Importance Weighted Inductive Transfer Learning for Regression

Jochen Garcke and Thomas Vanck

Institut für Numerische Simulation,
Wegelerstr. 6, 53115 Bonn, Germany
garcke@ins.uni-bonn.de, vanck@math.tu-berlin.de

Abstract. We consider inductive transfer learning for dataset shift, a situation in which the distributions of two sampled, but closely related, datasets differ. When the target data to be predicted is scarce, one would like to improve its prediction by employing data from the other, secondary, dataset. Transfer learning tries to address this task by suitably compensating such a dataset shift. In this work we assume that the distributions of the covariates and the dependent variables can differ arbitrarily between the datasets. We propose two methods for regression based on importance weighting. Here to each instance of the secondary data a weight is assigned such that the data contributes positively to the prediction of the target data. Experiments show that our method yields good results on benchmark and real world datasets.

Keywords: inductive transfer learning, importance weighting, dataset shift

1 Introduction

In a standard machine learning setting one has given data $\mathcal{X} \subset \mathbb{R}^{N \times D}$ and corresponding labels $\mathcal{Y} \subset \mathbb{R}^N$. It is assumed that the data is distributed according to $p(x, y)$ and that this distribution never changes; in particular it remains the same for new data. According to this assumption, a good model learned with the training data will also perform well when predicting for such new data. However, this assumption might not always be true, and there are quite often situations in which the underlying distribution changes. In general these situations are called dataset shift. Mathematically speaking, a dataset shift is given if two datasets are samples from two different distributions [5,12]. For instance, suppose one had given the dataset $(\mathcal{X}^P, \mathcal{Y}^P)$, which is distributed according to $p^P(x, y)$, and additionally $(\mathcal{X}^S, \mathcal{Y}^S)$, which was sampled according to $p^S(x, y)$, called primal data and secondary (or supplementary) data, respectively. A dataset shift is given if $p^P(x, y) \neq p^S(x, y)$. An example for such a dataset shift is the so-called covariate shift where the functional relationship between the dependent variable y remains the same, i.e. $p(y|x^P) = p(y|x^S)$ if $x^P = x^S$, but the distribution of the covariates are not the same, i.e. $p(x^P) \neq p(x^S)$ [5,12,16]. Another example is a situation where the distribution of the dependent variable y changes but the distribution of the covariates remains the same. This is referred to as prior probability shift [5].

In this work we will investigate situations where the distribution of the primal (or P) data differs from the distribution of the secondary (or S) data in both dependent variable

y and covariate x, i.e. $p^P(x,y) \neq p^S(x,y)$. For example, consider earthquake data that has been measured in California. A model learned with this data is suitable for making predictions for California, but might not be appropriate for making predictions for earthquakes in Japan due to a shift in the data caused by a change of location. However, if the data provided for Japan is very small a separate model learned solely on the Japan data might not provide a good prediction quality. Although the distributions for the California data and Japan data differ in general, it is reasonable to assume that in some aspects the distributions are very similar or almost equal. Therefore, it might be helpful to augment the Japan data with the California data to improve the prediction quality. This augmentation, also called knowledge borrowing, is commonly known as inductive transfer learning (ITL). Here, the data for California is the supplementary data and the Japan data the primal data. Other such data shift situations occur when the distribution drifts in time. A situation like this occurs, for example, in data that describe the causes of delays of aircrafts. This shift might be due to new airports that have been opened recently or a new aircraft model that is more reliable. Therefore, the data can shift from year to year. Other examples arise in the case of classification of text data where one would like to transfer knowledge obtained on texts about one topic to texts about a different topic.

Formally speaking, inductive transfer learning (ITL) refers to a situation of at least two datasets, which are sampled from the distributions $p^P(x,y)$ and $p^S(x,y)$ with, in general, $p^P(x,y) \neq p^S(x,y)$. Furthermore, the number of the P data is typically much smaller then that of the S data. Additionally, due to the small number of data, a model learned solely on the P data will usually not provide a good prediction quality. However, it is assumed that the distribution p^P and p^S are similar to some degree, which even could result in some connected sets of (x,y) with $p^P(x,y) \approx p^S(x,y)$. In ITL one tries to achieve a good prediction quality of a model for the P data by employing the S data.

In this work we will, motivated by the concept of importance sampling, investigate two new approaches for improving regression in the ITL setting by assigning each instance in the S data a weight. The first one is a supervised and the second an unsupervised method. Although both methods employ labels from the S and P data, we consider the one approach unsupervised since it does not directly employ a cost function for estimating an error between actual and predicted labels. The resulting weights are then used in a modified ridge regression in combination with the S and P data in order to improve the prediction quality on the P data. Experiments show that both approaches yield good results.

This work is structured in the following way: section 2 presents an overview of related work on the topic of ITL while section 3 gives a brief description of ITL in general. Section 4, 5 and 6 explain our idea and state practical instructions. Finally section 7 demonstrates the performance of our algorithm on several datasets.

2 Related Work

The task of inductive transfer learning has been tackled in the past by various approaches. One is the so-called instance based transfer, where each instance in the S domain gets some weight for indicating how much influence it will get for predicting the target data. TrAdaBoost [8] and an extension [1] are methods that assign a weight to each datapoint such that some S data have an influence on the prediction quality for the P data. [13] states a similar boosting approach for regression. Other recent work based on instance transfer has been put forward by [18], [20] and [22] in which they use multiple input sources to improve the prediction quality of classifiers.

Other existing work that implement instance based reweighting methods focus primarily on the covariate shift setting. Important work on this topic has been put forward by [7,10,17], see also [5,16]. It is possible to apply these methods to inductive transfer learning setting. However, the major drawback of these methods in the ITL setting is that they do not take the information about the target labels from the P data into account. This can lead to situations where a S datapoint still gets a high weight assigned due to the similarity to the covariates of the P data although the label, which eventually is what one wants, is fundamentally different from the ones in the P data. To compensate this shortcoming our second approach (explained in section 6.3) is inspired by [17] such that it takes also the labels of the P data into account.

Kernel based ideas have been presented by [15,6], where a special kernel matrix is learned that reflects the similarities between the S and P data. A further method is given in [14], in which an informative prior is constructed from the S data in order to improve a model on the P data. An additional advance is feature representation transfer [3]. This method learns a projection of the S and P data onto a lower dimensional subspace such that the common or shared information of both data can be used for the model on the P data. Learning feature representation is in particular common in the domain of natural language processing (NLP). Since due to differences in vocabulary and writing style learning approaches tend to perform worse in different domains. In this area, [9] proposed a simple, but often well performing, kernel-mapping function for NLP problems, which maps the data from both source and target domains to a high-dimensional feature space, where standard discriminative learning methods are used.

Model transfer or hypothesis transfer learning comprise another class of approaches for treating ITL. In the model transfer setting a model parameter θ_S is learned on the S data. Assuming that the models should be similar, the idea is to regularize the model parameter for the P data θ_P with the help of the parameter θ_S. Recent work on this topic is given by [11] and [19].

Note that, in contrast to multi-task learning [3], inductive transfer learning is not concerned with the prediction quality on both the S and P data, but concentrates only on the prediction of the P data; the S data is exclusively used as data that helps to improve the prediction quality for the P data.

3 Problem Formulation

For inductive transfer learning we now assume a situation where the two datasets $(\mathcal{X}^S, \mathcal{Y}^S)$, the S data, and $(\mathcal{X}^P, \mathcal{Y}^P)$, the P data, are given by:

$$(\mathcal{X}^S, \mathcal{Y}^S) \sim p^S(x, y) \quad \text{and} \quad (\mathcal{X}^P, \mathcal{Y}^P) \sim p^P(x, y).$$

Further, the number M of P data is assumed to be much smaller than the number N of S data, i.e. $|\mathcal{X}^{\mathcal{P}}| \ll |\mathcal{X}^S|$, and the two distributions from which the data was sampled are not equal, i.e. $p^P(x, y) \neq p^S(x, y)$. Nevertheless, it is assumed that the two datasets are somehow related to each other, so that in some parts of the domain the distributions are similar (or even equal), i.e.:

$$p^S(\tilde{x},\tilde{y}) \approx p^P(\tilde{x},\tilde{y}) \text{ for some } (\tilde{x},\tilde{y}).$$

Therefore, one can employ the S data to improve the prediction on the P data. By assumption, we have $p^P(x,y) \neq p^S(x,y)$, and consequently we cannot simply combine the S and P data. The crucial part is to determine points from the S data that contribute positively to the P data prediction and neglect points that have a negative influence. A solution to this problem is based on a measure of similarity between the two distributions. A common way to achieve this is importance sampling, a technique that reweights a given distribution p such that the reweighted p equals another distribution q. Defining the importance weight function as $w(x,y) := \frac{p^P(x,y)}{p^S(x,y)}$ one could reweight the S data distribution by:

$$p^P(x,y) = w(x,y)p^S(x,y) = \frac{p^P(x,y)}{p^S(x,y)}p^S(x,y). \tag{1}$$

With the help of the function $w(x,y)$ it becomes possible to assign each S datapoint (x^S, y^S) an individual and appropriate weight. A weight close to one indicates a preferable point, while a weight far from one indicates the opposite. Hence this approach seems suitable for tackling the induction transfer learning setting. However, this definition of the importance function requires knowledge of both distributions, which is not available. Therefore, an approximation of the importance function $w(x,y)$ is needed instead. By employing an appropriate approximation, the idea of importance sampling offers a guideline for solving the task of ITL.

4 New Instance Based Approach

4.1 Reweighting of the Prediction Function

We start by assuming that the given data $(\mathcal{X},\mathcal{Y})$ is distributed according to an (unknown) distribution $p(x,y)$. This distribution can be expressed by:

$$p(x,y) = p(y|x)p(x) \quad \text{or} \quad p(x,y) = p(x|y)p(y).$$

Although our suggested method can be applied to both cases, the discriminative and the generative one, we will concentrate in the following on the first equation for the discriminative approach. Predictions are obtained by:

$$\hat{y}^* = \text{argmax}_y \left(p(y|x^*)p(x^*)\right). \tag{2}$$

By assumption, the new data x^* and its corresponding (unknown) label y^* is distributed according to $p(x,y)$, and therefore one can make a prediction by applying (2).

However, in the setting of inductive transfer learning we have two different distributions, which gives the following two expressions for the prediction of y^P:

$$\begin{aligned} y^P_P &= \text{argmax}_y \left(p^P(y|x^P)p^P(x^P)\right) \\ y^P_S &= \text{argmax}_y \left(p^S(y|x^P)p^S(x^P)\right). \end{aligned}$$

In general the prediction of y^P based on p^S for the S data, namely y^P_S, can differ arbitrarily from the prediction y^P_P based on the P distribution. Therefore, in order to make better predictions for the P data using the distribution of the S data, we will now reweight the S distribution as suggested in (1):

$$\begin{aligned} y^P &= \text{argmax}_y \left(p^P(y|x^P)p^P(x^P)\right) \\ &= \text{argmax}_y \left(\frac{p^P(y|x^P)p^P(x^P)}{p^S(x^P,y)} p^S(y|x^P)p^S(x^P)\right) \\ &= \text{argmax}_y \left(w(x^P,y)p^S(y|x^P)p^S(x^P)\right). \end{aligned} \tag{3}$$

From this derivation one can see that this also is an unbiased estimator for the P data.

Due to the lack of knowledge about the true distributions p^P and p^S one cannot obtain the correct importance function. Instead we will aim for an approximation $\hat{w}(x,y)$. To determine suitable weights $\hat{w}$ we will now introduce two approaches for their estimation.

4.2 Model Based Estimation of the Weight Function

The first approach will be referred to as the direct method or DITL (Direct ITL) because it will directly rely on the prediction performance of a model learned on the S data. The goal of our model is to minimize the prediction error, i.e.

$$\min ||Y^P - \hat{Y}^P||^2$$

where Y^P is the vector of the real labels $\{y_i\}_{i=1,\dots,M}$ and $\hat{Y}^P$ the vector of the model predictions. Therefore, by following this approach, and with the help of expression (3), an optimization problem for the estimation of a weight function can be stated as:

$$\min_{\hat{w}} \sum_{i=1}^{M} \left(y_i^P - \text{argmax}_y \left(\hat{w}(x_i^P,y)p^S(y|x_i^P)p^S(x_i^P)\right)\right)^2 .$$

The idea behind this approach is that the computation of the weights $\hat{w}$ is performed with respect to the known labels Y^P. Therefore this approach provides a supervised method for adjusting the weights $\hat{w}$. Since for a given point x^P the argmax does not depend on $p^S(x^P)$ that term can be omitted, which leads to:

$$\min_{\hat{w}} \sum_{i=1}^{M} \left(y_i^P - \text{argmax}_y \left(\hat{w}(x_i^P,y)p^S(y|x_i^P)\right)\right)^2 . \tag{4}$$

4.3 Distribution Based Estimation of the Weight Function

Additionally, we propose a method which does not depend directly on prediction models and can be regarded as an unsupervised approach. Following the idea of [17] we will minimize the Kullback-Leibler divergence between two distributions and straightforwardly extend the approach [17] for covariate shift by also taking the labels into account:

$$\begin{aligned}
&\text{argmin}_{\hat{w}}\text{KL}(p^P(x,y)||\hat{w}(x,y)p^S(x,y)) \\
&= \text{argmin}_{\hat{w}}\left(\int p^P(x,y)\log\left(\frac{p^P(x,y)}{\hat{w}(x,y)p^S(x,y)}\right)dxdy\right) \\
&= \text{argmin}_{\hat{w}}\left(-\int p^P(x,y)\log\left(\hat{w}(x,y)\right)dxdy\right).
\end{aligned}$$

The last expression can be approximated by the empirical mean:

$$\Rightarrow \min_{\hat{w}}\sum_{i=1}^{M} -\log\left(\hat{w}(x_i^P,y_i^P)\right). \tag{5}$$

Additionally, one obtains the following constraint for normalization [17]:

$$\begin{aligned}
p^P(x,y) &= w(x,y)p^S(x,y) \\
\Rightarrow 1 = \int p^P(x,y)dxdy &= \int w(x,y)p^S(x,y)dxdy \\
\Rightarrow N &= \sum_{j=1}^{N}\hat{w}(x_j^S,y_j^S),
\end{aligned} \tag{6}$$

it enforces that the reweighted distribution $w \cdot p^S$ still has measure one. We will refer to this approach as the indirect method or KLITL (Kullback-Leibler ITL).

5 Weighted Kernel Ridge Regression for ITL

Assuming one has obtained suitable weights, their application in regression requires adjusted models for prediction. We will propose a weighted kernel ridge regression model, which we will call ITL-KRR. The modified ridge regression model is given by:

$$J_W(\theta) = \frac{1}{2}\left(\sum_{i=1}^{M}(\theta^t\phi(x_i^P)-y_i^P)^2+\sum_{j=1}^{N}w_j(\theta^t\phi(x_j^S)-y_j^S)^2\right)+\frac{\lambda}{2}\sum_{d=1}^{D}\theta_d^2 \tag{7}$$

where $\theta \in \mathbb{R}^D$ denotes the model parameter, λ the regularization parameter, ϕ the feature map that maps the input x into the feature space (see e.g. [4]), and $w_j := \hat{w}(x_j^S, y_j^S)$ denotes the weight for each supplementary datapoint from $\left(\mathcal{X}^S,\mathcal{Y}^S\right)$. Somewhat surprisingly, such a natural extension of a regression approach for applying importance weights has, to our knowledge, not been stated and used in the context of ITL so far. Dualization is given straightforwardly by defining the diagonal matrix $W \in \mathbb{R}^{(M+N)\times(M+N)}$:

$$W := \begin{bmatrix} I_M & 0 \\ 0 & \text{diag}\left(w(x_1^S,y_1^S),\ldots,w(x_N^S,y_N^S)\right)\end{bmatrix}$$

with I_M the identity matrix for M dimensions and appending the S data to the P data ('|' denotes vertical concatenation):

$$X^{PS} = \left(X^P|X^S\right) \text{ and } Y^{PS} = \left(Y^P|Y^S\right)$$
$$K = \phi(X^{PS})^t\phi(X^{PS}) \text{ the kernel matrix,}$$

with the data matrices $X^P \in \mathbb{R}^{M\times D}, X^S \in \mathbb{R}^{N\times D}$ and label vectors $Y^P \in \mathbb{R}^M, Y^S \in \mathbb{R}^N$. As the dual optimization problem one obtains:

$$\frac{1}{2}a^tKWKa - a^tKWY^{PS} + \frac{1}{2}Y^{PS}WY^{PS} + \frac{\lambda}{2}a^tKa.$$

We apply Gaussian kernels, with the bandwidth denoted by σ, in our experiments.

6 Determination of Individual Weights

We will now specify how the weights can be obtained computationally for both approaches.

6.1 Weight Function

Until now we have not been specific in the concrete representation of the weight function $\hat{w}(x,y)$. We employ in this work the common approach of linear combination of Gaussian kernels for an approximation of the importance function, i.e.:

$$\hat{w}^\alpha(x,y) = \sum_{l=1}^{N} \alpha_l \exp\left(-\frac{||(x,y)-(x'_l,y'_l)||^2}{2\eta^2}\right).$$

The centerpoints $(x'_l,y'_l)_{l=1}^N$ will be set to the S datapoints. We use the S data instead of the P data since in (14) we optimize over the P data; using the P data as centerpoints would exhibit a higher risk of overfitting. Hence each $\hat{w}_l$ in (13) becomes

$$\hat{w}_j^\alpha(x^*,y) = \alpha_j \exp\left(-\frac{||(x^*,y)-(x_j^S,y_j^S)||^2}{2\eta^2}\right). \tag{8}$$

Other function representations are possible as well, but out of the scope of this work.

6.2 Direct Approach (DITL)

To derive the direct approach, let us remind the abstract modeling of a prediction function in a standard machine learning setting for the discriminative case:

$$\hat{y}^* = \text{argmax}_y p(y|x^*). \tag{9}$$

Here, x^* denotes a data point to be predicted on, and $\hat{y}^*$ the prediction. As a concrete model $f(x)$ for the S data following (9) we again employ kernel ridge regression, which can be stated as:

$$\operatorname{argmax}_y p(y|x^*) \approx f(x^*) = a^t \underline{k}(x^*), \tag{10}$$

where $\underline{k}(x^*) := (k(x_1, x^*), \dots, k(x_N, x^*))^t$ is the kernel map of the new datapoint x^* and the data $\mathcal{X} \subset \mathbb{R}^{N \times D}$ on which the model has been learned, with $k(x_l, x^*) := \phi(x_l)^t \phi(x^*)$, and a is the vector of coefficients for the linear combination in the feature space. Hence for (4) one needs a different mathematical approximation:

$$\operatorname{argmax}_y \left(\hat{w}(x^*, y) p(y|x^*)\right) \approx f_{\hat{w}(x^*,y)}(x^*) \tag{11}$$

where the model f now also depends on the weight function $\hat{w}$.

We now suggest a weighted prediction model derived from the kernel ridge regression approximation and consider a weighted formulation:

$$J_W(\theta) = \frac{1}{2} \sum_{l=1}^{N} \hat{w}_l \left(y_l - \theta^t \phi(x_l)\right)^2 + \frac{\lambda}{2} ||\theta||^2 \tag{12}$$

where θ again denotes the model parameter, ϕ is the feature map and $\hat{w}_l$ is a weight coefficient for each datapoint x_l. By the process of dualization of the ridge regression [4], one gets the weighted prediction function as:

$$0 = \nabla J_W(\theta) \Leftrightarrow \theta = \sum_{l=1}^{N} \underbrace{\hat{w}_l \left(-\frac{1}{\lambda}(y_l - \theta^t \phi(x_l))\right)}_{=:\hat{a}_l} \phi(x_l).$$

Here, $\hat{a}_l = a_l \hat{w}_l$ are the coefficients for the linear combination in the feature space. Analogously to (10), this prediction function can be taken as an approximation for the weighted prediction, i.e.:

$$\operatorname{argmax}_y \left(\hat{w}(x^*, y) p(y|x^*)\right) \approx f_{\hat{w}(x^*,y)}(x^*) = a^t \hat{W}(x^*, y) \underline{k}(x^*) \tag{13}$$

where $\hat{W}$ denotes a $N \times N$ diagonal matrix where each entry is a weight $\hat{w}_l$ that corresponds to the kernel function $k_l(\cdot) := k(x_l, \cdot)$ and coefficient a_l of the lth-data point of S. Obviously this prediction function contains the label that is to be predicted. Therefore, label prediction for new data points is not possible with (13). However, we are not actually interested in making predictions using this model; rather we would like to estimate appropriate weights $\hat{w}$ for the subsequent step, in which we apply the weights to learn a model on the P data combined with the weighted S data. (4) provides a framework for getting the best possible weights by conditioning the expression to the labels of the P data. Inserting (13) into (4) we get:

$$\min_{\hat{w}} \sum_{i=1}^{M} \left(y_i^P - a^t \hat{W}(x_i^P, y_i^P) \underline{k}(x_i^P)\right)^2. \tag{14}$$

By making the approximation (8) we get a weight function that is defined by a given set of αs, which can now be estimated by (14).

Note that in early experiments we saw that a direct application of (14) sometimes returns αs where only one or very few elements dominate. In order to avoid such an overfitting we additionally add a regularization term to (14) which penalizes large coefficients:

$$\min_{\alpha \geq 0} \sum_{i=1}^{M} \left(y_i^P - a^t \hat{W}^\alpha (x_i^P, y_i^P) \underline{k}(x_i^P) \right)^2 + \gamma ||\alpha||^2. \tag{15}$$

The estimated αs define the weight function $\hat{w}$ which will then be subsequently used during the actual ITL-KRR.

Learning the weights and a better model jointly from the P data and weighted S data requires a three step procedure for the direct approach. Problem (15) depends on a model of the S data for adjusting the αs. Therefore the first step requires the inference of a kernel ridge regression model solely on the S data, which returns the coefficients a for the prediction function (13). With these a a solution to (15) has to be found which yields proper αs. These αs are then used in (7) for calculating the weight for each S datapoint. The procedure can be stated as:

1. Learn a model a for the normal kernel ridge regression using solely the S data and ignore any P data.
2. Use the coefficients vector a from step 1 to determine appropriate αs for the weight function (8) by using the weighted prediction model (13) and solve (15).
3. After having determined the αs in step 2, use these to calculate the weight for the application of the ITL-KRR (7). Use the resulting model to make predictions for new P data.

The optimization in step 2 is convex and therefore guarantees a single optimal solution. Good parameters in each step are estimated by performing standard cross-validation on the P data. We employ Gaussian kernels in the kernel ridge regression, therefore we need to estimate σ and λ in step 1 and 3 similarly to the two parameters γ and η in step 2.

6.3 Indirect Approach (KLITL)

In addition to the direct approach we state a procedure for the indirect approach. Following the derivation in section 4.3, using expression (5) as the objective and expression (6) as the constraint, we proceed as follows:

1. Optimize the following with a standard solver for constrained problems:

$$\max_{\alpha} \frac{1}{M} \sum_{i=1}^{M} \log \left(\hat{w}^\alpha (x_i^P, y_i^P) \right) \qquad \text{s.t. } N = \sum_{j=1}^{N} \hat{w}^\alpha (x_j^S, y_j^S) \text{ and } \alpha \geq 0. \tag{16}$$

2. Use the αs from step 1 to compute the weights $\hat{w}$ of each S datapoint for the optimization of the ITL-KRR (7). Use the resulting model to make predictions for new P data.

We employ here the same representation of the weight function (8) as for the direct approach. For the estimation of a good η in (16) we will apply a modified version of cross-validation that is explained in the experimental section 7.1 of this work.

6.4 Comparison of the Direct and Indirect Approach

Comparing the two approaches, an advantage for the indirect approach is that it does not require the estimation of a model on the S data. This might be advantageous in case when a lot of S data is available. Additionally, the method requires the estimation of just one parameter η for the kernel width used in the weight function. However, on the downside is the fact that this is an unsupervised method. By this we mean a method that does not consider an objective cost function for the parameter inference. Therefore it is less likely to obtain robust or reliable estimations for α. On the other hand DITL applies a supervised optimization problem that takes a subset of the target labels in order to assess the quality of parameter inference. As mentioned further in section 6.2 the additional regularization term allows a higher control of the fitting process. As a consequence the DITL method is much more robust in compensating the dataset shift. The experimental section shows the conditions under which this becomes advantageous. The disadvantage is a higher calculation cost since it requires the calculation of an additional model on the S data and the parameters η and γ.

7 Experiments

In the experimental section we will compare the performance of the direct (DITL) and indirect (KLITL) approaches versus the boosting for transfer learning (TLB) method, another instance-weighted approach, described in [13]. Further we applied the "Frustratingly Easy Domain Adaptation" by [9], a simple, but often well performing feature learning approach, in combination with kernel ridge regression (in the following referred to as FS-KRR). As the final approach for dataset shift, we compare with ATL [6], which is based on Gaussian process (GP) regression and calculates a special correlation matrix for the GP. As a natural baseline, we provide the performance of a normal kernel ridge regression for regression problems learned from the three dataset combinations: P data, S data, and P & S data. As a weighted baseline we also take KLIEP [17] in a normal covariate shift setting for determining instance weights into account, i.e. this approach does not see the labels of the data during weight estimation, only their distribution in x. As an alternative we also employ Kernel Mean Matching (KMM) [10].

7.1 Parameter Selection

DITL applies a kernel ridge regression (KRR), a weight estimation procedure and the ITL-KRR. In each of the three steps we will perform 5-fold standard cross-validation for the parameter estimation. For the KRR and the ITL-KRR we used RBF kernel functions for the calculation of the kernel matrix K. Denoting the bandwidth parameter of the RBF kernels with σ we have to estimate two parameters σ and the regularization parameter λ in step 1 (KRR) on the S data, and step 3 (ITL-KRR) on the S and P data. In the second

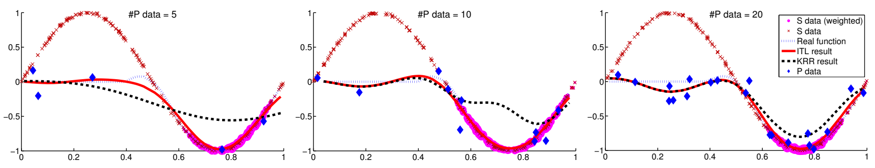

Fig. 1. Illustrative toy example for DITL. From left to right the number of P data is: 5, 10 and 20 datapoints. The location of an S datapoint is marked by a red cross '$\times$'. The round purple points indicate how much weight an S datapoint gets assigned. The thicker the point the more weight it has. As can be seen from the example, in one dimension 20 datapoints are already dense enough to learn a reliable kernel ridge regression.

step DITL requires the estimation of the parameters η (the bandwidth for the importance function approximation) and γ (the regularization parameter for the α vector). Since all problems are quadratic, one can use standard algorithms for quadratic programming.

KLITL is different in the parameter estimation from the DITL method. KLITL requires just two steps. In the first step we solve problem (16); i.e. we simply maximize the sum under the normalization constraint. In order to get a good estimate for η we propose a selection criteria that will choose the η from all the proposed η values that maximizes (16). Since KLITL in the first step is unsupervised, we use a similar method to cross-validation to get a more stable selection result. Given the original S dataset, $\left(\mathcal{X}^S, \mathcal{Y}^S\right)$, we split the dataset into five disjoint parts, $\left(\mathcal{X}^S, \mathcal{Y}^S\right)_{k=1}^5$. Each split $\left(\mathcal{X}^S, \mathcal{Y}^S\right)_k$ should contain enough samples of the S data but due to our assumption this is not a problem. Now for a fixed parameter η we will maximize expressions (16) for each dataset combination $\{\left(\mathcal{X}^S, \mathcal{Y}^S\right)_k, \left(\mathcal{X}^P, \mathcal{Y}^P\right)\}$. We pick the parameter with the highest mean of these five maximas. Therefore we obtain a more robust method for estimating an adequate parameter.

7.2 Datasets

First, for illustration purposes, we show by using a toy example how the proposed DITL algorithm learns weights, and how these weights influence the model prediction. The performance of our methods is then verified on some standard benchmark datasets that have been slightly modified. Finally, we apply our methods to three real world datasets, a dataset describing earthquakes, a second describing delays of aircrafts and a third describing radio signal strengths from WiFi access points for indoor location estimation.

Toy Examples The toy example mainly serves as an illustrative demonstration of how and where the DITL algorithm learns weights for the S data, and shows the consequences for the prediction of the P data when taking additional S data into account. Similar results can be obtained by applying KLITL, which we omit for space reasons.

The dataset is generated by sampling datapoints from two functions that are - as we assume for our methods - partially almost identical. The S data is sampled as:

$$f_s(x) = sin(2\pi x) + \sigma_S \mathcal{N}(0,1), \tag{17}$$

where σ_S is a factor for controlling the influence of the variance (in our experiments we used $\sigma_S = 0.1$). The P data is sampled according to:

$$f_P(x) = \begin{cases} 0 + \sigma_P \mathcal{N}(0,1) & 0 \leq x \leq 1/2 \\ \sin(2\pi x) + \sigma_P \mathcal{N}(0,1) & 1/2 < x \leq 1 \end{cases} \tag{18}$$

where σ_P, as in the case for the S data, is the sample variance (in our experiments $\sigma_P = 0.4$). Parameter selection is performed as described in the previous section 7.1. The experiments in Fig. 1 only apply a very small number of P datapoints (just 5,10 and 20). The reason for this is that, for our example, the performance of a standard KRR is already very good at 20 datapoints. This is due to the fact that in one dimension we get a non-sparse dataset very quickly. Since we want to illustrate that the lack of datapoints (as by our assumption), and hence sparseness of data, leads to models that perform poorly on predicting new data, this setting for our toy example is reasonable. However, in high dimensions the situation is different and the number of P datapoints can be much larger, in parts due to the empty space phenomenon.

Benchmark Datasets We now apply DITL, KLITL, ATL, FS-KRR, KMM, KLIEP and TLB to standard benchmark datasets. The experimental setup is as follows: We took the following standard benchmark datasets for evaluation: abalone, elevators[1], and the kin family datasets[2]. From the kin dataset we took the n datasets (n for nonlinear) with 8 dimensions. We used the nm (non linear medium variance) as the S data and nh (non linear high variance) data as the P data. Since abalone and elevators do not necessarily comprise a dataset shift we will determine the S and P data according to a special selection criteria. The selection process is performed up front and independently of the ITL method. In the first step we normalized the covariates $\mathcal{X}$ to $[0,1]$ for each dimension. Then the following three values are calculated randomly; First, a dimension $d \in \{1,\ldots,D\}$ is selected randomly. In the same way we choose a threshold value $\vartheta \in [0,1]$ randomly and finally we sample a selection probability $p_{select} \in [0,1]$. All values are selected according to a uniform distribution on the corresponding domain. After that we fix these three values for the actual data generation process. For the dataset generation we select a datapoint (x,y) from the set $(\mathcal{X},\mathcal{Y})$, take the $x \in \mathcal{X}$ and then consider the value for dimension d, i.e. x^d. If x^d is larger than the threshold ϑ we will add this (x,y) combination with probability p_{select} to the S data $(\mathcal{X}^{\mathcal{S}},\mathcal{Y}^{\mathcal{S}})$, and to the P dataset $(\mathcal{X}^{\mathcal{P}},\mathcal{Y}^{\mathcal{P}})$ otherwise. That way we randomly generate 50 instances of the data sets for each individual experiment with a drift, i.e. a covariate shift, in the distribution. In order to get also a shift in the labels we apply the function

$$f(y) = y + \nu \sin(2\pi y),\ \nu \in [0,1] \tag{19}$$

to the labels of the P data only. For instance $\nu = 0$ means no shift in the labels. This way we generate datasets that account for the ITL setting and, due to the ν parameter, gives control about the strength of the shift such that S and P data still have something in common.

[1] abalone and elevators can be found on mldata.org

[2] kin datasets are part of the delve dataset repository

Table 1. Results on different benchmark datasets for mean square error. Sampling of the S and P data is explained in the text. Each experiment has been performed 50 times and the results have been normalized by the error on the P data. Therefore, each number in the other columns denotes the proportion in percent. Further comments on the results can be found in the text. Error calculation has been performed on a randomly sampled P_{eval} for each trial. (KRR* = KRR on $S \cup P$). Best results are marked as bold text.

	KRR (on P)	KRR (S)	KRR*	FS-KRR	KMM	ATL	TLB	KLIEP	KLITL	DITL
Abalone $\nu = 0$ (no additional label shift), error on $\|P_{eval}\| = 1000$ and $\|S\| = 1000$										
#P 50	0.0017 / 1.00	0.88	**0.72**	0.87	0.91	0.85	0.83	0.90	0.84	0.78
#P 100	0.0016 / 1.00	0.94	**0.77**	0.94	0.95	0.92	0.90	0.94	0.89	0.85
#P 200	0.0014 / 1.00	0.98	**0.85**	0.97	0.96	0.96	0.98	0.98	0.96	0.89
#P 300	**0.0012 / 1.00**	1.03	**0.99**	1.02	**1.00**	**1.00**	**1.00**	**1.00**	1.01	**0.99**
Abalone ($\nu = 1/2$), error on $\|P_{eval}\| = 1000$ and $\|S\| = 1000$										
#P 50	0.0024 / 1.00	1.53	1.46	0.92	0.93	0.89	0.81	1.48	0.80	**0.76**
#P 100	0.0019 / 1.00	1.41	1.38	0.93	0.96	0.91	0.85	1.38	0.87	**0.80**
#P 200	0.0016 / 1.00	1.42	1.27	0.96	1.01	0.94	0.93	1.25	0.92	**0.89**
#P 300	0.0013 / 1.00	1.45	1.20	1.01	1.00	**0.99**	**0.99**	1.21	1.00	**0.97**
Elevators ($\nu = 1.0$), error on $\|P_{eval}\| = 1000$ and $\|S\| = 2000$										
#P 50	6.5e-6 / 1.00	1.61	1.51	0.91	0.89	0.88	0.74	1.53	0.76	**0.68**
#P 100	5.7e-6 / 1.00	1.51	1.40	0.97	0.95	0.91	0.78	1.45	0.79	**0.71**
#P 200	4.1e-6 / 1.00	1.42	1.38	0.99	0.98	0.97	0.94	1.35	**0.90**	**0.89**
#P 300	**3.6e-6 / 1.00**	1.49	1.29	1.01	1.02	1.01	**1.00**	1.30	1.01	**0.99**
kin dataset ($\nu = 1/4$), error on $\|P_{eval}\| = 1000$ and $\|S\| = 2000$										
#P 50	0.065 / 1.00	1.30	1.28	0.88	0.90	0.87	0.83	1.27	0.84	**0.79**
#P 100	0.056 / 1.00	1.34	1.23	0.91	0.93	0.89	0.88	1.24	0.88	**0.84**
#P 150	0.050 / 1.00	1.32	1.19	0.95	0.95	0.94	**0.92**	1.18	**0.91**	**0.91**
#P 200	**0.044 / 1.00**	1.30	1.15	1.03	**1.00**	**1.00**	**1.00**	1.12	**1.00**	**1.00**

Table 1 shows the results for each method for a different number of P data. For illustration we give one result with $\nu = 0$, i.e. with only a covariate shift. As one would expect, a standard KRR using both S and P data performs best, since for $\nu = 0$ the datasets only contain a covariate shift. Nevertheless, this experiment verifies that the introduced ITL methods learn proper weights in order to employ the right S datapoints for improving prediction of the P data. Their prediction performance is best over all approaches which aim to take a shift into account, both the covariate shift procedures and the full dataset shift procedures. Experimental results are qualitatively the same for the other datasets, therefore we omit them.

When adding a shift to the labels with $\nu > 0$ to have a full dataset shift setting the situation is as expected differently. The KRR learned exclusively on the S data does not show any performance gain by adding P data. This is to be expected since the P data has no influence on the learning procedure but only serves as an evaluation dataset. On the other hand, if learned on $P \cup S$ the results improve slightly but they are still biased by the S data. Over all approaches, as the proportion of the P data grows the error gets reduced. FS-KRR and ATL show comparable errors, this can be explained by the similarity in these approaches, by construction both do not use weights for each instance but one weight for the correlation of P and S data. Consequently, each S datapoint has an equal influence. For KMM we considered the $\frac{p^P(x,y)}{p^S(x,y)}$ for the ratio calculation since that better fits the ITL setting. Note that KMM does not provide a method for parameter selection, and it is unsupervised since it does not use a subset of the target labels to adjust the parameters, which overall makes it less robust and shows moderate performance. KLIEP used as a baseline covariate shift approach shows a poor performance, which is reasonable since it is not adapted to the ITL setting. The performance differences to the other methods show that it makes sense to treat ITL and covariate shift as two separate problem classes. Note that we also considered other (related) methods for covariate shift [16] in our experiments, their performance was similar to KLIEP and we therefore do not report their detailed results. TLB and KLITL show a similar performance. DITL performs best, we assume that this is due to the supervised way for estimating the weights. Nearly all methods eventually converge to a value of 1.00 because, as demonstrated by the toy example in section 7.2, with some data set size the P data provides enough information about its structure to allow a good prediction performance.

Real World Datasets We now investigate the more interesting situation of real data that very likely contains a distribution shift. The first dataset [2] decribes measurements taken during earthquakes in Japan and California. The features describe values such as magnitude or distance to the center. A categorical feature describes the type of the earthquake. We augmented the dataset and assigned a separate dimension for each category, which turns one dimension into three. It seems natural to assume that the shift within this data is due to the different locations. The label to predict is the so-called PGA (Peak Ground Acceleration) value.

The second real world dataset describes the flight arrival and departure details for all commercial flights within the USA[3]. The complete dataset contains records from October 1987 to April 2008. We took the data from 2007 as the S data and the 2008 data as the P data. Also here one can argue that the measurement taken in 2008 are different to 2007 due to a shift in time. The predicted value is the delay of a particular flight.

The third dataset [21] comprises data for indoor location estimation from radio signal strengths received by a user device (like a PDA) from various WiFi Access Points. The measurements are taken at different locations and therefore contain a dataset shift.

The results are shown in table 2. Besides FS-KRR and ATL all approaches which take a shift into account consistently improve the result in comparison to the baseline approach of KRR on P (and/or S). Adjusting for a covariate shift with KLIEP only slightly improves the result, whereas approaches which also adjust with weights stemming from a dataset shift view achieve much better performance. The supervised approach DITL consistently performs best, with KLITL and TLB as second.

In a final experiment we added additional distortion to the labels with (19) and thereby increased the shift in the labels artificially. The purpose of this additional shift is to investigate the robustness of the methods, assuming that with a stronger shift the

[3] Flight dataset available at http://stat-computing.org/dataexpo/2009/

Table 2. Results for the mean square error on the real world datasets. Since these datasets exhibit real dataset shifts the advantage of applying weighted S data becomes obvious. Further the robustness of the methods become apparent when the shift is artificially intensified (i.e. (19) with $\nu > 0$). Best results are marked as bold text.

	KRR (on P)	KRR (S)	KRR*	FS-KRR	KMM	ATL	TLB	KLIEP	KLITL	DITL
	Earthquake $\nu = 0$, error on $\lvert P_{eval}\rvert = 1000$ and $\lvert S\rvert = 841$									
#P 20	0.0138 / 1.00	1.13	1.12	1.13	0.91	1.04	0.64	0.97	0.60	**0.51**
#P 30	0.0106 / 1.00	1.15	1.07	1.14	0.95	1.09	0.88	0.99	0.83	**0.78**
#P 50	0.0076 / 1.00	1.20	1.04	1.19	0.98	1.13	0.96	1.00	0.95	**0.93**
#P 70	**0.0064 / 1.00**	1.23	1.04	1.24	1.02	1.16	**0.99**	1.01	1.02	**1.00**
	Flight Data $\nu = 0$, error on $\lvert P_{eval}\rvert = 1000$ and $\lvert S\rvert = 2000$									
#P 50	898.01 / 1.00	0.96	0.95	1.01	0.88	0.92	0.53	0.92	0.55	**0.51**
#P 200	611.39 / 1.00	1.02	0.99	1.04	0.92	0.99	0.78	0.96	0.76	**0.71**
#P 400	265.97 / 1.00	1.36	1.23	1.35	0.99	1.10	0.89	0.97	0.90	**0.86**
#P 800	**211.12 / 1.00**	1.41	1.36	1.42	1.01	1.14	1.01	1.02	**1.00**	**0.99**
	Wireless $\nu = 0$, error on $\lvert P_{eval}\rvert = 1000$ and $\lvert S\rvert = 2000$									
#P 50	256.83 / 1.00	1.02	0.96	0.99	0.91	0.93	0.74	0.95	0.71	**0.69**
#P 100	230.74 / 1.00	0.98	1.00	1.01	0.95	0.97	0.82	0.97	**0.78**	**0.79**
#P 200	197.23 / 1.00	1.10	1.12	1.05	0.97	1.02	0.93	0.99	0.89	**0.87**
#P 400	153.21 / 1.00	1.13	1.15	1.10	1.03	1.08	0.99	1.01	0.98	**0.96**
	Wireless $\nu = 1$ (with additional label shift), error on $\lvert P_{eval}\rvert = 1000$ and $\lvert S\rvert = 2000$									
#P 50	431.23 / 1.00	1.78	1.17	1.20	1.10	1.18	0.86	1.34	0.84	**0.74**
#P 100	398.19 / 1.00	1.65	1.13	1.14	1.05	1.12	0.90	1.38	0.88	**0.83**
#P 200	354.21 / 1.00	1.77	1.10	1.09	1.07	1.10	0.97	1.40	0.94	**0.92**
#P 400	**299.85 / 1.00**	1.59	1.07	1.04	1.02	1.05	1.01	1.45	**0.99**	**1.00**

methods become more sensitive in the weight calculation, which might lead to a higher error rate. The results confirm this expectation, but also show that it is reasonable to assume that DITL provides a better robustness to stronger shifts than other methods. We only give results for one dataset, the results for other datasets are qualitatively similar.

8 Conclusions

In this paper we suggest two new approaches for tackling the problem of inductive transfer learning. The first one DITL, a supervised method, is motivated by a reweighted and unbiased prediction function of the S data. The second method uses an approximation of the Kullback-Leibler divergence to measure the difference in the distributions of the S and P data. The results indicate that both methods are suitable to account for dataset shifts while the supervised method performs better.

Due to its unsupervised nature, future work on the robustness of KLITL will be an interesting topic. Furthermore, we will investigate the application of the methods in a classification setting. Here, the direct method will need a different optimization than the current formulation (4), which is not suited for classification. Of interest would also be the case of a small number of labeled P data, but large number of unlabeled P data, here one might want to combine covariate shift adaptation with inductive transfer learning.

References

1. Al-Stouhi, S., Reddy, C.K.: Adaptive boosting for transfer learning using dynamic updates. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) ECML/PKDD (1). Lecture Notes in Computer Science, vol. 6911, pp. 60–75. Springer (2011)
2. Allen, T., Wald, D.: Evaluation of ground-motion modeling techniques for use in global shakemap—a critique of instrumental ground-motion prediction equations, peak ground motion to macroseismic intensity conversions, and macroseismic intensity predictions in different tectonic settings. U.S. Geological Survey Open-File Report 2009—1047 (2009)
3. Argyriou, A., Evgeniou, T., Pontil, M.: Convex multi-task feature learning. Machine Learning 73(3), 243–272 (2008)
4. Bishop, C.M.: Pattern recognition and machine learning. Springer (2006)
5. Quiñonero Candela, J.N., Sugiyama, M., Schwaighofer, A., Lawrence, N.D. (eds.): Dataset Shift in Machine Learning. MIT Press (2009)
6. Cao, B., Pan, S.J., Zhang, Y., Yeung, D.Y., Yang, Q.: Adaptive transfer learning. In: AAAI. AAAI Press (2010)
7. Cortes, C., Mohri, M.: Domain adaptation and sample bias correction theory and algorithm for regression. Theoretical Computer Science 519, 103–126 (2014)
8. Dai, W., Yang, Q., rong Xue, G., Yu, Y.: Boosting for transfer learning. In: International Conference on Machine Learning (ICML) (2007)
9. Daumé III, H.: Frustratingly easy domain adaptation. In: ACL. pp. 256–263 (2007)
10. Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., Schölkopf, B.: Covariate Shift by Kernel Mean Matching, pp. 131–160. MIT Press (2009)
11. Kuzborskij, I., Orabona, F.: Stability and hypothesis transfer learning. In: International Conference on Machine Learning (ICML) (2013)
12. Pan, S.J., Yang, Q.: A Survey on Transfer Learning. IEEE Transactions on Knowledge and Data Engineering 22(10), 1345–1359 (Oct 2010)
13. Pardoe, D., Stone, P.: Boosting for regression transfer. In: International Conference on Machine Learning (ICML) (2010)
14. Raina, R., Ng, A.Y., Koller, D.: Constructing informative priors using transfer learning. In: International Conference on Machine Learning (ICML). pp. 713–720 (2006)
15. Rückert, U., Kramer, S.: Kernel-based inductive transfer. In: ECML/PKDD 2008. pp. 220–233. Springer (2008)
16. Sugiyama, M., Kawanabe, M.: Machine learning in non-stationary environments: Introduction to covariate shift adaptation. MIT Press, Cambridge, Mass. (2012)
17. Sugiyama, M., Nakajima, S., Kashima, H., Bünau, P., Kawanabe, M.: Direct importance estimation with model selection and its application to covariate shift adaptation. In: NIPS 20. pp. 1433–1440 (2008)
18. Tan, B., Zhong, E., Wei, E., Yang, X.Q.: Multi-transfer: Transfer learning with multiple views and multiple sources. In: SDM (2013)
19. Tommasi, T., Orabona, F., Caputo, B.: Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In: CVPR. pp. 3081–3088. IEEE (2010)
20. Yang, P., Gao, W.: Multi-view discriminant transfer learning. In: Rossi, F. (ed.) IJCAI. IJCAI/AAAI (2013)
21. Yang, Q., Pan, S.J., Zheng, V.W.: Estimating location using wi-fi. IEEE Intelligent Systems 23(1), 8–13 (2008)
22. Zhang, D., He, J., Liu, Y., Si, L., Lawrence, R.D.: Multi-view transfer learning with a large margin approach. In: Apté, C., Ghosh, J., Smyth, P. (eds.) KDD. pp. 1208–1216. ACM (2011)