# Covariate Shift in Hilbert Space: A Solution via Surrogate Kernels

**Kai Zhang** KZHANG@NEC-LABS.COM
NEC Laboratories America, Inc., 4 Independence Way, Suite 200, Princeton, NJ 08540

**Vincent W. Zheng** VINCENT.ZHENG@ADSC.COM.SG
Advanced Digital Sciences Center, 1 Fusionopolis Way, Singapore 138632

**Qiaojun Wang** QJWANG@EDEN.RUTGERS.EDU
Department of Electrical and Computer Engineering, Rutgers, The State University of New Jersey, NJ 08854

**James T. Kwok** JAMESK@CSE.UST.HK
Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong

**Qiang Yang** QYANG@@CSE.UST.HK
Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong

**Ivan Marsic** MARSIC@ECE.RUTGERS.EDU
Department of Electrical and Computer Engineering, Rutgers, The State University of New Jersey, NJ 08854

## Abstract

Covariate shift is a unconventional learning scenario in which training and testing data have different distributions. A general principle to solve the problem is to make the training data distribution similar to the test one, such that classifiers computed on the former generalizes well to the latter. Current approaches typically target on the sample distribution in the input space, however, for kernel-based learning methods, the algorithm performance depends directly on the geometry of the kernel-induced feature space. Motivated by this, we propose to match data distributions in the Hilbert space, which, given a pre-defined empirical kernel map, can be formulated as aligning kernel matrices across domains. In particular, to evaluate similarity of kernel matrices defined on arbitrarily different samples, the novel concept of *surrogate kernel* is introduced based on the Mercer's theorem. Our approach caters the model adaptation specifically to kernel-based learning mechanism, and demonstrate promising results on several real-world applications.

*Proceedings of the $30^{th}$ International Conference on Machine Learning*, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. 

## 1. Introduction

In standard supervised learning, it is commonly assumed that the training and test data are drawn from the same distribution, such that a classifier learned on the former generalizes well to the latter. However, this assumption can be violated in practical situations. For example, the training and test data may be collected under different situations as in applications in bioinformatics or sensor networks. On the other hand, if the training procedure is expensive or labels in the new domain are limited, one might want to apply the knowledge learned from one domain to a different but related domain. In these cases, the traditional learning framework is no longer suited, and how to handle the discrepancy of data distributions across domains becomes a crucial problem.

In this paper, we consider the situation when the training and test data are from different distributions, *i.e.*, $P_{tr}(\mathbf{x}) \neq P_{te}(\mathbf{x})$, but are supposed to share some identical or similar conditional distributions $P_{tr}(y|\mathbf{x}) = P_{te}(y|\mathbf{x})$. This setting is commonly known as the covariate shift or sample selection bias.

There have been a number of attempts to solve this problem. For example, in (Bickel et al., 2007), a discriminative model (MAP classifier) is proposed that directly characterizes the divergence between the training and test distributions. Daumé III & Marcu (2006) investigated how to train a general model with

data from both the source domain and target domain for domain adaptation in natural language processing tasks. Mansour et al. (2009) presented theoretical results on a distribution-weighted combining rule that has a loss of at most a pre-defined value w.r.t. any target mixture of source distributions.

Recently, several work (Shimodaira, 2000; Zadrozny, 2004; Huang et al., 2007; Sugiyama et al., 2008) has converged to the direction of estimating a pointwise re-weighting on the training data to minimize the generalization error in testing. For example, Huang et al. (2007) applied the *kernel mean matching* (KMM) to account for the distribution difference, such that the means of the training and test samples in a reproducing kernel Hilbert space (RKHS) are close. A theoretical analysis was given by (Yu & Szepesvári, 2012). Sugiyama et al. (2008) proposed a framework to estimate the importance ratio with simultaneous model selection, which finds an estimate of the density ratio such that the Kullback-Leibler divergence from the true test input density to its estimate is minimized.

Instead of learning a re-weighting scheme, Pan et al. (2011) proposed to learn the transfer components in the form of pre-parameterized empirical kernel maps, such that the kernel mean of $\Psi(\mathcal{X}_{tr})$ is close to that of $\Psi(X_{te})$. For other recent methods on the more general problem of transfer learning, see (Pan & Yang, 2010).

Most of the current methods study how to make the training and testing data have similar distributions in the input space (Shimodaira, 2000; Zadrozny, 2004; Sugiyama et al., 2008). In (Huang et al., 2007) and (Pan et al., 2011), although the objective function considered is the difference between the sample mean in the feature space, it is used as an indicator of the distance between two distributions in the input space. Therefore, minimizing such an objective is ultimately used to control the difference of distributions in the input space. While one may consider data distribution in the input space for non-kernel-based methods, the behavior of kernel methods are determined in a more complex mechanism due to the interplay between the kernel function and the data distribution. In particular, kernel methods work by applying a linear algorithm in the kernel-induced feature space, where the algorithm performance depends directly on the data distribution in the Hilbert space.

Motivated by this observation, we propose to make the training and testing data have similar distributions in the Hilbert space, which we believe is a more direct way in tackling the covariate shift problem for kernel-based learning algorithms. In particular, considering that the feature space geometry is determined uniquely by the kernel matrix, this can be reformulated as requiring the kernel matrix to be similar for the two domains under certain conditions[1]. One big technical difficulty, however, is that the kernel matrices are data-dependent, and how to evaluate similarity between kernel matrices across domains remains unclear. To bridge this gap, we introduce the concept of *surrogate kernels* based on the Mercer's theorem, the fundamental theorem underlying the reproducing kernel Hilbert space (RKHS). It provides a convenient interface for kernel matrices to compare with each other. By using the surrogate kernel, we can apply an explicit (linear) transform on the kernel matrix of the training data, forcing it to properly "approach" that of the test data, such that the kernel machine learned on the training data generalizes well to the test domain.

The rest of the paper is organized as follows. Section 2 introduces concept of surrogate kernel. In Section 3, we propose a symmetric transform to align kernel matrices across domains. Section 4 provides experimental results, and the last section concludes the paper.

## 2. Surrogate Kernel

Handling the distribution of data in the Hilbert space is difficult, since the kernel-induced feature map usually cannot be explicitly represented. In particular, not all operations involved can be reduced to inner products as required by the kernel trick. Therefore, in order for two samples (e.g., the training and testing samples) to have similar feature-space distributions, we instead require them to have similar kernel matrices. The latter can be somehow viewed as a sufficient condition of the former *given a pre-defined empirical kernel map*. To see this, note that given a kernel matrix $K = (KK^{-\frac{1}{2}})(K^{-\frac{1}{2}}K)$, the corresponding empirical kernel map can be written as $\Psi_{emp} = K^{\frac{1}{2}}$ (Schölkopf et al., 1998; Pan et al., 2011). Therefore, if two kernel matrices are the same, i.e., $K_{\mathcal{Z}} = K_{\mathcal{X}}$, then their corresponding (empirical) feature maps will also be the same, i.e., $\Psi(\mathcal{Z}) = \Psi(\mathcal{X})$, and as a result the distributions of the data in the kernel-induced feature space will be the same, i.e., $p[\Psi(\mathcal{Z})] = p[\Psi(\mathcal{X})]$. In other words, matching two data distributions in the feature space can be conveniently cast as aligning two kernel matrices, which avoids the difficulty of handling the feature vectors $\Psi(\mathbf{x})$'s.

Inspired by this simple observation, we propose to transform the kernel matrix in the source domain such that it is more similar to that in the target domain. By doing this, the feature map (RKHS) embodied via

[1] See detailed justification at the beginning of Section 2.

the kernel matrices will be similar for the two domains, allowing models trained in one domain to generalize well to the other. However, the kernel matrix is data-dependent. Given kernel matrices defined on two different data sets, it is difficult even to evaluate the closeness between them, since they may be of different dimensions and their rows/columns do not correspond, not to mention aligning one to the other.

To solve this problem, we propose the concept of *surrogate kernel*. More specifically, suppose that we have a kernel matrix $K_{\mathcal{X}}$ defined on the data set $\mathcal{X}$. On the other hand, we are given a new data set $\mathcal{Z}$. Here, we want to generate a surrogate kernel of $K_{\mathcal{X}}$ by somehow "projecting" it from $\mathcal{X}$ to $\mathcal{Z}$, denoted $K_{\mathcal{Z}\leftarrow\mathcal{X}}$. The surrogate kernel $K_{\mathcal{Z}\leftarrow\mathcal{X}}$ should inherit key structures of $K_{\mathcal{X}}$ but, instead of being defined on $\mathcal{X}$, $K_{\mathcal{Z}\leftarrow\mathcal{X}}$ is defined on $\mathcal{Z}$. Therefore, $K_{\mathcal{Z}\leftarrow\mathcal{X}}$ can be used in replacement of $K_{\mathcal{X}}$ when we want to compare $K_{\mathcal{X}}$ with any kernel matrix defined on $\mathcal{Z}$.

In order to define the structure of the kernel matrix and how to faithfully preserve it across domains, we will resort to the following theorem.

**Theorem 1.** *(Mercer) Let $K(\boldsymbol{x}, \boldsymbol{y})$ be a continuous symmetric non-negative function which is positive definite and square integrable w.r.t. the distribution $p(\cdot)$, then*

$$K(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{\infty} \lambda_i \phi_i(\boldsymbol{x}) \phi_i(\boldsymbol{y}). \tag{1}$$

*Here, the non-negative eigenvalues $\lambda_i$'s and the orthonormal eigenfunctions $\phi_i$'s are the solutions of the following integral equation*

$$\int K(\boldsymbol{x}, \boldsymbol{y}) p(\boldsymbol{y}) \phi_i(\boldsymbol{y}) d\boldsymbol{y} = \lambda_i \phi_i(\boldsymbol{y}). \tag{2}$$

The Mercer's theorem (Schölkopf & Smola, 2001) is the fundamental theorem underlying reproducing kernel Hilbert space. It states that any psd kernel can be reconstructed by the kernel eigenfunctions (1). In particular, given a data set $\mathcal{X}$ with distribution $p(\cdot)$ and corresponding kernel matrix $K_{\mathcal{X}}$, if we can compute the kernel's eigenspectrum $\lambda_i$'s and continuous eigenfunctions $\phi_i(\cdot)$'s in (2), we will then be able to evaluate the kernel (1) on arbitrary pairs of points. If the evaluation is performed on a new data set $\mathcal{Z}$, a regenerated kernel matrix on $\mathcal{Z}$ will be obtained. In other words, the Mercer's theorem provides an explicit way to generate a kernel matrix on any sample. This regenerated kernel matrix builds entirely on the eigensystem of the kernel matrix $K_{\mathcal{X}}$. Therefore, we believe that it preserves key structures of $K_{\mathcal{X}}$, and can be used as its surrogate on the new sample $\mathcal{Z}$.

### 2.1. Estimating Kernel Eigenfunctions

Next comes the problem of estimating the eigenspectrum and continuous eigenfunctions, i.e., the solution of the integral equation (2). Thanks to (Shawe-Taylor et al., 2005), it can be approximated asymptotically by a finite-sample eigenvalue-decomposition on the empirical kernel matrix $K_{\mathcal{X}}$. In the following, we derive a concrete approximation (Williams & Seeger, 2001). Suppose sample $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$ is drawn from $p(\cdot)$. Then we can approximate the integral in (2) by the empirical average:

$$\frac{1}{n} \sum_{j=1}^{n} k(\mathbf{x}, \mathbf{x}_j) \phi_i(\mathbf{x}_j) \simeq \lambda_i \phi_i(\mathbf{x}). \tag{3}$$

Choosing $\mathbf{x}$ in (3) from $\mathcal{X}$ leads to a standard eigenvalue decomposition $K_{\mathcal{X}}\Phi_{\mathcal{X}} = \Phi_{\mathcal{X}}\Lambda_{\mathcal{X}}$, where $K_{\mathcal{X}}(i,j) = k(\mathbf{x}_i, \mathbf{x}_j)$, $\Phi_{\mathcal{X}} \in \mathbb{R}^{n\times n}$ has orthonormal columns and $\Lambda_{\mathcal{X}} \in \mathbb{R}^{n\times n}$ is a diagonal matrix. The eigenfunctions $\phi_i(\cdot)$'s and eigenvalues $\lambda_i$'s in (2) can be approximated respectively by the columns of $\Phi$ and the diagonal entries of $\Lambda$, up to a scaling constant. According to (3), the eigenfunction $\phi_i(\mathbf{x})$ at any point $\mathbf{x}$ can be extrapolated by $\phi_i(\mathbf{x}) = \frac{1}{\lambda_i n} \sum_{j=1}^{n} k(\mathbf{x}, \mathbf{x}_j)\phi_i(\mathbf{x}_j)$. Therefore, if we want to evaluate the eigenfunctions $\phi_i(\cdot)$'s $(i = 1, ..., n)$ on the new set $\mathcal{Z}$, we can write them in matrix form as

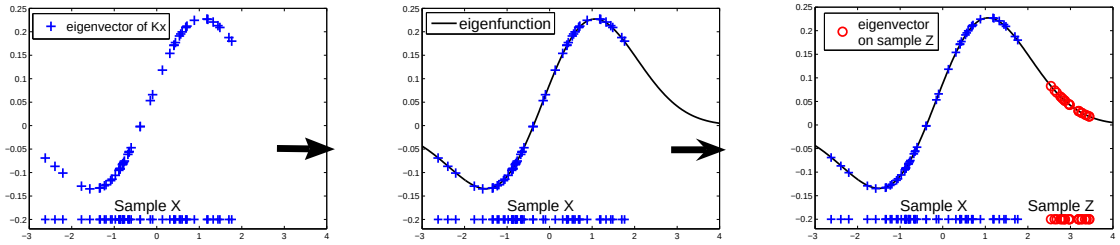
$$\Phi_{\mathcal{Z}} = K_{\mathcal{Z}\mathcal{X}} \Phi_{\mathcal{X}} \Lambda_{\mathcal{X}}^{-1}, \tag{4}$$

where $K_{\mathcal{Z}\mathcal{X}}$ is the cross-similarity matrix between $\mathcal{Z}$ and $\mathcal{X}$, evaluated using kernel $k$.

We illustrate the idea in Figure 1. Let $\mathcal{X}$ be drawn from a normal distribution. In Figure 1(a), we compute the RBF kernel matrix $K_{\mathcal{X}}$, and plot one of its eigenvectors. In Figure 1(b), we estimate the continuous eigenfunction underlying this eigenvector (4) and plot it with solid curve. In Figure 1(c), we project the eigen-structure of $K_{\mathcal{X}}$ from $\mathcal{X}$ to a new sample set $\mathcal{Z}$ by evaluating this continuous eigenfunction on $\mathcal{Z}$, which gives a reconstructed eigenvector on $\mathcal{Z}$. As can be seen, in the process of projecting the eigenvector of $K_{\mathcal{X}}$ from $\mathcal{X}$ to $\mathcal{Z}$, we proceed from a discrete eigenvector (on $\mathcal{X}$) to a continuous eigenfunction (on the whole real domain), and again to a discrete eigenvector (on $\mathcal{Z}$). Here, sample $\mathcal{X}$ is the source of information, and sample $\mathcal{Z}$ only passively receives information from $\mathcal{X}$. In other words, $\mathcal{Z}$ is only a sample on which we choose to reflect and rebuild the kernel matrix $K_{\mathcal{X}}$.

### 2.2. Definition of Surrogate Kernels

The projected eigenvector on $\mathcal{Z}$ can be used to reconstruct a new kernel matrix on $\mathcal{Z}$. In practice, $K_{\mathcal{X}}$




(a) Step1: Compute one eigenvector of $K_{\mathcal{X}}$ on sample $\mathcal{X}$.

(b) Step2: Estimate the eigenfunction underlying the eigenvector.

(c) Step3: Evaluate the eigenfunction on a new sample $\mathcal{Z}$.

*Figure 1.* Projecting one eigenvector of the kernel matrix $K_{\mathcal{X}}$ from sample $\mathcal{X}$ to a new sample $\mathcal{Z}$.

has multiple eigenvectors, each weighted by the corresponding eigenvalue. Therefore, a natural solution to project the eigen-structure of the kernel matrix from $\mathcal{X}$ to $\mathcal{Z}$ is to project each of the eigenvectors of $K_{\mathcal{X}}$ from $\mathcal{X}$ to $\mathcal{Z}$ as shown in Figure 1, and combine them using the eigenvalues of $K_{\mathcal{X}}$, i.e.,

$$\begin{aligned} K_{\mathcal{Z}\leftarrow\mathcal{X}} &= \Phi_{\mathcal{Z}}\Lambda_{\mathcal{X}}\Phi'_{\mathcal{Z}} \\ &= \left(K_{\mathcal{Z}\mathcal{X}}\Phi_{\mathcal{X}}\Lambda_{\mathcal{X}}^{-1}\right)\Lambda_{\mathcal{X}}\left(K_{\mathcal{Z}\mathcal{X}}\Phi_{\mathcal{X}}\Lambda_{\mathcal{X}}^{-1}\right)' \\ &= K_{\mathcal{Z}\mathcal{X}}K_{\mathcal{X}}^{-1}K_{\mathcal{X}\mathcal{Z}}. \end{aligned}$$

**Definition 1.** *Given two samples $\mathcal{X}$ and $\mathcal{Z}$, and a kernel function $k(\cdot,\cdot)$. Let $K_{\mathcal{X}} \in \mathbb{R}^{|X|\times|\mathcal{X}|}$ and $K_{\mathcal{Z}} \in \mathbb{R}^{|\mathcal{Z}|\times|\mathcal{Z}|}$ be the kernel matrices defined on $\mathcal{X}$ and $\mathcal{Z}$, respectively, and $K_{\mathcal{X}\mathcal{Z}} \in \mathbb{R}^{|\mathcal{X}|\times|\mathcal{Z}|}$ be the kernel matrix defined among $\mathcal{X}$ and $\mathcal{Z}$. The surrogate kernel of $K_{\mathcal{X}}$ on sample $\mathcal{Z}$, denoted by $K_{\mathcal{Z}\leftarrow\mathcal{X}}$, is defined as*

$$K_{\mathcal{Z}\leftarrow\mathcal{X}} = K_{\mathcal{Z}\mathcal{X}}K_{\mathcal{X}}^{-1}K_{\mathcal{X}\mathcal{Z}}. \tag{5}$$

In case $K_{\mathcal{X}}$ is positive semi-definite, a peudo-inverse or a samll jittering factor (added to $K_{\mathcal{X}}$) can be used.

**Comments 1.** *The kernel matrix $K_{\mathcal{X}}$ and its surrogate kernel $K_{\mathcal{Z}\leftarrow\mathcal{X}}$ share the same generating mechanism: they are constructed using the same set of eigenfunctions and eigenvalues, but on different samples.*

## 3. Aligning Kernel Matrices via the Surrogate Kernel

The notion of surrogate kernel allows us to "project" a given kernel matrix from one sample to an arbitrary sample while preserving the key eigen-structures. This then serves as a bridge that allows us to compare (henceforth transform among) different kernel matrices. In the following, we propose a parametric transform to rectify the kernel matrix from the training domain, such that it becomes more aligned to the kernel matrix in the test domain.

Suppose that $\mathcal{X}$ comes from the test domain, $\mathcal{Z}$ comes from the training domain, and the two domains have different data distributions. Here we want to adapt the kernel matrix $K_{\mathcal{Z}}$ from the training domain using a symmetric transformation matrix $T \in \mathbb{R}^{|\mathcal{Z}|\times|\mathcal{Z}|}$, as

$$\tilde{K}_{\mathcal{Z}} = T'K_{\mathcal{Z}}T. \tag{6}$$

The goal is that the transformed kernel matrix will be more similar to that in the test domain $\mathcal{X}$. The linear transform on $K_Z$ (6) implicitly enforces a nonlinear transform on $\mathcal{Z}$, i.e., $\tilde{Z} = \mu(\mathcal{Z})$, such that

$$\langle\Psi(\tilde{\mathcal{Z}}), \Psi(\tilde{\mathcal{Z}})\rangle = \langle\Psi(\mathcal{Z})T, \Psi(\mathcal{Z})T\rangle, \tag{7}$$

where $\Psi(\cdot)$ is the kernel map underlying the kernel function. From (7), we can see that the transform $\mu$ underlying (6) is indeed a linear transformation $\Psi(\tilde{\mathcal{Z}}) = \Psi(\mathcal{Z})T$ in the feature space.

The transformed kernel matrix $\tilde{K}_{\mathcal{Z}}$ is of size $|\mathcal{Z}| \times |\mathcal{Z}|$, while the test-domain kernel matrix $K_{\mathcal{X}}$ is $|\mathcal{X}| \times |\mathcal{X}|$. Therefore, in order to align these two matrices, we will replace $K_{\mathcal{X}}$ with its currogate kernel on $\mathcal{Z}$, as $K_{\mathcal{Z}\leftarrow\mathcal{X}} = K_{\mathcal{Z}\mathcal{X}}K_{\mathcal{X}}^{-1}K_{\mathcal{X}\mathcal{Z}}$ (5). Then, we enforce the closeness between $\tilde{K}_{\mathcal{Z}}$ and $K_{\mathcal{Z}\leftarrow\mathcal{X}}$ by using the following optimization problem

$$\min_{T\in\mathbb{R}^{|\mathcal{Z}|\times|\mathcal{Z}|}} \|T'K_{\mathcal{Z}}T - K_{\mathcal{Z}\leftarrow\mathcal{X}}\|_F^2 + \gamma\|T\|_F^2. \tag{8}$$

Here, the term $\|T\|_F^2$ controls the complexity of the transformation $T$, and $\gamma$ controls the balance between enforcing an exact fit and the model complexity. Figure 2 gives an illustration of kernel matrix alignment via surrogate kernels.

By setting derivative of (8) w.r.t. $T$ to zero,

$$T = K_{\mathcal{Z}}^{-\frac{1}{2}}\left(K_{\mathcal{Z}\leftarrow\mathcal{X}} - \frac{1}{2}\gamma K_{\mathcal{Z}}^{-1}\right)^{\frac{1}{2}}. \tag{9}$$

Generally, if $\gamma$ is too large, solution could be complex. This can be avoided by removing negative eigenvectors if it happens (very rare though) as we did in our

experiments. Empirically, we simply fix $\gamma$ at a small value and it is observed that the performance of our algorithm is quite insensitive to the choice of $\gamma$.

Once $T$ is obtained, we can compute the transformed kernel $T'K_{\mathcal{Z}}T$, which corresponds to the inner product of the transformed training data $\tilde{\mathcal{Z}}$.

### 3.1. Cross-Domain Similarity

In order to use the transformed training data $\tilde{\mathcal{Z}}$ in kernel-based learning algorithms, we will need to compute the composite kernel matrix defined on $\tilde{\mathcal{Z}} \cup \mathcal{X}$:

$$G = \begin{bmatrix} \langle \Psi(\tilde{\mathcal{Z}}), \Psi(\tilde{\mathcal{Z}}) \rangle & \langle \Psi(\tilde{\mathcal{Z}}), \Psi(\mathcal{X}) \rangle \\ \langle \Psi(\mathcal{X}), \Psi(\tilde{\mathcal{Z}}) \rangle & \langle \Psi(\mathcal{X}), \Psi(\mathcal{X}) \rangle \end{bmatrix}.$$

By design, we have

$$\begin{aligned} \langle \Psi(\tilde{\mathcal{Z}}), \Psi(\tilde{\mathcal{Z}}) \rangle &= T'K_{\mathcal{Z}}T, \\ \langle \Psi(\mathcal{X}), \Psi(\mathcal{X}) \rangle &= K_{\mathcal{X}}. \end{aligned}$$

We also need to compute the inner product between the transformed training data $\Psi(\tilde{\mathcal{Z}})$ and the original test data $\Psi(\mathcal{X})$, both of which could lie in an infinite-dimensional feature space. By using (7), we have $\Psi(\tilde{\mathcal{Z}}) = \Psi(\mathcal{Z})T$, and so

$$\langle \Psi(\tilde{\mathcal{Z}}), \Psi(\mathcal{X}) \rangle = T'\Psi(\mathcal{Z})'\Psi(\mathcal{X}) = T'K_{\mathcal{Z}\mathcal{X}}.$$

So we have the following composite kernel which can be used in any kernel-based learning algorithm:

$$G = \begin{bmatrix} T'K_{\mathcal{Z}}T & T'K_{\mathcal{Z}\mathcal{X}} \\ K_{\mathcal{X}\mathcal{Z}}T & K_{\mathcal{X}} \end{bmatrix}. \tag{10}$$

The kernel matrix $G$ is always positive semi-definite since it is the inner product of $[\Phi(\tilde{\mathcal{Z}})^\top \Phi(\mathcal{X})^\top]^\top$.

### 3.2. Complexity

Let $|\mathcal{Z}| = n_1$, and $|\mathcal{X}| = n_2$. The space complexity of our approach is $O((n_1+n_2)^2)$. Computing (9) takes $O(n_1n_2+n_1^3)$ time; computing (10) takes $O((n_1+n_2)^2)$ time. Hence, the time complexity of our approach is $O(n_1^3 + (n_1+n_2)^2)$. This can be reduced by low-rank approximation, and will be studied in the future.

### 3.3. Prediction

Let $G_{\mathcal{Z}}$ be the sub-kernel matrix on the training sample $\mathcal{Z}$, and $G_{\mathcal{X}\mathcal{Z}}$ be the sub-block of $G$ corresponding to $\mathcal{X}$ and $\mathcal{Z}$. The learned kernel matrix $G$ in (10) can be used in various kernel-based learning algorithms. For example, in kernel ridge regression, we predict the labels for the test data $\mathcal{X}$ as

$$\mathbf{y}_{\mathcal{X}} = G_{\mathcal{X}\mathcal{Z}}(G_{\mathcal{Z}} + \eta\mathbf{I})^{-1}\mathbf{y}_{\mathcal{Z}}.$$

For SVM, after using $(\tilde{G}_{\mathcal{Z}}, \mathbf{y}_{\mathcal{Z}})$ to train a classifier, we can predict the labels of the test data by

$$\mathbf{y}_{\mathcal{X}} = G_{\mathcal{X}\mathcal{Z}}(\alpha \odot \mathbf{y}_{\mathcal{Z}}) + b,$$

where $\alpha$ is the Lagrange multipliers and $b$ is the bias.

## 4. Experiments

In this section, we perform extensive empirical evaluation on a number of real-world data sets, including text mining (classification) and WiFi-localization (regression). The following methods will be compared:

1. support vector machine (SVM) for classification tasks;
2. kernel ridge regression (KRR) for regression tasks;
3. kernel mean matching (KMM) (Huang et al., 2007);
4. Kullback-Leibler importance estimation procedure (KLIEP) (Sugiyama et al., 2008);
5. transductive SVM (TSVM) (Joachims, 1999);
6. Laplacian-regularized least squares (LAP-RLS) (Belkin et al., 2006);
7. transfer component analysis (TCA) (Pan et al., 2011); and
8. the proposed method.

For SVM and KRR, we apply them on the training data $\mathcal{Z}$ to obtain the model, and then use the learned model for prediction on the test domain $\mathcal{X}$. For KMM and KLIEP, a set of re-weighting coefficients are learned from the training data $\mathcal{Z}$. These are then applied either on a SVM or KRR to obtain a predictive model (using the LIBSVM package), which is used for prediction on the test data $\mathcal{X}$. For the TSVM and LAP-RLS, we combine the data from the training and testing domains together, and use the data from the training domain as labeled data, and the data from the test domain as unlabeled data. For TCA and the proposed method, a kernel matrix $G$ is learned from the training and testing data $\mathcal{X} \cup \mathcal{Z}$; then the sub-kernel matrix $G_{\mathcal{Z}}$ is used in SVM/KRR to obtain the model, and prediction is performed on the test data using this learned model together with $G_{\mathcal{X}\mathcal{Z}}$.

In all the experiments, we randomly choose 60% of the samples from the training and test domains. The experiments is repeated 10 times, and the average performance (together with the standard deviation) is reported. Similar to (Pan et al., 2011), we randomly

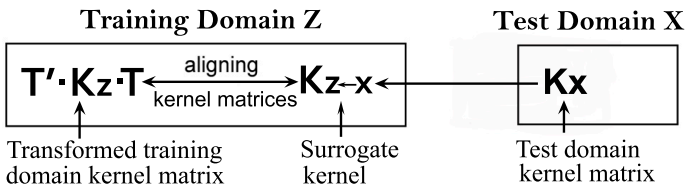



*Figure 2.* Aligning the kernel matrices across domains using the surrogate kernel.

select a very small subset of the test domain data to tune parameters for all the methods.

### 4.1. Text Classification

In this experiment, evaluations are performed on the 20-newsgroups data[2], which has been frequently used to benchmark the performance of transfer learning algorithms with the covariate shift assumption (Pan & Yang, 2010). The data set is a collection of approximately 20,000 newsgroup documents, partitioned across 20 different newsgroups and organized in a hierarchical structure. Data from different subcategories under the same parent category are considered to be from different but related domains. They typically have difference distributions and will be used respectively as training and testing data to examine the performance of transfer learning algorithms. The task is to predict the labels of the parent categories.

Table 1 shows the five binary classification tasks: `rec-vs-talk`, `rec-vs-sci`, `sci-vs-talk`, `comp-vs-sci`, and `comp-vs-talk`. SVM with the linear kernel is used. Accuracies, averaged over 10 randomly drawn subset from the two domains, are reported in Table 2. As can be seen, our approach attains the best performance on most of the tasks.

*Table 1.* 20-newsgroup data for text classification.

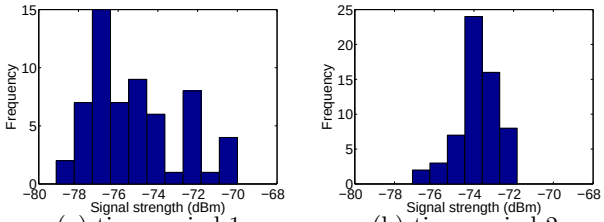
| | $\lvert D_{tr}\rvert$ | $\lvert D_{te}\rvert$ | d |
|---|---|---|---|
| `comp-vs-sci` | 3930 | 4900 | 9893 |
| `comp-vs-talk` | 4482 | 3652 | 10625 |
| `rec-vs-sci` | 3961 | 3965 | 14975 |
| `rec-vs-talk` | 3669 | 3561 | 15254 |
| `sci-vs-talk` | 3374 | 3828 | 15328 |

### 4.2. WiFi Localization

In WiFi localization, we collected WiFi signals $\mathcal{X} = \{x_i \in \mathbb{R}^d\}$ that record the strengths of signals received from $d$ access points, where $i$ is thre measurement index. The corresponding set of locations (labels) is denoted $\mathcal{Y} = \{y_i \in \mathbb{R}\}$. Our WiFi data were collected from the hallway area of an academic building, which is around 64×50 squared meters. The hallway area is discretized into a space of 119 grids, each grid is of 1.5 × 1.5 squared meters. The task is to learn the mapping from the signal space to the location space. This is usually formulated as regression. Due to differences of devices, environment, the training and testing data can have different distributions. Therefore, this is a suitable task for evaluating transfer learning algorithms. We used RBF kernel in this experiment.

#### 4.2.1. Transfer across Time Periods

Here, we collected WiFi data from 119 grids in three different time periods in the same hallway. Figure 3 demonstrates that WiFi signals collected at different time periods can have different distributions. The three times are indexed `t1`, `t2`, and `t3`. We use the data from one time period for training, and the data from another for testing. This gives rise to three tasks, namely, `t1-vs-t2`, `t1-vs-t3`, and `t2-vs-t3`. In each task, the first period provides the training data and the second period provides the test data (Table 3).




(a) time period 1. (b) time period 2.

*Figure 3.* WiFi signals collected at different time periods.

*Table 3.* WiFi data sets over different time periods.

| | $\lvert D_{tr}\rvert$ | $\lvert D_{te}\rvert$ | $d$ |
|---|---|---|---|
| `t1-vs-t2` | 792 | 792 | 67 |
| `t1-vs-t3` | 792 | 792 | 67 |
| `t2-vs-t3` | 792 | 792 | 67 |

For performance evaluation, we transform the regres-

[2] `http://qwone.com/~jason/20Newsgroups/`

*Table 2.* Accuracies (%) on the text classification data sets. The best and comparable results (according to the pairwise t-test with 99.5% confidence) are highlighted.

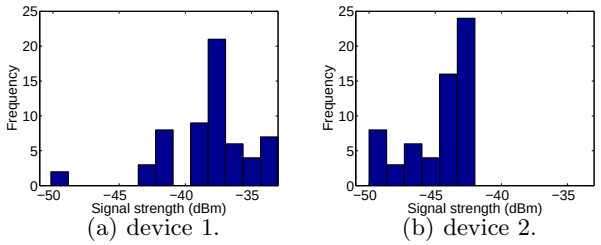
| | SVM | KMM | KLIEP | TSVM | TCA | ours |
|---|---|---|---|---|---|---|
| comp-vs-sci | 63.96±1.69 | 60.96±6.03 | 64.00±1.66 | 63.38±5.81 | **65.50±6.75** | **65.05±3.19** |
| comp-vs-talk | 64.48±2.08 | 64.95±2.01 | 64.92±1.87 | 68.03±1.72 | 71.98±4.12 | **76.08±1.53** |
| rec-vs-sci | 57.91±3.35 | 54.75±2.03 | 58.43±3.52 | **62.03±2.39** | 56.31±4.62 | **62.10±2.32** |
| rec-vs-talk | 62.83±2.52 | 63.73±3.09 | 62.51±1.18 | 65.63±2.64 | 63.40±3.02 | **66.17±2.26** |
| sci-vs-talk | 60.43±2.35 | 60.15±2.82 | 59.83±1.63 | 61.80±1.52 | 56.51±1.64 | **66.00±2.15** |

sion error to localization accuracy as is common in the wireless sensor networks literature. For each signal $x_i$ to be localized, the localization is accurate if the predicted position is within 3 meters from the true position. The accuracy averaged over all the signals is shown in Table 4. As can be seen, our approach gives the best performance on most of the tasks.

#### 4.2.2. Transfer Across Devices

Here we perform WiFi localization on different devices. Different wireless devices usually have different signal sensing capacities, and consequently the signals from different devices will vary from each other. Figure 4 illustrates this by showing the signals from two different devices at the same location.

We collected WiFi data from two different devices at 3 straight-line hallways. The first hallway includes 18 grids, the second hallway include 22 grids, and the third hallway includes 19 grids. We thus have 3 tasks, namely, `hallway1`, `hallway2`, and `hallway3`. In each task, the first device provides the labeled training data, and the second device provides the test data. In pre-processing the data, we remove those dimensions whose signal strengths are all zeros. Due to the difference of the hallways, the effective dimensions for the three tasks are different. (Table 5).




(a) device 1. (b) device 2.

*Figure 4.* WiFi signals collected by different devices at the same location.

Similar to the performance evaluation in Section 4.2.1, localization of each signal is considered accurate if the predicted position is within 6 meters from the true position. Results are shown in Table 6. Again, proposed approach yields best performance on all three tasks.

*Table 5.* WiFi data sets for transfer across devices.

| | $\lvert D_{tr}\rvert$ | $\lvert D_{te}\rvert$ | $d$ |
|---|---|---|---|
| hallway1 | 450 | 450 | 45 |
| hallway2 | 550 | 550 | 53 |
| hallway3 | 475 | 475 | 80 |

## 5. Conclusion and Future Work

We proposed a novel concept of surrogate kernel to align kernel matrices across domains, so as to match data distributions to compensate for the covariate shift in the Hilbert space. In the future, we will study different types of transformation, as well as the use of labeled and unlabeled samples (Kulis et al., 2012). The surrogate kernel has interesting connection with the Nyström method (Williams & Seeger, 2001), the latter mainly used for low-rank approximation of kernel matrices. An interesting topic is how to choose useful landmark points to compute the surrogate kernel for model adaptation, based on sampling or clustering like in (Zhang et al., 2008).

## Acknowledgement

Vincent Zheng is supported by the HSSP research grant from Singapores Agency for Science, Technology and Research (A*STAR). James Kwok is supported in part by the Research Grants Council of the Hong Kong Special Administrative Region (Grant 614311). Qiang Yang thanks the support of Hong Kong CERG projects 621211 and 620812.

## References

Belkin, M., Niyogi, P., and Sindhwani, V. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, (7):2399 – 2434, 2006.

*Table 4.* Accuracies (%) for WiFi localization with transfer over time periods. The best and comparable results (according to the pairwise t-test with 99.5% confidence) are highlighted.

| | KRR | KMM | KLIEP | LAP-RLS | TCA | ours |
|---|---|---|---|---|---|---|
| `t1-vs-t2` | 80.84±1.14 | 81.84±1.25 | 82.67±1.32 | 82.35±1.08 | 86.85±1.61 | **90.36±1.22** |
| `t1-vs-t3` | 76.44±2.66 | 76.42±2.64 | 75.54±1.15 | **94.96±1.04** | 80.48±2.73 | **94.97±1.29** |
| `t2-vs-t3` | 67.12±1.28 | 69.24±1.67 | 70.21±1.05 | **85.34±1.88** | 72.02±1.32 | **85.83±1.31** |

*Table 6.* Accuracies (%) for WiFi localization with transfer across devices The best and comparable results (according to the pairwise t-test with 99.5% confidence) are highlighted.

| | KRR | KMM | KLIEP | LAP-RLS | TCA | ours |
|---|---|---|---|---|---|---|
| `hallway1` | 60.02±2.60 | 55.97±0.80 | 48.57±6.77 | 53.68±0.45 | 65.93±0.86 | **76.36±2.44** |
| `hallway2` | 49.38±2.30 | 42.25±1.16 | 41.71±4.09 | 56.18±0.59 | 62.44±1.25 | **64.69±0.77** |
| `Hallway3` | 48.42±1.32 | 47.36±0.19 | 44.84±3.44 | 51.53±1.04 | 59.18±0.56 | **65.73±1.57** |


Bickel, S., Brückner, M., and Scheffer., T. Discriminative learning for differing training and test distributions. In *International Conference on Machine Learning*, 2007.

Daumé III, H. and Marcu, D. Domain adaptation for statistical classifiers. *Journal of Artifical Intelligence Research*, 26:101–126, 2006.

Huang, J., Smola, A., Gretton, A., Borgwardt, K.M., and Schölkopf, B. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems 19*, pp. 601–608, 2007.

Joachims, T. Transductive inference for text classification using support vector machines. In *International Conference on Machine Learning*, 1999.

Kulis, B., Saenko, K., and Darrell, T. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1785–1792, 2012.

Mansour, Y., Mohri, M., and Rostamizadeh, A. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems*, 2009.

Pan, J. and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22:1345–1359, 2010.

Pan, S. J., Tsang, I.W., Kwok, J.T., and Yang, Q. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22: 199–210, 2011.

Schölkopf, B. and Smola, A. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press, 2001.

Schölkopf, B., Smola, A., and Müller, Klaus-Robert. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

Shawe-Taylor, J., Williams, C., Cristianini, N., and Kandola, J. On the eigenspectrum of the Gram matrix and the generalization error of kernel-PCA. *IEEE Transactions on Information Theory*, 51: 2510–2522, 2005.

Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90:227–244, 2000.

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

Williams, C. and Seeger, M. Using the nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems 13*, 2001.

Yu, Y. and Szepesvári, C. Analysis of kernel mean matching under covariate shift. In *International Conference on Machine Learning*, pp. 478–486, 2012.

Zadrozny, B. Learning and evaluating classifiers under sample selection bias. In *International Conference on Machine Learning*, pp. 903–910, 2004.

Zhang, K., Tsang, I., and Kwok, J. Improved nyström low-rank approximation and error analysis. In *International Conference on Machine Learning*, pp. 1232–1239, 2008.