ z_L(\mathbf{x}) &= \mathbf{W}_L\mathbf{g}_{L-1}(\mathbf{x}) + b_L,\end{aligned}$$

where L is the last layer, $\mathbf{W}_L$ is a connection parameter, b_L is the bias parameter and $\mathbf{g}_{L-1}(\mathbf{x})$ is a reshaped output of the last convolution layer. h is the *softplus* function $h(x) = \log(1+e^x)$, which is a smooth approximation to the *rectifier* $\max(0,x)$ [23].

3.2 Density Ratio Estimation with CNN

The kernel-based uLSIF is a computationally very efficient method. Using a deep CNN, however, would cast huge demand to the computation power during the training process. To economize on the computation cost at every iteration, stochastic gradient descent algorithm is used in out implementation. Thus we train the CNN model with the uLSIF criterion (1) by the *stochastic gradient method* for a pair of samples $(\mathbf{x}^{\mathrm{te}}, \mathbf{x}^{\mathrm{tr}})$ to obtain a local minimizer:

$$\widetilde{J}(\{\mathbf{W}_L\},\{b_L\}) = g_L(\mathbf{x}^{\mathrm{te}})^2 - 2g_L(\mathbf{x}^{\mathrm{tr}}). \tag{2}$$

The gradients of $\widetilde{J}$ with respect to $\mathbf{W}_L$ and b_L are given by

$$\frac{\partial \widetilde{J}}{\partial \mathbf{W}_L} = \delta_L(\mathbf{x}^{\mathrm{te}})(\mathbf{g}_{L-1}(\mathbf{x}^{\mathrm{te}}))^\top - \delta_L(\mathbf{x}^{\mathrm{tr}})(\mathbf{g}_{L-1}(\mathbf{x}^{\mathrm{tr}}))^\top,$$
$$\frac{\partial \widetilde{J}}{\partial b_L} = \delta_L(\mathbf{x}^{\mathrm{te}}) - \delta_L(\mathbf{x}^{\mathrm{tr}}),$$

where

$$\delta_L(\mathbf{x}^{\mathrm{te}}) = 2g_L(\mathbf{x}^{\mathrm{te}})h'(z_L(\mathbf{x}^{\mathrm{te}})),$$
$$\delta_L(\mathbf{x}^{\mathrm{tr}}) = 2h'(z_L(\mathbf{x}^{\mathrm{tr}})).$$

The gradient of the bias for a feature map of the lth convolution layer is given by summing all the elements in each error term:

$$\frac{\partial \widetilde{J}}{\partial b_l^u} = \sum_{n,m} (\boldsymbol{\delta}_l^u(\mathbf{x}^{te}))_{n,m} - (\boldsymbol{\delta}_l^u(\mathbf{x}^{tr}))_{n,m}.$$

The gradient for the mask weight of the lth convolution layer is given by

$$\frac{\partial \widetilde{J}}{\partial \mathbf{k}_l^{uv}} = \sum_{n,m} (\boldsymbol{\delta}_l^u(\mathbf{x}^{te}))_{n,m}(\mathbf{M}_l^v(\mathbf{x}^{te}))_{n,m} - (\boldsymbol{\delta}_l^u(\mathbf{x}^{tr}))_{n,m}(\mathbf{M}_l^v(\mathbf{x}^{tr}))_{n,m},$$

where

$$\mathbf{M}_l^v(\mathbf{x}) = \mathbf{g}_{l-1}^v(\mathbf{x}) \bullet \mathbf{k}_l^{uv},$$

"$\bullet$" denotes the element-wise product, and the error terms in the convolution layer are expressed as

$$\boldsymbol{\delta}_l^u(\mathbf{x}^{te}) = f'(\mathbf{z}_l^u(\mathbf{x}^{te})) \bullet \mathrm{up}(\boldsymbol{\delta}_{l+1}^u(\mathbf{x}^{te})),$$
$$\boldsymbol{\delta}_l^u(\mathbf{x}^{tr}) = f'(\mathbf{z}_l^u(\mathbf{x}^{tr})) \bullet \mathrm{up}(\boldsymbol{\delta}_{l+1}^u(\mathbf{x}^{tr})).$$

Here, "up" denotes the up-sampling operation, which can be implemented with the Kronecker product. At the last convolution layer, an error term includes connection parameters of the fully connected layer:

$$\boldsymbol{\delta}_l^u(\mathbf{x}^{te}) = \mathbf{W}_L\left(f'(\mathbf{z}_l^u(\mathbf{x}^{te})) \bullet \mathrm{up}(\boldsymbol{\delta}_{l+1}^u(\mathbf{x}^{te}))\right),$$
$$\boldsymbol{\delta}_l^u(\mathbf{x}^{tr}) = \mathbf{W}_L\left(f'(\mathbf{z}_l^u(\mathbf{x}^{tr})) \bullet \mathrm{up}(\boldsymbol{\delta}_{l+1}^u(\mathbf{x}^{tr}))\right).$$

The gradient of the bias for the lth sub-sampling layer is given by

$$\frac{\partial \widetilde{J}}{\partial b_l^u} = \sum_{n,m} (\boldsymbol{\delta}_l^u(\mathbf{x}^{te}))_{n,m} - (\boldsymbol{\delta}_l^u(\mathbf{x}^{tr}))_{n,m},$$

where

$$\boldsymbol{\delta}_l^u(\mathbf{x}^{te}) = \boldsymbol{\delta}_{l+1}^u(\mathbf{x}^{te}) * \mathrm{rot}(\mathbf{k}_{l+1}^u),$$
$$\boldsymbol{\delta}_l^u(\mathbf{x}^{tr}) = \boldsymbol{\delta}_{l+1}^u(\mathbf{x}^{tr}) * \mathrm{rot}(\mathbf{k}_{l+1}^u).$$

Here "rot" denotes the rotation operator which rotates the mask by 180 degrees.

4 Experiments

In this section, the proposed CNN-based uLSIF is compared with the kernel-based uLSIF and the kernel-based KLIEP in inlier-based outlier detection. Note that, in [20], the kernel-based uLSIF and KLIEP were demonstrated to outperform other outlier detection methods such as the *one-class support vector machine* [24] and the *local outlier factor* [25].

4.1 Inlier-Based Outlier Detection

The objective of inlier-based outlier detection is to find outliers in a test data set $\{\mathbf{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$ given a training data set $\{\mathbf{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ which are known to be inliers. Here, following [20], the problem of inlier-based outlier detection is formulated as the problem of estimating the density ratio,

$$r(\mathbf{x}) = \frac{p_{\mathrm{tr}}(\mathbf{x})}{p_{\mathrm{te}}(\mathbf{x})},$$

from $\{\mathbf{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ and $\{\mathbf{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$. Given that outliers tend to exist in regions with small inlier density $p_{\mathrm{tr}}(\mathbf{x})$, the density ratio $r(\mathbf{x})$ tends to take a small value if $\mathbf{x}$ is an outlier. Outliers are hard to universally define. By contrast, inlier samples are often stable and available abundantly in practice. Therefore the setting of inlier-based outlier detection would be more practical than the (semi) supervised setting.

4.2 Experimental Setup

We use a CNN with 5 layers, i.e., the first 4 layers contain two alternate convolution layers and sub-sampling layers, followed by an additional convolution layer for making a vector input which is given to the final fully-connected layer (see Figure 1 again). The size of input images depends on the data set. The first layer has 6 convolution masks of size 9 × 9 with a stride of a pixel for image of size 32 × 32 (for smaller images, we use 5 × 5 convolution masks). In the following sub-sampling layer, each unit calculates the average over the output from neighbors of 2 × 2 pixels. The sub-sampling layer reduces the resolution of output of the previous layer and the reduced feature is robust to small variations. The second convolution layer has 12 convolution masks of size 5×5 (for smaller images, we use 3×3 convolution masks). Sub-sampling is applied to its output again. The third convolution layer reshapes 12 feature maps generated by the previous sub-sampling layer and provide the vector input to the final fully-connected layer. We use the sigmoid activation function for convolution layers and the ReLU activation function for the last layer. At the learning stage, all parameters are optimized through the stochastic gradient method. We use the constant learning rate 0.1 for all layers. All the model parameters such as the size of the convolution mask, the number of layers and learning rate are tuned in the same way as the previous studies [26], [27].

Table 1: Computation time of USPS dataset (8 and 3)

Method	uLSIF (CNN)	uLSIF (kernel)	KLIEP (kernel)
Computation time (sec)	833.4969	0.0028	0.0164

We use the MNIST handwritten digit dataset, the USPS handwritten digit dataset, the PIE face image dataset, and the CIFAR-10 image classification dataset. We select one class (e.g., `airplane` in CIFAR-10) and all training samples is this class are regarded as a training set in inlier-based outlier detection. Then all evaluation samples of a selected class and a fraction ρ of evaluation samples from another class (e.g., `bird`) are regarded as a test set in inlier-based outlier detection. Namely, we regard samples in a selected class as inliers and samples in another class as outliers. The *area under the ROC curve* (AUC) value is used as an error metric. Kernel-based uLSIF and KLIEP were designed to use selected 100 test input points as Gaussian centers randomly.

4.3 Results

Experimental results are exhibited below.

4.3.1 MNIST and USPS

The MNIST dataset contains hand-written digit images in gray-scale: a training set of 60,000 images and a test set of 10,000 images. Each image consists of 1,024 (= 32 × 32) pixels. The USPS hand-written digit dataset contains 9,298 gray-scale images of size 16 × 16 which are spilt into a training set of 7,291 images and a test set of 2,007 images. Both datasets have 10 class labels which are integers between 0 and 9.

The mean AUC values and the standard deviations over 20 trials for MNIST and USPS are summarized in Table 2, and the mean MSE (mean squared error) values and the standard deviations over 20 trials for MNIST and USPS are shown in Table 3. Figure 2 depicts the ROC curves for the experiment with inlier class 9 and outlier class 8. The results show that the proposed CNN-based uLSIF works significantly better than kernel-based KLIEP and uLSIF.

Table 1 shows the computation time of each algorithm for the USPS dataset (8 and 3). As we mentioned above, kernel-based uLSIF has the best performance in terms of computational efficiency.

Mean AUC values (with standard deviations in parentheses) over 50 trials for the PASCAL VOC dataset. The best method in terms of the mean AUC and comparable methods according to the *t-test* at the significance level 5% are specified by bold face.

4.3.2 PIE

Next, we consider outlier detection on the PIE face image dataset. Face images contain more complex objects than digits and data variability caused by posing variations, illu-

Table 2: Mean AUC values and the standard deviations over 20 trials for MNIST and USPS. The best method in terms of the mean AUC and comparable methods according to the *t-test* at the significance level 5% are specified by bold face.

Dataset (Inlier & outlier)		ρ	uLSIF (CNN)	uLSIF (kernel)	KLIEP (kernel)
MNIST	4 & 9	0.01	0.86 ± 0.12	0.64 ± 0.00	0.82 ± 0.00
		0.05	**0.89 ± 0.01**	0.65 ± 0.04	0.79 ± 0.06
	2 & 5	0.01	0.95 ± 0.05	0.79 ± 0.00	0.95 ± 0.00
		0.05	**0.98 ± 0.01**	0.91 ± 0.01	0.97 ± 0.00
	8 & 3	0.01	**0.97 ± 0.02**	0.91 ± 0.00	0.94 ± 0.00
		0.05	**0.97 ± 0.01**	0.83 ± 0.01	0.91 ± 0.00
	9 & 8	0.01	**0.97 ± 0.05**	0.88 ± 0.00	0.92 ± 0.00
		0.05	**0.97 ± 0.01**	0.71 ± 0.02	0.87 ± 0.00
USPS	4 & 9	0.03	**0.87 ± 0.11**	0.53 ± 0.00	0.73 ± 0.00
		0.05	**0.95 ± 0.13**	0.68 ± 0.03	0.82 ± 0.01
	2 & 5	0.03	0.97 ± 0.11	0.91 ± 0.00	0.93 ± 0.00
		0.05	0.99 ± 0.02	0.94 ± 0.01	0.98 ± 0.00
	8 & 3	0.03	**0.95 ± 0.04**	0.73 ± 0.00	0.77 ± 0.00
		0.05	**0.91 ± 0.07**	0.67 ± 0.02	0.83 ± 0.02
	9 & 8	0.03	0.91 ± 0.14	0.92 ± 0.00	0.94 ± 0.00
		0.05	**0.95 ± 0.06**	0.59 ± 0.02	0.80 ± 0.03

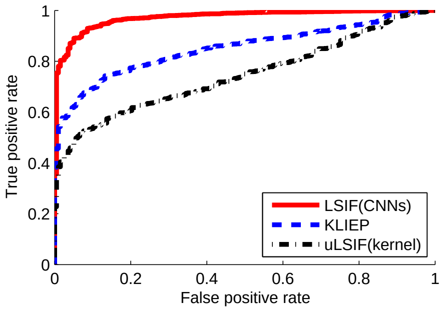

Figure 2: ROC curves of MNIST 9 and 8 ($\rho = 0.05$).

mination conditions, and facial expressions is higher. These facts make outlier detection more challenging for the PIE dataset. The original PIE dataset contains 41,368 color face images with 68 individuals, each person is under 13 different poses, 43 different illumination conditions, and with 4 different expressions. We use the processed PIE data for face recognition [28], which chooses five near-frontal poses and uses all the images under different illuminations and expressions; thus 170 images are given for each individual. We allocate randomly chosen 5,780 images for training samples and the remaining 5,780 images for test samples in this experiment. All the images are manually aligned and cropped. The cropped images are 32 × 32 pixels with 256 gray-levels per pixel.

Table 3: Mean MSE values over 20 trials for MNIST and USPS. The best method in terms of the mean MSE and comparable methods according to the *t-test* at the significance level 5% are specified by bold face.

Dataset (Inlier & outlier)		ρ	uLSIF (CNN)	uLSIF (kernel)	KLIEP (kernel)
MNIST	4 & 9	0.01	-0.50 ± 0.00	-0.50 ± 0.00	-0.50 ± 0.00
		0.05	**-0.51 ± 0.00**	-0.50 ± 0.00	-0.50 ± 0.00
	2 & 5	0.01	-0.50 ± 0.00	-0.50 ± 0.00	-0.50 ± 0.00
		0.05	-0.50 ± 0.00	-0.50 ± 0.00	-0.50 ± 0.00
	8 & 3	0.01	-0.50 ± 0.00	-0.50 ± 0.00	-0.50 ± 0.00
		0.05	**-0.52 ± 0.00**	-0.51 ± 0.01	-0.51 ± 0.00
	9 & 8	0.01	**-0.50 ± 0.00**	-0.50 ± 0.00	-0.50 ± 0.00
		0.05	**-0.51 ± 0.00**	-0.50 ± 0.01	-0.50 ± 0.00
USPS	4 & 9	0.03	-0.49 ± 0.02	-0.50 ± 0.00	-0.50 ± 0.00
		0.05	-0.50 ± 0.02	-0.50 ± 0.00	-0.50 ± 0.00
	2 & 5	0.03	-0.49 ± 0.04	-0.50 ± 0.00	-0.50 ± 0.00
		0.05	-0.51 ± 0.01	-0.50 ± 0.00	-0.50 ± 0.00
	8 & 3	0.03	-0.49 ± 0.04	-0.50 ± 0.00	-0.50 ± 0.00
		0.05	-0.50 ± 0.03	-0.50 ± 0.00	-0.50 ± 0.00
	9 & 8	0.03	-0.49 ± 0.00	-0.50 ± 0.00	-0.50 ± 0.00
		0.05	-0.50 ± 0.02	-0.50 ± 0.00	-0.50 ± 0.00

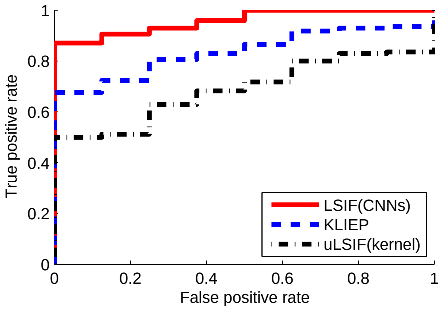

Figure 3: ROC curves of PIE 7 and 33 ($\rho = 0.05$).

The mean AUC values and the standard deviations over 20 trials are summarized in Table 4, and Figure 3 depicts the ROC curves for the experiment with inlier class 7 and outlier class 33. The results show that our proposed method achieves better performance for all classes than the kernel-based KLIEP and uLSIF.

4.3.3 CIFAR-10

Finally, we use the CIFAR-10 image classification dataset, which consists of 32 × 32 color images in 10 classes (e.g., `car`, `cat`, `dog`, and `ship`) with 6,000 images per class.

Table 4: Mean AUC values and the standard deviations over 20 trials for PIE. The best method in terms of the mean AUC and comparable methods according to the *t-test* at the significance level 5% are specified by bold face.

PIE (Inlier & outlier)	ρ	uLSIF (CNNs)	uLSIF (kernel)	KLIEP (kernel)
1 & 12	0.03	**0.97 ± 0.02**	0.92 ± 0.00	0.96 ± 0.00
	0.05	**0.96 ± 0.02**	0.86 ± 0.00	0.93 ± 0.00
2 & 58	0.03	**0.85 ± 0.02**	0.53 ± 0.00	0.75 ± 0.00
	0.05	**0.81 ± 0.08**	0.37 ± 0.00	0.66 ± 0.00
4 & 35	0.03	0.86 ± 0.04	0.81 ± 0.00	0.85 ± 0.00
	0.05	0.69 ± 0.07	0.42 ± 0.00	0.53 ± 0.00
7 & 33	0.03	**0.90 ± 0.04**	0.64 ± 0.00	0.82 ± 0.00
	0.05	**0.94 ± 0.05**	0.65 ± 0.00	0.84 ± 0.00
12 & 62	0.03	0.56 ± 0.21	0.57 ± 0.00	0.65 ± 0.00
	0.05	0.67 ± 0.18	0.60 ± 0.00	0.63 ± 0.00
18 & 28	0.03	0.80 ± 0.07	0.62 ± 0.00	0.77 ± 0.00
	0.05	**0.81 ± 0.05**	0.51 ± 0.00	0.67 ± 0.00
19 & 32	0.03	**0.85 ± 0.02**	0.69 ± 0.00	0.83 ± 0.00
	0.05	0.84 ± 0.05	0.59 ± 0.00	0.76 ± 0.00
21 & 30	0.03	0.71 ± 0.12	0.51 ± 0.00	0.69 ± 0.00
	0.05	**0.95 ± 0.10**	0.66 ± 0.00	0.84 ± 0.00
24 & 37	0.03	0.91 ± 0.04	0.86 ± 0.00	0.90 ± 0.00
	0.05	**0.92 ± 0.03**	0.84 ± 0.00	0.89 ± 0.00
21 & 47	0.03	0.65 ± 0.13	0.51 ± 0.00	0.63 ± 0.00
	0.05	**0.67 ± 0.13**	0.33 ± 0.00	0.36 ± 0.00
40 & 11	0.03	0.81 ± 0.05	0.81 ± 0.00	0.83 ± 0.00
	0.05	0.86 ± 0.07	0.83 ± 0.00	0.85 ± 0.00

Since images in CIFAR-10 contain more data variability and noise than those in the PIE dataset, outlier detection is expected to be even harder. There are 50,000 training images and 10,000 test images, and we convert color images to gray-scale.

The mean AUC values and the standard deviations over 20 trials are summarized in Table 5, and Figure 4 depicts the ROC curves for the experiment with inlier class `car` and outlier class `cat`. The results show that the proposed CNN-based uLSIF performs excellently.

5 Discussion and Conclusion

We proposed to use CNNs in least-squares direct density-ratio estimation, uLSIF, and demonstrated its usefulness in inlier-based outlier detection of images.

In order to investigate the change in AUC values for different number of kernel func-

Table 5: Mean AUC values and the standard deviations over 20 trials for CIFAR-10. The best method in terms of the mean AUC and comparable methods according to the *t-test* at the significance level 5% are specified by bold face.

CIFAR (Inlier & outlier)	ρ	uLSIF (CNN)	uLSIF (kernel)	KLIEP (kernel)
Car and cat	0.03	**0.71 ± 0.06**	0.57 ± 0.01	0.65 ± 0.00
	0.05	**0.79 ± 0.03**	0.50 ± 0.01	0.52 ± 0.00
Airplane and bird	0.03	**0.63 ± 0.03**	0.51 ± 0.00	0.61 ± 0.00
	0.05	**0.65 ± 0.05**	0.50 ± 0.01	0.61 ± 0.00
Truck and car	0.03	0.58 ± 0.06	0.45 ± 0.02	**0.66 ± 0.00**
	0.05	0.66 ± 0.02	0.51 ± 0.00	0.65 ± 0.00
Ship and airplane	0.03	**0.64 ± 0.03**	0.61 ± 0.00	0.61 ± 0.00
	0.05	0.66 ± 0.02	0.49 ± 0.00	0.64 ± 0.00
Dog and frog	0.03	0.78 ± 0.08	0.59 ± 0.03	0.74 ± 0.00
	0.05	0.77 ± 0.10	0.64 ± 0.02	0.73 ± 0.01

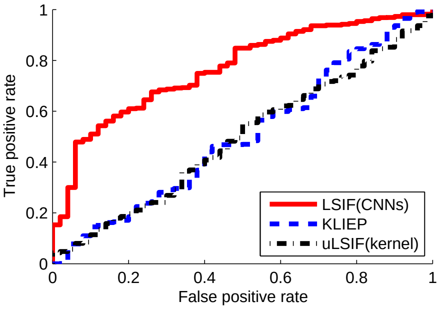

Figure 4: ROC curves of CIFAR-10 `car` and `cat` ($\rho = 0.05$).

tion, we executed uLSIF and KLIEP with different number of kernel centers, for the CIFAR-10 dataset (`car` and `cat`). As shown in Figure 6. This shows that using 100 kernels is reasonable in terms of both accuracy and computation time.

Multi-layer structure of CNNs gives us a chance to look into the outlier detection process. Figure 5 displays the first convolution masks of size 9×9 learned by the proposed method for the MNIST experiment with (a) inlier class 4 and outlier class 9 and (b) inlier class 8 and outlier class 3. In (a), masks have oblique and rectilinear patterns and they seem like a part of digit 4. In contrast, in (b), masks have U-shaped and turned U-shaped patterns, which are rarely found in outlier class 3. Thus, the masks have learned to detect inliers' edges and lines at different positions and angles in the images, implying that our algorithm is translation-invariant.

We used uLSIF for training CNNs, but there are many other density ratio estimators [1]. Developing algorithms for training CNN with other density ratio estimation approaches is an important future work.

We focused on inlier-based outlier detection in this paper, but we believe that the

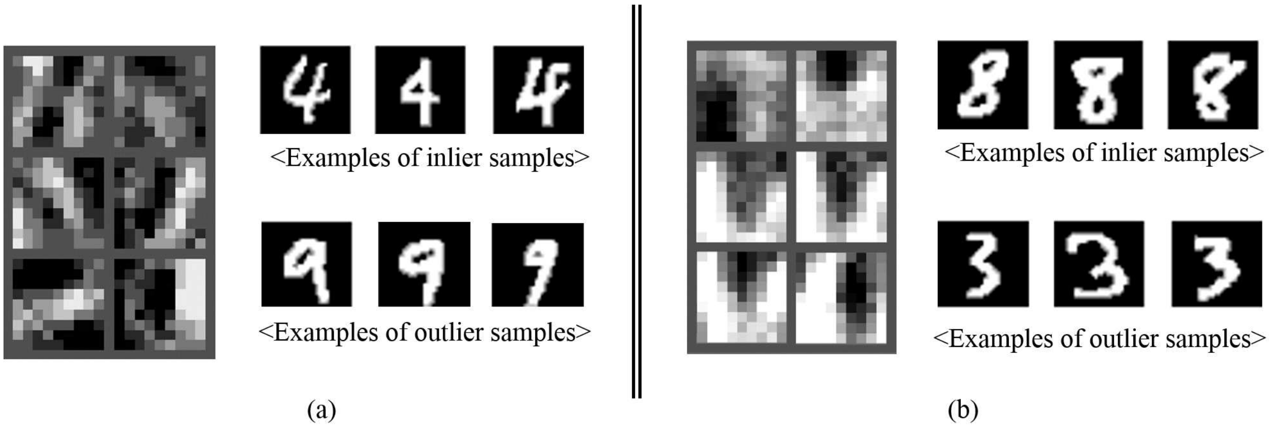

Figure 5: The first convolution layer masks.
Left (a) depicts first convolution layer masks of MNIST experiments of class 4 and 9. class 4 is inlier class.
Right (b) depicts first convolution layer masks of MNIST experiments of class 8 and 3. class 8 is inlier class.

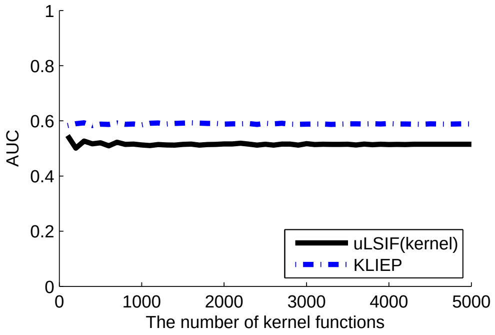

Figure 6: Change in AUC values for different number of kernel functions (CIFAR-10 `car` and `cat`)

proposed method is also useful in other machine learning tasks that can be tacked via density ratio estimation [1]. This is another important future work.

References

[1] M. Sugiyama, T. Suzuki, and T. Kanamori, Density Ratio Estimation in Machine Learning, Cambridge University Press, Cambridge, UK, 2012.

[2] S. Bickel, M. Brückner, and T. Scheffer, "Discriminative learning for differing training and test distributions," Proceedings of the 24th international conference on Machine learning, pp.81–88, ACM, 2007.

[3] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, "Covariate shift by kernel mean matching," Dataset Shift in Machine Learning, ed. J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, and N. Lawrence, Cambridge, MA, USA, pp.131–160, MIT Press, 2009.

[4] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe, "Direct importance estimation for covariate shift adaptation," Annals of the Institute of Statistical Mathematics, vol.60, no.4, pp.699–746, 2008.

[5] X. Nguyen, M.J. Wainwright, and M.I. Jordan, "Estimating divergence functionals and the likelihood ratio by convex risk minimization," IEEE Transactions on Information Theory, vol.56, no.11, pp.5847–5861, 2010.

[6] T. Kanamori, S. Hido, and M. Sugiyama, "A least-squares approach to direct importance estimation," Journal of Machine Learning Research, vol.10, pp.1391–1445, Jul. 2009.

[7] M. Sugiyama, T. Suzuki, and T. Kanamori, "Density ratio matching under the Bregman divergence: A unified framework of density ratio estimation," Annals of the Institute of Statistical Mathematics, vol.64, no.5, pp.1009–1044, 2012.

[8] T. Kanamori, T. Suzuki, and M. Sugiyama, "Statistical analysis of kernel-based least-squares density-ratio estimation," Machine Learning, vol.86, no.3, pp.335–367, 2012.

[9] T. Kanamori, T. Suzuki, and M. Sugiyama, "Computational complexity of kernel-based density-ratio estimation: A condition number analysis," Machine Learning, vol.90, no.3, pp.431–460, 2013.

[10] M. Sugiyama, "Machine learning with squared-loss mutual information," Entropy, vol.15, no.1, pp.80–112, 2013.

[11] M. Sugiyama, S. Liu, M.C. du Plessis, M. Yamanaka, M. Yamada, T. Suzuki, and T. Kanamori, "Direct divergence approximation between probability distributions and its applications in machine learning," Journal of Computing Science and Engineering, vol.7, no.2, pp.99–111, 2013.

[12] M. Sugiyama, M. Yamada, and M.C. du Plessis, "Learning under non-stationarity: Covariate shift and class-balance change," WIREs Computational Statistics, vol.5, no.6, pp.465–477, 2013.

[13] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," Proceedings of the 24th international conference on Machine learning, pp.473–480, ACM, 2007.

[14] A. Krizhevsky, I. Sutskever, and G.E. Hinton, “Imagenet classification with deep convolutional neural networks,” Advances in neural information processing systems, pp.1097–1105, 2012.

[15] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, “What is the best multi-stage architecture for object recognition?,” Computer Vision, 2009 IEEE 12th International Conference on, pp.2146–2153, IEEE, 2009.

[16] H. Lee, R. Grosse, R. Ranganath, and A.Y. Ng, “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations,” Proceedings of the 26th Annual International Conference on Machine Learning, pp.609–616, ACM, 2009.

[17] S.C. Turaga, J.F. Murray, V. Jain, F. Roth, M. Helmstaedter, K. Briggman, W. Denk, and H.S. Seung, “Convolutional networks can learn to generate affinity graphs for image segmentation,” Neural Computation, vol.22, no.2, pp.511–538, 2010.

[18] J. Ngiam, Z. Chen, D. Chia, P.W. Koh, Q.V. Le, and A.Y. Ng, “Tiled convolutional neural networks,” Advances in Neural Information Processing Systems, pp.1279–1287, 2010.

[19] K. Fukushima, “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position,” Biological Cybernetics, vol.36, no.4, pp.93–202, 1980.

[20] S. Hido, Y. Tsuboi, H. Kashima, M. Sugiyama, and T. Kanamori, “Statistical outlier detection using direct density ratio estimation,” Knowledge and Information Systems, vol.26, no.2, pp.309–336, 2011.

[21] A. Smola, L. Song, and C.H. Teo, “Relative novelty detection,” Proceedings of Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS2009), ed. D. van Dyk and M. Welling, JMLR Workshop and Conference Proceedings, vol.5, Clearwater Beach, FL, USA, pp.536–543, 2009.

[22] T. Kanamori, S. Hido, and M. Sugiyama, “A least-squares approach to direct importance estimation,” The Journal of Machine Learning Research, vol.10, pp.1391–1445, 2009.

[23] V. Nair and G.E. Hinton, “Rectified linear units improve restricted boltzmann machines,” Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp.807–814, 2010.

[24] B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, and R.C. Williamson, “Estimating the support of a high-dimensional distribution,” Neural computation, vol.13, no.7, pp.1443–1471, 2001.

[25] M.M. Breunig, H.P. Kriegel, R.T. Ng, and J. Sander, "Lof: Identifying density-based local outliers," SIGMOD Rec., vol.29, no.2, pp.93–104, May 2000.

[26] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol.86, no.11, pp.2278–2324, 1998.

[27] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on, pp.253–256, IEEE, 2010.

[28] X. He, S. Yan, Y. Hu, P. Niyogi, and H.J. Zhang, "Face recognition using laplacianfaces," IEEE Trans. Pattern Anal. Mach. Intelligence, vol.27, no.3, pp.328–340, 2005.