Direct Density Ratio Estimation for Large-scale Covariate Shift Adaptation

Yuta Tsuboi (*), Hisashi Kashima, Shohei Hido,
IBM Research, Tokyo Research Laboratory
* Nara Institute of Science and Technology
Steffen Bickel,
Max Planck Institute for Computer Science
Masashi Sugiyama
Tokyo Institute of Technology, Department of Computer Science
# Table of Contents

1. Motivation
   - Covariate shift situation
   - Direct density estimation methods

2. Proposed Method
   - LL-KLiep: Log-linear model for KLiep
     - Log-linear model: natural modeling for density ratio function
     - Standard optimization techniques to learn a density ratio function
   - LL-KLiep(LS): Another optimization technique for LL-KLiep
     - For applications with large numbers of unlabeled test inputs
Covariate shift situation
Training and test inputs x follow different distributions

- Input distribution changes:
  \[ p_{\text{train}}(x) \neq p_{\text{test}}(x) \]

- Functional relation remains unchanged:
  \[ p_{\text{train}}(y | x) = p_{\text{test}}(y | x) \]

Labeled training inputs from \( p_{\text{train}}(x) \)

Unlabeled test inputs from \( p_{\text{test}}(x) \)

Classification under Covariate Shift
Examples of covariate shift situation
Domain Adaptation & Selective Sampling (Active Learning)

- **Domain adaptation of statistical classifiers**
  - The data distribution in the test domain is different from that in the training domain. (Note: the functional relation can also be changed)
  - E.g.: Spam filters can be trained on public collections of spam, but are applied to an individual person’s inbox. (Personalization)

- **Selective sampling (active learning) of statistical classifiers**
  - The learning algorithm can actively query the teacher for labels.
  - Since the learner chooses the examples by design, the data distribution of the labeled training examples is different from that of a sample pool.
A common approach for covariate shift situation
Weighting the training examples by importance.

- **Density ratio (importance):**
  \[ w(x) = \frac{p_{test}(x)}{p_{train}(x)}. \]

- **Example: Importance Weighted Logistic Regression (IWLR)**
  - Weighted Log-likelihood for Logistic Regression (LR)
  \[
  \int \int p_{test}(x)p(y \mid x)p_0(y \mid x) \, dx \, dy \\
  = \int \int p_{train}(x) \frac{p_{test}(x)}{p_{train}(x)} p(y \mid x) \log p_0(y \mid x) \, dx \, dy \\
  \approx \sum_{(x,y) \in Z_{train}} w(x)(y f_\theta(x) - \log(1 + \exp(y f_\theta(x)))).
  \]

LR model

Labeled training inputs

Training data

Density ratio (Importance)

Log-likelihood

w(x) < 1

w(x) > 1
We need to estimate the density ratio from samples. Importance Estimation

- **Problem setting:** i.i.d. training and test samples are given

  Training inputs: \( D_{tr} = \{x_i\}_{i=1}^{N_{tr}} \) from \( P_{\text{train}}(x) \)

  Test inputs: \( D_{te} = \{x_i\}_{i=1}^{N_{te}} \) from \( P_{\text{test}}(x) \)

- Naïve approach: estimate \( P_{\text{train}}(x) \) and \( P_{\text{test}}(x) \) separately, and take the ratio of the density estimates

- However, density \( P(x) \) estimation is the hard problem particularly in high dimensional cases.
Table of Contents

1. Motivation
   - Covariate shift situation
   - Direct density estimation methods

2. Proposed Method
   - LL-KLIEP: Log-linear model for KLIEP
     - Log-linear model: natural modeling for density ratio function
     - Standard optimization techniques to learn a density ratio function
   - LL-KLIEP(LS): Another optimization technique for LL-KLIEP
     - For applications with large numbers of unlabeled test inputs
Modeling Density ratio by Log-linear Model

- We use a log-linear model:
  \[
  \hat{w}(\mathbf{x}) = \frac{\exp(\langle \alpha, \psi(\mathbf{x}) \rangle)}{\frac{1}{N_{tr}} \sum_{\mathbf{x}' \in D_{tr}} \exp(\langle \alpha, \psi(\mathbf{x}') \rangle)}
  \]
  \(\alpha\) : model parameter
  \(\psi(\mathbf{x})\) : basis function
  \(<,>\) : inner product
  Training data set

- \(\hat{w}(\mathbf{x})\) takes only non-negative values.

  \(\Rightarrow\) **natural modeling for ratio** (\(\alpha\) and \(\psi(\mathbf{x})\) can be an arbitrary value)

- The denominator guarantees \(\hat{p}_{test}(\mathbf{x})\) be a probability density function

- **Test density** is approximated by
  \[
  \hat{p}_{te}(\mathbf{x}) = p_{tr}(\mathbf{x}) \hat{w}(\mathbf{x})
  \]

- Learn \(\alpha\) so that \(\hat{p}_{test}(\mathbf{x})\) approximates \(p_{test}(\mathbf{x})\) well.
Kullback—Leibler (KL) Divergence

- Minimize KL divergence between \( p_{\text{test}}(\mathbf{x}) \) and \( \hat{p}_{\text{test}}(\mathbf{x}) \):

\[
\arg\min_{\alpha} KL[p_{\text{test}}(\mathbf{x}) \| \hat{p}_{\text{test}}(\mathbf{x})] = \int p_{\text{test}}(\mathbf{x}) \log \frac{p_{\text{test}}(\mathbf{x})}{\hat{p}_{\text{test}}(\mathbf{x})} d\mathbf{x} - \int p_{\text{test}}(\mathbf{x}) \log \hat{w}(\mathbf{x}) d\mathbf{x}
\]

\[
\hat{p}_{\text{test}}(\mathbf{x}) = p_{\text{train}}(\mathbf{x}) \hat{w}(\mathbf{x})
\]

\( KL[p_{\text{test}}(\mathbf{x}) \| \hat{p}_{\text{test}}(\mathbf{x})] \) is constant and relevant.
Kullback-Leibler Importance Estimation Procedure (KLIEP) for Log-linear Models: LL-KLIEP

- Thus, \( \arg \min_{\alpha} KL[p_{\text{test}}(x) \| \hat{p}_{\text{test}}(x)] \)

  \( \Leftrightarrow \arg \max_{\alpha} \int p_{\text{test}}(x) \log \hat{w}(x) \, dx \)

- Empirical approximation of objective function (LL-KLIEP)

  \[
  J_{\text{LL-KLIEP}}(\alpha) = \frac{1}{N_{\text{te}}} \sum_{x \in D_{\text{te}}} \log \hat{w}(x)
  \]

  \[
  = \frac{1}{N_{\text{te}}} \sum_{x \in D_{\text{te}}} \langle \alpha, \psi(x) \rangle - \log \frac{1}{N_{\text{tr}}} \sum_{x \in D_{\text{tr}}} \exp(\langle \alpha, \psi(x) \rangle)
  \]

- Unconstrained convex optimization:
  - standard gradient ascent method can be used.
  - unique global solution is available.
Mean Matching via LL-KLIEP

- Gradient of the objective function

\[
\frac{\partial J_{\text{LL-KLIEP}}(\alpha)}{\partial \alpha} = \frac{1}{N_{\text{test}}} \sum_{x \in D_{\text{test}}} \psi(x) - \frac{1}{N_{\text{train}}} \sum_{x \in D_{\text{train}}} w(x) \psi(x)
\]

The mean of Test data

The mean of Weighted Training data

At the optimum, the mean \( \psi(x) \) of test inputs = the mean \( w(x) \psi(x) \) of training inputs.

\( \rightarrow \) Finding \( w(x) \) matching the mean of two distributions.
Samples were generated from two Gaussian distributions. We used 100 Gaussian basis functions (Gaussian kernels) centered at randomly chosen test input samples.

\[
\hat{w}(x) = \frac{\exp(\langle \alpha, \psi(x) \rangle)}{\frac{1}{N_{tr}} \sum_{x' \in D_{tr}} \exp(\langle \alpha, \psi(x') \rangle)}
\]

\[
\psi_l(x) = \exp\left(-\frac{\|x - x_l^{test}\|^2}{2s^2}\right)
\]

**Training and Test Densities**

**Estimated Importance**
Model selection of KLI\text{EP}/LL-KLI\text{EP}

Likelihood Cross Validation (LCV)

- The performance of KLI\text{EP} depends on the choice of the basis functions $\psi(x)$
  - How to choose hyper parameters, e.g., the kernel width $s$ for Gaussian kernels:
    \[ K_s(x, x_t) = \exp \left\{-\frac{||x - x_t||^2}{2s^2}\right\}, \]

- However, the correct value of importance for each $x$ is not available for unknown distributions $p_{\text{train}}(x)$ and $p_{\text{test}}(x)$
  - unsupervised learning setting

- LCV: Select the model such that maximized $f(\alpha)$
  1. Divide test samples into $R$ disjoint subsets: $\{D_{te}^r\}_{r=1}^R$
  2. Learn importance: \[ \hat{w}^r(x) \text{ from } \{D_{te}^r\}_{r\neq r}^R \]
  3. Evaluate a model by likelihood:
     \[ \frac{1}{|D_{te}^r|} \sum_{x \in D_{te}^r} \hat{w}^r(x) \]
Classification example under Covariate shift
2-dimensional samples were generated from Gaussian distributions

- We used Importance Weighted Logistic Regression (IWLR)

<table>
<thead>
<tr>
<th></th>
<th>Training $p_{tr}(x, y)$</th>
<th>Test $p_{te}(x, y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y = 0$</td>
<td>$y = 1$</td>
<td>$y = 0$</td>
</tr>
<tr>
<td>$\mu$</td>
<td>(-1,-1)</td>
<td>(3,-1)</td>
</tr>
<tr>
<td>$\sum$</td>
<td>$\begin{pmatrix} 0.25 &amp; 0 \ 0 &amp; 4 \end{pmatrix}$</td>
<td>$\begin{pmatrix} 0.25 &amp; 0 \ 0 &amp; 0.25 \end{pmatrix}$</td>
</tr>
</tbody>
</table>

Correct classification rate of LR is 99.1% while that of IWLR is 100%.
Classification example under Covariate shift
2-dimensional samples were generated from Gaussian distributions

- We used Importance Weighted Logistic Regression (IWLR).

\[
\begin{array}{c|cc|cc}
\mu & (1,0) & (4,2) & (0,2) & (3,1) \\
\Sigma & (0.75, 0) & (0.75, 0.5) & (0.01, 0.1) \\
\end{array}
\]

Correct classification rate of LR is 97.2% while that of IWLR is 99.1%.
Related Work of Density Ratio Estimation

- **Kernel density estimator (KDE)**
  - Separately estimate training and test input densities.
  - Gaussian kernel width is chosen by likelihood cross-validation.

- **Kernel Mean Matching (KMM)** (Huang et al., NIPS2006)
  - Direct importance estimation method in universal reproducing kernel Hilbert spaces (RKHS)
  - There is no model selection method for kernel width.

- **Logistic regression (LogReg)** (Beckel et al., ICML2007)
  - Classifier discriminating training and test input data.
  - Gaussian kernel width is chosen by likelihood cross-validation.

- **Kullback-Leibler Importance Estimation Procedure (KLIEP)** (Sugiyama et al., NIPS2007)
  - Direct importance estimation method using KL Divergence.
  - Gaussian kernel width is chosen by likelihood cross-validation.
Experiments varying input dimension

Mean NMSE over 100 trials.
KMM (s) denotes KMM with kernel width s

**NMSE:**
Normalized Mean Squared Error

\[ \text{NMSE} = \frac{1}{N_{tr}} \sum_{x \in D_{tr}} \left( \frac{\hat{w}(x)}{\sum_{x' \in D_{tr}} \hat{w}(x')} - \frac{w(x)}{\sum_{x' \in D_{tr}} w(x')} \right)^2. \]

**KDE:** Suffers from the curse of dimensionality
**KMM:** Performance depends on kernel width

**KLIEN, LogReg, and LL-KLIEN:** Model selection by LCV works well
## Table of Contents

1. Motivation
   - Covariate shift situation
   - Direct density estimation methods
2. Proposed Method
   - LL-KLIEP: Log-linear model for KLIEP
     - Log-linear model: natural modeling for density ratio function
     - Standard optimization techniques to learn a density ratio function
   - LL-KLIEP(LS): Another optimization technique for LL-KLIEP
   - For applications with large numbers of unlabeled test inputs
Disadvantage: LL-KLIEP and previous methods require to use all test inputs in their optimization procedure.

- We need to iterate over all test inputs when computing the values of the **objective function**:

\[
J_{\text{LL-KLIEP}}(\alpha) = \frac{1}{N_{\text{test}}} \sum_{x \in D_{\text{test}}} \langle \alpha, \psi(x) \rangle - \log \frac{1}{N_{\text{train}}} \sum_{x \in D_{\text{train}}} \log \langle \alpha, \psi(x) \rangle
\]

**Evaluation over Test data set**

**Evaluation over Training data set**

- However, the **gradient** of the objective function requires the evaluation of all test samples once.

\[
\frac{\partial J_{\text{LL-KLIEP}}(\alpha)}{\partial \alpha} = \frac{1}{N_{\text{test}}} \sum_{x \in D_{\text{test}}} \psi(x) - \frac{1}{N_{\text{train}}} \sum_{x \in D_{\text{train}}} w(x) \psi(x)
\]

**Independent from** \(\alpha\) ➔ **Pre-computing the value**
An optimization technique w/o the objective function evaluation

**LL-KLIEP(LS1)**

- Idea: the derivative of the convex objective function to be zero at the optimum point.
  - Minimizing a squared norm to measure the ‘magnitude’ of the derivative:

\[
J_{LL-KLIEP(LS1)} = \frac{1}{2} \left\| \frac{\partial J_{LL-KLIEP}(\alpha)}{\partial \alpha} \right\|^2
\]

- Computation time & memory size are independent of \(N_{\text{test}}\).
  - However, the derivative is a quadratic function of the number of parameters, which could be a bottleneck in high dimensional problems.

The partial derivative of LL-KLIEP(LS1)

\[
\frac{J_{LL-KLIEP(LS1)}(\alpha)}{\alpha} = \frac{\partial^2 J_{LL-KLIEP}(\alpha)}{\partial^2 \alpha} \frac{\partial J_{LL-KLIEP}(\alpha)}{\partial \alpha}
\]
LL-KLIEP(LS) for the high-dimensional data

**LL-KLIEP(LS2)**

- Idea: representing the parameter $\alpha$ as a linear combination of the training inputs (representer theorem (Wahba 1990)):

  $$\alpha = \sum_{x \in D_{tr}} \psi(x) \beta_x$$

  where $\{\beta_x\}_{x \in D_{tr}}$ is a data-wise parameter.

- Now, the computation time is linear w.r.t. the number of parameters, $\alpha$ (quadratic w.r.t. the number of the training inputs, $N_{\text{train}}$).
LL-KLIEP (LS): No iteration and no storage for $N_{te}$ in optimization → Well-suited to the applications with the large amount of test samples

Computational complexity and space requirements. $N_{tr}$ is the number of training samples, $N_{te}$ is the number of test samples, $b$ is the number of parameters, and $c$ is the average number of non-zero basis entries.

<table>
<thead>
<tr>
<th></th>
<th>Computational complexity</th>
<th>Space requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pre. Comp. (once)</td>
<td>Objective</td>
</tr>
<tr>
<td>KLI EP</td>
<td>0</td>
<td>$bN_{tr} + bN_{te}$</td>
</tr>
<tr>
<td>LL-KLIEP</td>
<td>$bN_{te}$</td>
<td>$bN_{tr} + bN_{te}$</td>
</tr>
<tr>
<td>LL-KLIEP (LS1)</td>
<td>$bN_{te}$</td>
<td>$bN_{tr}$</td>
</tr>
<tr>
<td>LL-KLIEP (LS2)</td>
<td>$bN_{te}$</td>
<td>$bN_{tr}^2$</td>
</tr>
</tbody>
</table>

- LL-KLIEP (LS1): For lower-dimensional and large-scale training data.
- LL-KLIEP (LS2): For higher-dimensional and moderate-size training data.
Average computation time (including Pre-comp.)
over 100 trials
We varied the number of test inputs, and fixed the number of training inputs.

- we used linear basis function so that the number of bases is equivalent to the input dimension.
- \(d\): input dimension = \#parameter,
\(N_{tr}\): The number of training inputs,
\(N_{te}\): The number of test inputs

The computation time of LL-KLIEP(LS) is independent from the number of test inputs.
Average computation time (including Pre-comp.) over 100 trials
We varied the number of test inputs, and fixed the number of training inputs.

- \( d \): input dimension = \#parameter,
  \( N_{tr} \): The number of training inputs, \( N_{te} \): The number of test inputs

**Moderate-dimensional data**

**Higher-dimensional data**
Conclusion

- We proposed a density ratio estimation method called \textit{LL-KLIEP}.

- We also proposed a scalable optimization technique for LL-KLIEP, in which all the test inputs are iterated once.
  - The computation time is nearly independent of the amount of test data
  - The memory requirement is independent of the amount of test data.
Thank you!
KLIEP/LL-KLIEP objective functions

- KLIEP has a log form in the evaluation of test inputs.

\[
J_{\text{KLIEP}}(\alpha) = \frac{1}{N_{\text{test}}} \sum_{x \in D_{\text{test}}} \log \langle \alpha, \psi(x) \rangle,
\]

subject to \[
\frac{1}{N_{\text{train}}} \sum_{x \in D_{\text{train}}} \langle \alpha, \psi(x) \rangle = 1, \text{ and } \alpha \geq 0
\]

- LL-KLIEP has a linear form in the evaluation of test inputs.

\[
J_{\text{LL-KLIEP}}(\alpha) = \frac{1}{N_{\text{test}}} \sum_{x \in D_{\text{test}}} \langle \alpha, \psi(x) \rangle - \log \frac{1}{N_{\text{train}}} \sum_{x \in D_{\text{train}}} \exp \langle \alpha, \psi(x) \rangle
\]
Kullback-Leibler Importance Estimation Procedure (KLIEP) for Log-linear Models: LL-KLIEP

- Regularized version of LL-KLIEP
  \[ j(\alpha) = \frac{1}{N_{te}} \sum_{x \in D_{te}} \langle \alpha, \psi(x) \rangle \]
  \[ - \log \frac{1}{N_{tr}} \sum_{x \in D_{tr}} \exp(\langle \alpha, \psi(x) \rangle) - \frac{||\alpha||^2}{2\sigma^2} \]

- Gradient of the objective function
  \[ \frac{\partial j(\alpha)}{\partial \alpha} = \frac{1}{N_{te}} \sum_{x \in D_{te}} \psi(x) \]
  \[ - \sum_{x \in D_{tr}} \left[ \frac{\exp(\langle \alpha, \psi(x) \rangle)}{\sum_{x' \in D_{te}} \exp(\langle \alpha, \psi(x') \rangle)} \right] \psi(x) - \frac{\alpha}{\sigma^2} \]

At the optimum, the mean \( \psi(x) \) of test inputs = the mean \( w(x) \psi(x) \) of training inputs.
LL-KLIEP(LS) for the high-dimensional data

**LL-KLIEP(LS2)**

- **Idea:** representing the parameter $\alpha$ as a linear combination of the training inputs (representer theorem (Wahba 1990)):

$$\alpha = \sum_{x \in D_{tr}} \psi(x) \beta_x$$

where \( \{\beta_x\}_{x \in D_{tr}} \) is a data-wise parameter.

**Objective function for LL-KLIEP(LS2)**

$$J_{LS}(\{\beta_x\}_{x \in D_{tr}}) = \frac{1}{2} \left\| F - \sum_{x \in D_{tr}} \psi(x) \omega(x) - \sum_{x \in D_{tr}} \frac{\psi(x) \beta_x}{\sigma^2} \right\|^2$$

where

$$\omega(x) = \frac{\exp\left(\sum_{x' \in D_{tr}} K(x, x') \beta_{x'}\right)}{\sum_{x'' \in D_{tr}} \exp\left(\sum_{x' \in D_{tr}} K(x'', x') \beta_{x'}\right)}$$

$$K(x, x') = \langle \psi(x), \psi(x') \rangle.$$

- Now, the computation time is linear w.r.t. the number of parameters (quadratic w.r.t. the number of the training inputs).
Related work: Kernel Mean Matching (KMM)  
LL-KLIEP (LS2) without a regularizer has the same form as the objective function of KMM.

- Moment matching method:  

\[
\min_{\{w(x)\}_{x \in D_{tr}}} \left[ \frac{1}{2} \sum_{x, x' \in D_{tr}} w(x)w(x') K_s(x, x') - \sum_{x \in D_{tr}} w(x) \kappa(x) \right]
\]

subject to \[
\left| \sum_{x \in D_{tr}} w(x) - N_{tr} \right| \leq N_{tr} \epsilon, \text{ and}
\]

\[
0 \leq w(x) \leq B \text{ for all } x \in D_{tr},
\]

where \[
\kappa(x) = \frac{N_{tr}}{N_{te}} \sum_{x' \in D_{te}} K_s(x, x').
\]

- The objective function of LL-KLIEP (LS2):

\[
\frac{1}{2} \sum_{x, x' \in D_{tr}} w(x)w(x') K_s(x, x') - \sum_{x \in D_{tr}} w(x) \kappa(x),
\]

Disadvantage of KMM.  
The estimates of \(w(x)\) are only available for training samples. Cannot optimize hyper parameters by CV.
Related work: Logistic regression (LogReg)
Classifier discriminating training and test input data

- Selector variable $\delta = -1$ to the training input samples and $\delta = 1$ to the test input samples:

$$p_{tr}(x) = p(x | \delta = -1), \quad p_{te}(x) = p(x | \delta = 1)$$

- Importance can be

$$w(x) = \frac{p(\delta = -1)}{p(\delta = 1)} \frac{p(\delta = 1 | x)}{p(\delta = -1 | x)}.$$  

- The conditional probability $p(\delta | x)$ may be learned by discriminating between the test input samples and the training input samples using LR, where $\delta$ plays the role of a class variable.

$$\hat{w}(x) = \frac{N_{tr}}{N_{te}} \exp(\langle \alpha, \psi(x) \rangle)$$

Empirical estimation

- Objective function: regularized maximum likelihood estimation

- Disadvantage: summation over both training and test samples in CV.
Related work: Kernel density estimator (KDE)

- Estimating $P_{\text{train}}(x)$ and $P_{\text{test}}(x)$ separately.
- KDE: non-parametric density estimator

\[
\hat{p}(x) = \frac{1}{(2\pi s^2)^{d/2}N} \sum_{l=1}^{N} K_s(x, x_l),
\]

- KDE suffers from the curse of dimensionality
An example of supervised learning under covariate shift
Importance weighted logistic regression (IWLR)

- Logistic Regression (LR) : binary case

\[ p_\theta(y|x) = \frac{\exp(yf_\theta(x))}{1 + \exp(yf_\theta(x))} \]

- LR classifier

\[ \hat{y} = \arg \max_y p_\theta(y|x) \]

- Training LR:
  Density ratio is used as weights in the log-likelihood function

\[ \sum_{(x,y) \in Z_{tr}} w(x) \left( \log (1 + \exp (yf_\theta(x))) - yf_\theta(x) \right) + \lambda \| \theta \|^2 \]

- Regularizer

\[ \lambda : \text{a hyper parameter} \]