Adaptive Transfer Learning

Bin Cao, Sinno Jialin Pan, Yu Zhang, Dit-Yan Yeung, Qiang Yang
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
{caobin,sinnopan,zhangyu,dyyeung,qyang}@cse.ust.hk

Abstract

Transfer learning aims at reusing the knowledge in some source tasks to improve the learning of a target task. Many transfer learning methods assume that the source tasks and the target task be related, even though many tasks are not related in reality. However, when two tasks are unrelated, the knowledge extracted from a source task may not help, and even hurt, the performance of a target task. Thus, how to avoid negative transfer and then ensure a "safe transfer" of knowledge is crucial in transfer learning. In this paper, we propose an Adaptive Transfer learning algorithm based on Gaussian Processes (AT-GP), which can be used to adapt the transfer learning schemes by automatically estimating the similarity between a source and a target task. The main contribution of our work is that we propose a new semi-parametric transfer kernel for transfer learning from a Bayesian perspective, and propose to learn the model with respect to the target task, rather than all tasks as in multi-task learning. We can formulate the transfer learning problem as a unified Gaussian Process (GP) model. The adaptive transfer ability of our approach is verified on both synthetic and real-world datasets.

Introduction

Transfer learning (or inductive transfer) aims at transferring the shared knowledge from one task to other related tasks. In many real-world applications, we expect to reduce the labeling effort of a new task (referred to as target task) by transferring knowledge from one or more related tasks (source tasks) which have plenty of labeled data. Usually, the accomplishment of transfer learning is based on certain assumptions and the corresponding transfer schemes. For example, (Lawrence and Platt 2004; Schwaighofer, Tresp, and Yu 2005; Raina, Ng, and Koller 2006; Lee et al. 2007) assume that related tasks should share some (hyper-)parameters. By discovering the shared (hyper-) parameters, the knowledge can be transferred across tasks. Other algorithms, such as (Dai et al. 2007; Raina et al. 2007), assume that some instances or features can be used as a bridge for knowledge transfer. If these assumptions fail to be satisfied, however, the transfer may be insufficient or unsuccessful. In the worst case, it may even hurt the performance, which can be referred to as *negative transfer* (Rosenstein and Dietterich 2005). Since it is not trivial to verify which assumptions hold for real-world tasks, we are interested in pursuing an adaptive transfer learning algorithm which can automatically adapt the transfer schemes in different scenarios and then avoid negative transfer. We expect the adaptive transfer learning algorithm to at least demonstrate the following properties:

- The shared knowledge between tasks should be transferred as much as possible when these tasks are related. An extreme case is that when they are exactly the same task, the performance of the adaptive transfer learning algorithm should be as good as that when it is considered as a single-task problem.
- Negative transfer should be avoided as much as possible when these tasks are unrelated. An extreme case is when these tasks are totally unrelated, the performance of the adaptive transfer learning algorithm should be no worse than that of the non-transfer-learning baselines.

Two basic transfer-learning schemes can be constructed based the above requirements. One is a *no transfer* scheme, which discards the data in the source task when training a model for the target task. This would be the best scheme when the source and the target tasks are not related at all. The other is *transfer all* scheme that considers the data in the source task to be the same as those in the target task. This would be the best scheme when the source and target tasks are exactly the same. What we wish to get is an adaptive scheme that is always no worse than the two schemes. However, given that there are so many transfer learning algorithms that have been proposed, a mechanism has been lacking to automatically adjust its transfer schemes to achieve this.

In this paper, we address the problem of constructing an adaptive transfer learning algorithm that satisfies both properties mentioned above. We propose an Adaptive Transfer learning algorithm based on Gaus-

sian Process (AT-GP) to achieve the goal of adaptive transfer. Advantages of Gaussian process methods include that the priors and hyper-parameters of the trained models are easy to interpret as well as that variances of predictions can be provided. Different from previous works on transfer learning and multi-task learning using GP which are either based on transferring through shared parameters (Lawrence and Platt 2004; Yu, Tresp, and Schwaighofer 2005; Schwaighofer, Tresp, and Yu 2005) or shared representation of instances (Raina et al. 2007), the model proposed in this paper can automatically learn the transfer scheme from the data. Our key idea is to learn a *transfer kernel* to model the correlation of the outputs when the inputs come from different tasks, which can be regarded as a measure of similarity between tasks. What to transfer is based on how similar the source is to the target task. On one hand, if the tasks are very similar then the knowledge would be transferred from the source data and the learning performance would tend to the *transfer all* scheme in the extreme case. On the other hand, if the tasks are not similar, the model would only transfer the prior information on the parameters to approximate the *no transfer* scheme. Since we have very few labeled data for the target task, we consider a Bayesian estimation of the task similarity rather than point estimation (Gelman et al. 2003). A significant difference between our problem and multitask learning is that we only care about the target task rather than all tasks, which is a very natural scenario in real world applications. For example, we may want to use the previous learned tasks to help learn a new task. Therefore, our target is to improve the new task rather than the old ones. For this purpose, the learning process should focus on the target task rather than all tasks. Therefore, we propose to learn the model based on the conditional distribution of the target task given the source task, which is a novel variation of the classical Gaussian process model.

The Adaptive Transfer Learning Model via Gaussian Process

We consider regression problems in this paper. Suppose that we have a regression problem as a source task $\mathcal{S}$ with a large amount of training data and another regression problem as a target task $\mathcal{T}$ with a small amount of training data. Let $y_i^{(\mathcal{S})}$ denote the observed output corresponding to the input $\mathbf{x}_i^{(\mathcal{S})}$ of the i^{th} instance in the source task and $y_j^{(\mathcal{T})}$ denote the observed output of the j^{th} instance $\mathbf{x}_j^{(\mathcal{T})}$ in the target task. We assume that the underlying latent function between the input and output for the source task is $f^{(\mathcal{S})}$. Let $\mathbf{f}^{(\mathcal{S})}$ be the vector with n^{th} element $f^{(\mathcal{S})}(\mathbf{x}_i^{(\mathcal{S})})$ and we have a notation $\mathbf{f}^{(\mathcal{T})}$ for the target task. Suppose we have N data instances for the source task and M data instances for the target data, then $\mathbf{f}^{(\mathcal{S})}$ is of length N and $\mathbf{f}^{(\mathcal{T})}$ is of length M. We model the noise on observations by an additive noise term,

$$y_i^{(\mathcal{S})} = f_i^{(\mathcal{S})} + \epsilon_i^{(\mathcal{S})}, \qquad y_j^{(\mathcal{T})} = f_j^{(\mathcal{T})} + \epsilon_j^{(\mathcal{T})}$$

where $f^{(\cdot)} = f^{(\cdot)}(\mathbf{x}^{(\cdot)})$ [1]. The prior distribution (GP prior) over the latent variables $\mathbf{f}^{(\cdot)}$, is given by a GP $p(\mathbf{f}^{(\cdot)}) = \mathcal{N}(\mathbf{f}^{(\cdot)}|\mathbf{0}, \mathbf{K}^{(\cdot)})$, with the kernel matrix $\mathbf{K}^{(\cdot)}$. The notation $\mathbf{0}$ denotes a vector with all entries being zero.

We assume that the noise $\epsilon^{(\cdot)}$ is a random noise variable whose value is independent for each observation $y^{(\cdot)}$ and follows a zero-mean Gaussian,

$$p(y^{(\cdot)}|f^{(\cdot)}) = \mathcal{N}(y^{(\cdot)}|f^{(\cdot)}, \beta_{(\cdot)}^{-1}) \tag{1}$$

where β_s and β_t are hyper-parameters representing the precision (inverse variance) of the noise in the source and target tasks, respectively.

Since the noise variables are i.i.d., the distribution of observed outputs $\mathbf{y}^{(\mathcal{S})} = (y_1^{(\mathcal{S})}, \cdots, y_N^{(\mathcal{S})})^{\mathrm{T}}$ and $\mathbf{y}^{(\mathcal{T})} = (y_1^{(\mathcal{T})}, \cdots, y_M^{(\mathcal{T})})^{\mathrm{T}}$ conditioned on corresponding inputs $\mathbf{f}^{(\mathcal{S})}$ and $\mathbf{f}^{(\mathcal{T})}$ can be written in a Gaussian form as follows

$$p(\mathbf{y}^{(\cdot)}|\mathbf{f}^{(\cdot)}) = \mathcal{N}(\mathbf{y}^{(\cdot)}|\mathbf{f}^{(\cdot)}, \beta_{(\cdot)}^{-1}\mathbf{I})) \tag{2}$$

where $\mathbf{I}$ is the identity matrix with proper dimensions.

In order to transfer knowledge from the source task $\mathcal{S}$ to the target task $\mathcal{T}$, we need to construct connections between them. In general, there are two kinds of connections between the source and the target tasks. One is that the two GP regression models for the source and target tasks share the same parameters $\boldsymbol{\theta}$ in their kernel functions. This indicates that the smoothness of the regression functions of the source and target tasks are similar. This type of transfer scheme is introduced in (Lawrence and Platt 2004) for GP models. Many other multi-task learning models also use similar schemes by sharing priors or regularization terms over tasks (Lee et al. 2007; Raina, Ng, and Koller 2006; Ando and Zhang 2005). The other kind of connection is the correlation between outputs of data instances between tasks (Bonilla, Agakov, and Williams 2007; Bonilla, Chai, and Williams 2008). Unlike the first kind (Lawrence and Platt 2004), we do not assume the data in different tasks to be independent of each other given the shared GP prior, but consider the joint distribution of outputs of both tasks. The connection through shared parameters gives it the *parametric* flavor while the connection through correlation of data instances gives it the *nonparametric* flavor. Therefore our model may be regarded as a *semiparametric* model.

Suppose the distribution of observed outputs conditioned on the inputs $\mathbf{X}$ is $p(\mathbf{y}|\mathbf{X})$, where $\mathbf{y} = (\mathbf{y}^{(\mathcal{S})}, \mathbf{y}^{(\mathcal{T})})$ and $\mathbf{X} = (\mathbf{X}^{(\mathcal{S})}, \mathbf{X}^{(\mathcal{T})})$. For multi-task learning problems where the tasks are equally important, the objective would be the likelihood $p(\mathbf{y}|\mathbf{X})$. However, for transfer learning where we have a clear

[1] We use $^{(\cdot)}$ to denote both $^{(\mathcal{S})}$ and $^{(\mathcal{T})}$ to avoid redundancy.

target task, it is not necessary to optimize the parameters with respect to the source task. Therefore, we directly consider the conditional distribution $p(\mathbf{y}^{(\mathcal{T})}|\mathbf{y}^{(\mathcal{S})}, \mathbf{X}^{(\mathcal{T})}, \mathbf{X}^{(\mathcal{S})})$. Let $\mathbf{f} = (\mathbf{f}^{(\mathcal{S})}, \mathbf{f}^{(\mathcal{T})})$, we first define a Gaussian process over $\mathbf{f}$,

$$p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K}),$$

and the kernel matrix $\mathbf{K}$ for transfer learning

$$\mathbf{K}_{nm} \sim k(\mathbf{x}_n, \mathbf{x}_m) e^{-\zeta(\mathbf{x}_n, \mathbf{x}_m)\rho}, \tag{3}$$

where $\zeta(\mathbf{x}_n, \mathbf{x}_m) = 0$ if $\mathbf{x}_n$ and $\mathbf{x}_m$ come from the same task, otherwise, $\zeta(\mathbf{x}_n, \mathbf{x}_m) = 1$. The intuition behind Equation (3) is that the additional factor makes the correlation between instances of the different tasks are less or equal to the correlation between the ones in the same task. The parameter ρ represents the dissimilarity between $\mathcal{S}$ and $\mathcal{T}$. One difficulty in transfer learning is to estimate the (dis)similarity with limit amount of data. We propose a Baysian approach to tackle this difficulty. Therefore, instead of using a point estimation, we can consider ρ is from a Gamma distribution

$$\rho \sim \Gamma(b, \mu).$$

We now have the transfer kernel as

$$\widetilde{\mathbf{K}}_{nm} = \mathbb{E}[\mathbf{K}_{nm}] = k(\mathbf{x}_n, \mathbf{x}_m) \int e^{-\zeta(\mathbf{x}_n, \mathbf{x}_m)\rho} \rho^{b-1} \frac{e^{-\rho/\mu}}{\mu^b \Gamma(b)} \mathrm{d}\rho.$$

By integrating out ρ, we can obtain,

$$\widetilde{\mathbf{K}}_{nm} = \begin{cases} k(\mathbf{x}_n, \mathbf{x}_m) \left(\frac{1}{1+\mu}\right)^b, \zeta(\mathbf{x}_n, \mathbf{x}_m) = 1, \\ k(\mathbf{x}_n, \mathbf{x}_m), \qquad otherwise. \end{cases} \tag{4}$$

The factor before kernel function has range of $[0, 1]$. Therefore, the above form of kernel does not consider the negative correlation between tasks. Therefore, we can further extend it into the following form

$$\widetilde{\mathbf{K}}_{nm} \sim k(\mathbf{x}_n, \mathbf{x}_m)(2e^{-\zeta(\mathbf{x}_n, \mathbf{x}_m)\rho} - 1), \tag{5}$$

and its Bayesian form

$$\widetilde{\mathbf{K}}_{nm} = \begin{cases} k(\mathbf{x}_n, \mathbf{x}_m) \left(2\left(\frac{1}{1+\mu}\right)^b - 1\right), \zeta(\mathbf{x}_n, \mathbf{x}_m) = 1, \\ k(\mathbf{x}_n, \mathbf{x}_m), \qquad otherwise. \end{cases} \tag{6}$$

Theorem 1 shows that the kernel matrices defined in Equation (4) and Equation (6) are positive semidefinite (PSD) matrices as long as k is a valid kernel function. Both transfer kernel models the correlation of outputs based on not only the similarity between inputs but also the similarity between tasks. Since the kernel in Equation (6) has the ability to model negative correlation between tasks and therefore has stronger expression ability, we use it as the transfer kernel. We will further discuss its properties in later section.

Thus, the conditional distribution of $\mathbf{f}^{(\mathcal{T})}$ given $\mathbf{f}^{(\mathcal{S})}$ can be written as follows

$$p(\mathbf{f}^{(\mathcal{T})}|\mathbf{f}^{(\mathcal{S})}, \mathbf{X}^{(\mathcal{T})}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{K_{21}}\mathbf{K}_{11}^{-1}\mathbf{f}^{(\mathcal{S})}, \mathbf{K}_{22} - \mathbf{K}_{21}\mathbf{K}_{11}^{-1}\mathbf{K}_{12}),$$

where $\mathbf{K} = \begin{pmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{pmatrix}$ is a block matrix. $\mathbf{K}_{11}$ and $\mathbf{K}_{22}$ are the kernel matrices of the data in the source task and target task, respectively. $\mathbf{K}_{12}$ $(= \mathbf{K}_{21}^T)$ is the kernel matrix across tasks.

Theorem 1. *Let* $\mathbf{K} = \begin{pmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{pmatrix}$ *be a PSD matrix with* $\mathbf{K}_{12} = \mathbf{K}_{21}^T$. *Then for* $|\lambda| \leq 1$, $\mathbf{K}^* = \begin{pmatrix} \mathbf{K}_{11} & \lambda\mathbf{K}_{12} \\ \lambda\mathbf{K}_{21} & \mathbf{K}_{22} \end{pmatrix}$ *is also a PSD matrix.*

We omit the proof here to reduce space. [2] So far, we have described how to construct a unified GP regression model for adaptive transfer learning. In the following subsections, we will discuss how to do inference and parameter learning in our proposed GP regression model.

Inductive Inference

For a test point x in the target task, we want to predict its output value y by determining the predictive distribution $p(y|\mathbf{y}^{(\mathcal{S})}, \mathbf{y}^{(\mathcal{T})})$, where, for simplicity, the input variables are omitted.

The inference process of the model is the same as that in standard GP models. The mean and variance of the predictive distribution of the target task data are given by

$$m(\mathbf{x}) = \mathbf{k_x}\widetilde{\mathbf{C}}^{-1}\mathbf{y}, \qquad \sigma^2(\mathbf{x}) = c - \mathbf{k_x}^{\mathrm{T}}\widetilde{\mathbf{C}}^{-1}\mathbf{k_x}, \tag{7}$$

where $\widetilde{\mathbf{C}} = \widetilde{\mathbf{K}} + \Lambda$ and $\Lambda = \begin{pmatrix} \beta_s^{-1}\mathbf{I}_N & \mathbf{0} \\ \mathbf{0} & \beta_t^{-1}\mathbf{I}_M \end{pmatrix}$, and $c = k(x, x) + \beta_t^{-1}$ and $\mathbf{k_x}$ can be calculated by the transfer kernel defined in Equation (3). Therefore, $m(x)$ can be further decomposed as follows

$$m(\mathbf{x}) = \sum_{\mathbf{x}_j \in \mathbf{X}^{(\mathcal{T})}} \alpha_j k(\mathbf{x}, \mathbf{x}_j) + \sum_{\mathbf{x}_i \in \mathbf{X}^{(\mathcal{S})}} \lambda\alpha_i k(\mathbf{x}, \mathbf{x}_i), \tag{8}$$

where $\lambda = 2(\frac{1}{1+\mu})^b - 1$ and α_i is the i^{th} element of $\widetilde{\mathbf{C}}^{-1}\mathbf{y}$. The first term in the above formula represents the correlation between the test data point and the data in the target task. The second term represents the correlation between the test data point and the source task data where a shrinkage is introduced based on the similarity between tasks.

Parameter Learning

Given the observations $\mathbf{y}^{(\mathcal{S})}$ in the source task and $\mathbf{y}^{(\mathcal{T})}$ in the target task, we wish to learn parameters $\{\theta_i\}_{i=1}^P$ (P is the number of parameters in the kernel function) in the kernel function as well as the parameter b, μ (denoted by θ_{P+1} and θ_{P+2} for simplicity) by maximizing the marginal likelihood of data of the target task. Multitask GP models (Bonilla, Chai, and Williams 2008) consider the joint distribution of source

[2] The proof of the theorem can be found at http://ihome.ust.hk/~caobin/papers/atgp_ext.pdf

and target tasks. However, for transfer learning problems, we may only have relatively few labeled data in the target task and optimize with respect to the joint distribution may bias the model towards source rather than target. Therefore, we propose to optimize the conditional distribution instead,

$$p(\mathbf{y}^{(\mathcal{T})}|\mathbf{y}^{(\mathcal{S})}, \mathbf{X}^{(\mathcal{T})}, \mathbf{X}^{(\mathcal{S})}). \tag{9}$$

As we analyzed before, this distribution is also a Gaussian and the model is still a GP. A slight difference between this model and classical GP is that its mean is not a zero vector any more and it is also a function of the parameters.

$$p(\mathbf{y}^{(\mathcal{T})}|\mathbf{y}^{(\mathcal{S})}, \mathbf{X}^{(\mathcal{T})}, \mathbf{X}^{(\mathcal{S})}) \sim \mathcal{N}(\boldsymbol{\mu}_t, \mathbf{C}_t), \tag{10}$$

where

$$\begin{aligned} \boldsymbol{\mu}_t &= \mathbf{K}_{21}(\mathbf{K}_{11} + \sigma_s^2\mathbf{I})^{-1}\mathbf{y}_s, \\ \mathbf{C}_t &= (\mathbf{K}_{22} + \sigma_t^2\mathbf{I}) - \mathbf{K}_{21}(\mathbf{K}_{11} + \sigma_s^2\mathbf{I})^{-1}\mathbf{K}_{12}, \end{aligned} \tag{11}$$

and $\mathbf{K}_{11}(\mathbf{x}_n, \mathbf{x}_m) = \mathbf{K}_{22}(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m)$ and $\mathbf{K}_{21}(\mathbf{x}_n, \mathbf{x}_m) = \mathbf{K}_{12}(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m)(2(\frac{1}{1+\mu})^b - 1)$.

The log-likelihood equation is given as follows

$$\ln p(\mathbf{y}_t|\boldsymbol{\theta}) = -\frac{1}{2}\ln|\widetilde{\mathbf{C}}_t| - \frac{1}{2}(\mathbf{y}_t - \boldsymbol{\mu}_t)^{\mathrm{T}}\mathbf{C}_t^{-1}(\mathbf{y}_t - \boldsymbol{\mu}_t) - \frac{N}{2}\ln(2\pi). \tag{12}$$

We can compute the derivative of the log-likelihood with respect to the parameters,

$$\begin{aligned} \frac{\partial}{\partial\theta_i}\ln p(\mathbf{y}|\boldsymbol{\theta}) = &-\frac{1}{2}\mathrm{Tr}(\mathbf{C}_t^{-1}\frac{\partial\mathbf{C}_t}{\partial\theta_i}) \\ &+ \frac{1}{2}(\mathbf{y}_t - \boldsymbol{\mu}_t)^{\mathrm{T}}\mathbf{C}_t^{-1}\frac{\partial\mathbf{C}_t}{\partial\theta_i}\mathbf{C}_t^{-1}(\mathbf{y}_t - \boldsymbol{\mu}_t) \\ &+ (\frac{\partial\boldsymbol{\mu}_t}{\partial\theta_i})^{\mathrm{T}}\mathbf{C}_t^{-1}(\mathbf{y}_t - \boldsymbol{\mu}_t) \end{aligned}$$

The difference between the proposed learning model and classical GP learning models is the existence of the last term in the above equation and non-zero mean Gaussian process. However, the standard inference and learning algorithms can still be used. Thus, many approximation techniques for GP models (Bottou et al. 2007) can also be applied directly to speed-up the inference and learning processes of AT-GP.

Transfer Kernel: Modeling Correlation Between Tasks

As mentioned above, our main contribution is the proposed semi-parametric transfer kernel for transfer learning. In this section, we further discuss its powerful properties for modeling correlations between tasks. In general, the kernel function in GP expresses that for points $\mathbf{x}_n$ and $\mathbf{x}_m$ that are similar, the corresponding values $y(\mathbf{x}_n)$ and $y(\mathbf{x}_m)$ will be more strongly correlated than for dissimilar points. In the transfer learning scenario, the correlation between $y(\mathbf{x}_n)$ and $y(\mathbf{x}_m)$ also depends on which tasks the inputs x_n and x_m come from and how similar the tasks are. Therefore the transfer kernel expresses that for points $\mathbf{x}_n$ and $\mathbf{x}_m$ from different tasks, how the corresponding values $y(\mathbf{x}_n)$ and $y(\mathbf{x}_m)$ are correlated. The transfer kernel can transfer through different schemes in three cases:

- Transfer over priors: $\lambda \to 0$, meaning we know the source and target tasks are not similar or have no confidence on their relation. When the correlations between data in the source and target tasks are slim, what we transfer is only the shared parameters in the kernel function k. So we only require the degree of smoothness of the source and target tasks to be shared.
- Transfer over data: $0 < |\lambda| < 1$. In this case, besides the smoothness information, the model directly transfers data from the source task to the target task. How much the data in the source task influence the target task depends on the value of λ.
- Single task problem: $\lambda = 1$, meaning we have high confidence the task is extremely correlated, we can treat the two tasks to be one. In this case, it is equivalent to the *transfer all* scheme.

The learning algorithm can automatically determine into which setting the problem falls. This is achieved by estimating λ on the labeled data from both the source and target tasks. Experiments in the next section show that only a few labeled data are required to estimate λ well.

Experiments

Synthetic Dataset

In this experiment, we show how our proposed AT-GP model performs when the similarity between the source task and target task changes. We generate a synthetic data set to test our AT-GP algorithm first, in order to better illustrate the properties of the algorithm under different parameter settings. We use a linear regression problem as a case study. First, we are given a linear regression function $f(x) = \mathbf{w}_0^T\mathbf{x} + \epsilon$ where $\mathbf{w}_0 \in \mathbb{R}^{100}$ and ϵ is a zero-mean Gaussian noise term. The target task is to learn this regression model with a few data generated by this model. In our experiment, we use this function to generate 500 data for the target task. Among them, 50 data are randomly selected for training and the rest is used for testing. For the source task, we use $g(x) = \mathbf{w}^T\mathbf{x} + \epsilon = (\mathbf{w}_0 + \delta\Delta\mathbf{w})^T\mathbf{x} + \epsilon$ to generate 500 data for training, where $\Delta\mathbf{w}$ is randomly generated vector and δ is the variable controlling the difference between g and f. In the experiment we increase δ and vary the distance between the two tasks $D_f = ||\mathbf{w} - \mathbf{w}_0||_F$. Figure (2) shows how the mean absolute error (MAE) on 450 target test data changes at different distance between the source and target tasks. The results are compared with the *transfer all* scheme (directly use all of the training data) and the *no transfer* scheme (only use training data in the target task). As we can see, when the two tasks are very similar, the AT-GP model performance is as good as *transfer all*, while when the tasks are very different, the AT-GP

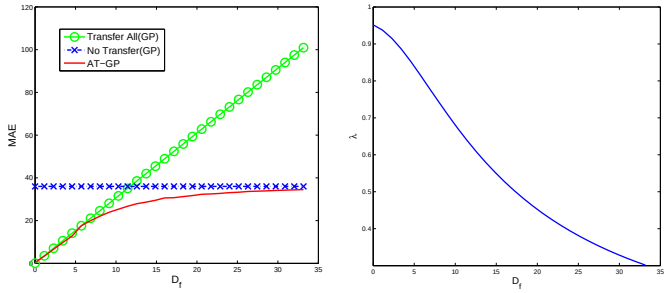

Figure 2: The left figure shows the change to MAE with increasing distance with f. The results are compared with transfer all and no transfer; The right figure shows the change to λ with increasing distance with f. We can see that λ is strongly correlated with D_f.

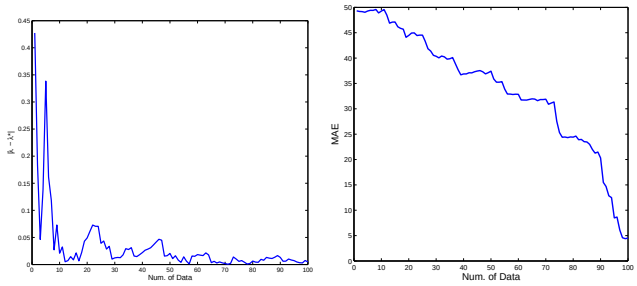

Figure 4: Learning with different numbers of labeled data in the target task. The left figure shows the convergence curve of λ with respect to the number of data. The right figure shows the change to MAE on test data. (λ^* is the value of λ after convergence and $\lambda^* = 0.3$ here.)

model is no worse than *no transfer*. Figure (4) shows the experimental results on learning λ under a varying number of labeled data in the target task. It is interesting to observe that the number of data required to learn λ well (left figure) is much less than the number of data required to learn the task well (right figure). This indicates why transfer learning works.

Real-World Datasets

In this section, we conduct experiments on three real world datasets.

WiFi Localization[3]: The task is to predict the location of each collection of received signal strength (RSS) values in an indoor environment, received from the WiFi Access Points (APs). A set of (RSS values, Location) data is given as training data. The training data are collected at a different time period from the test data, so there exists a distribution change between the training and test data. In WiFi location estimation, when we use the outdated data as the training data, the error can be very large. However, because the location information is constant across time, there is a certain part of the data that can be transferred. If this can be done successfully, we can save a lot of manual labelling effort for the new time period. Therefore, we want to use the outdated data as the source task to help predict the location for current signals. Different from multi-task learning which cares about the performances of all tasks, in this scenario we only care about the performance of current data corresponding to the target task.

Wine[4]: The dataset is about wine quality including red and white wine samples. The features include objective tests (e.g. PH values) and the output is based on sensory data. The labels are given by experts with grades between 0 (very bad) and 10 (very excellent). There are 1599 records for the red wine and 4898 for the white wine. We use the quality prediction problem for the white wine as the source task and the quality prediction problem for red wine as the target task.

SARCOS[5]: The dataset relates to an inverse dynamics problem for a seven degrees-of-freedom SARCOS anthropomorphic robot arm. The task is to map from a 21-dimensional input space (7 joint positions, 7 joint velocities, 7 joint accelerations) to the corresponding 7 joint torques. The original problem is a multi-output regression problem. It can also be treated as multi-task learning problem by treating the seven mappings as seven tasks. In this paper we use one of the task as the target task and another as the source task to test our algorithm. Therefore, we can form 49 task pairs in total for our experiments.

In our experiments, all data in the source task and 5% of the data in the target task are used for training. The remaining 95% data in the target task are used for evaluation. We use NMSE (Normalized Mean Square Error) for the evaluation of results on Wine and SARCOS datasets and error distance (in meter) for WiFi. A smaller value indicates a better performance for both evaluation criteria. The average performance results are shown in Table 1, where *No* and *All* are GP models with no-transfer and transfer-all schemes, and Multi-1 is (Lawrence and Platt 2004) and Multi-2 is (Bonilla, Chai, and Williams 2008).

Discussion

We further discuss the experimental results in this section. For the task pairs in the datasets, sometimes the source task and target task would be quite related, such as the case of WiFi dataset. In these cases, the λ parameter learned in the model would be large, allowing the shared knowledge to be transferred successfully. However, in other cases such as the ones on the SARCOS dataset, the source and target tasks may not be related and negative transfer may occur. A safer way is to use parameter transfer scheme (Multi-1 in (Lawrence and Platt 2004)) or the no transfer scheme to avoid

[3]http://www.cs.ust.hk/∼qyang/ICDMDMC07/

[4]http://archive.ics.uci.edu/ml/datasets/Wine+Quality

[5]http://www.gaussianprocess.org/gpml/data/

Data	No	All	Multi-1	Multi-2	AT
Wine	1.33+0.3	1.37+0.7	1.69+0.5	1.27+0.3	**1.16+0.3**
SARCOS	0.21+0.1	1.58+1.3	0.24+0.1	0.26+0.3	**0.18+0.1**
WiFi	9.18+1.5	5.28+1.3	9.35+1.4	11.92+1.8	**4.98+0.6**

Table 1: Results on three real world datasets. The NMSE of all source/target-task pairs are reported for the dataset Wine and SARCOS, while error distances (in meter) are reported for the dataset WiFi. Both means (before plus) and standard deviation (after plus) are reported. We have conduct t-tests which show the improvements are significant with significance level 0.05.

negative transfer. The drawback of parameter transfer transfer scheme or no transfer scheme is that they may lose a lot of shared knowledge when the tasks are similar. Besides, since multi-task learning cares about both the source and target tasks with no difference and the source task may dominate the learning of parameters, the performance of the target task may even worse than no transfer case, as for the SARCOS dataset. However, what we should be focused on is the target task. In our method, we conduct the learning process on the target task and the learned parameters would fit the target task. Therefore, the AT-GP model performs the best on all three datasets. In many real world applications, it is hard to know exactly whether the tasks are related or not. Since our method can adjust the transfer schema automatically according to the similarity of the two tasks, we are able to adaptively transfer the shared knowledge as much as possible and avoid negative transfer.

Related Work

Multi-task learning is closely related to transfer learning. Many papers (Yu, Tresp, and Schwaighofer 2005; Schwaighofer, Tresp, and Yu 2005) consider multi-task learning and transfer learning as the same problem. Recently, various GP models have been proposed to solve multi-task learning problems. Yu et al. in (Yu, Tresp, and Schwaighofer 2005; Schwaighofer, Tresp, and Yu 2005) proposed the hierarchical Gaussian process model for multi-task learning. Lawrence in (Lawrence and Platt 2004) also proposed a multi-task learning model based on Gaussian process. This model tries to discover the common kernel parameters over different tasks and the informative vector machine was introduced to solve large-scale problems. In (Bonilla, Chai, and Williams 2008) Bonilla et al. proposed a multi-task regression model using Gaussian process. They considered the similarity between tasks and constructed a free-form kernel matrix to represent task relations. The major difference between their model and ours is the constructed kernel matrix. They consider a point estimation of the correlations between tasks, which may not be robust when data in target task is small. They also treat the tasks equally important rather than the transfer setting.

One difference of transfer learning from multi-task learning is that in transfer learning we are particularly interested in transferring knowledge from one or more source tasks to a target task rather than learning these tasks simultaneously. What we concern is the performance in the target task only. On the problem of adaptive transfer learning, to our best knowledge, only (Rosenstein and Dietterich 2005) addressed the problem of negative transfer, but they still failed to achieve adaptive transfer.

Conclusion

In this paper, we proposed an adaptive transfer Gaussian process (AT-GP) model for adaptive transfer learning. Our proposed model can automatically learn the similarity between tasks. According to our method, how much to transfer is based on how similar the tasks are and negative transfer can be avoided. The experiments on both synthetic and real-world datasets verify the effectiveness of our proposed model.

Acknowledgments

Bin Cao, Sinno Jialin Pan and Qiang Yang thank the support of RGC/NSFC grant N_HKUST624/09.

References

Ando, R. K., and Zhang, T. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.* 6.

Bonilla, E. V.; Agakov, F.; and Williams, C. 2007. Kernel multi-task learning using task-specific features. In *In Proc. of the Eleventh International Conference on Artificial Intelligence and Statistics AISTATS'07*.

Bonilla, E.; Chai, K. M.; and Williams, C. 2008. Multi-task gaussian process prediction. In Platt, J.; Koller, D.; Singer, Y.; and Roweis, S., eds., *NIPS 20*. MIT Press.

Bottou, L.; Chapelle, O.; DeCoste, D.; and Weston, J., eds. 2007. *Large Scale Kernel Machines*. Cambridge: MIT Press.

Dai, W.; Yang, Q.; Xue, G.-R.; and Yu, Y. 2007. Boosting for transfer learning. In *Proc. of the 24th ICML*. ACM.

Gelman, A.; Carlin, J. B.; Stern, H. S.; and Rubin, D. B. 2003. *Bayesian Data Analysis*. Chapman & Hall/CRC, second edition.

Lawrence, N. D., and Platt, J. C. 2004. Learning to learn with the informative vector machine. In *Proc. of the 21st ICML*. Banff, Alberta, Canada: ACM.

Lee, S.-I.; Chatalbashev, V.; Vickrey, D.; and Koller, D. 2007. Learning a meta-level prior for feature relevance from multiple related tasks. In *Proc. of the 24th ICML*, 489–496. Corvalis, Oregon: ACM.

Raina, R.; Battle, A.; Lee, H.; Packer, B.; and Ng, A. Y. 2007. Self-taught learning: transfer learning from unlabeled data. In *ICML'07*. New York, NY, USA: ACM.

Raina, R.; Ng, A. Y.; and Koller, D. 2006. Constructing informative priors using transfer learning. In *Proc. of the 23rd ICML*. Pittsburgh, Pennsylvania: ACM.

Rosenstein, M. T., M. Z. K. L. P., and Dietterich, T. G. 2005. To transfer or not to transfer. In *NIPS 2005 Workshop on Transfer Learning*.

Schwaighofer, A.; Tresp, V.; and Yu, K. 2005. Learning gaussian process kernels via hierarchical bayes. In *NIPS 17*.

Yu, K.; Tresp, V.; and Schwaighofer, A. 2005. Learning gaussian processes from multiple tasks. In *Proc. of the 22nd ICML*. Bonn, Germany: ACM.