Easy-but-effective domain sub-similarity learning for transfer regression

Wei, Pengfei; Sagarna, Ramon; Ke, Yiping; Ong, Yew-Soon

2020

Wei, P., Sagarna, R., Ke, Y. & Ong, Y. (2020). Easy-but-effective domain sub-similarity learning for transfer regression. IEEE Transactions On Knowledge and Data Engineering. <https://dx.doi.org/10.1109/TKDE.2020.3039806>

<https://hdl.handle.net/10356/147569>

<https://doi.org/10.1109/TKDE.2020.3039806>

© 2020 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The published version is available at: <https://doi.org/10.1109/TKDE.2020.3039806>.

Downloaded on 26 Apr 2023 13:37:45 SGT

Easy-but-effective Domain Sub-similarity Learning for Transfer Regression

Pengfei Wei^{1,*}, Ramon Sagarna^{2,*}, Yiping Ke³, and Yew-Soon Ong³

Abstract—Transfer covariance function, which can model domain similarity and adaptively control the knowledge transfer across domains, is widely used in transfer learning. In this paper, we concentrate on Gaussian process (GP) models using a transfer covariance function for regression problems in a black-box learning scenario. Precisely, we investigate a family of rather general transfer covariance functions, T_* , that can model the heterogeneous sub-similarities of domains through multiple kernel learning. A necessary and sufficient condition to obtain valid GPs using T_* (GP_{T_*}) for any data is given. This condition becomes specially handy for practical applications as (i) it enables semantic interpretations of the sub-similarities and (ii) it can readily be used for model learning. In particular, we propose a computationally inexpensive model learning rule that can explicitly capture different sub-similarities of domains. We propose two instantiations of GP_{T_*} , one with a set of predefined constant base kernels and one with a set of learnable parametric base kernels. Extensive experiments on 36 synthetic transfer tasks and 12 real-world transfer tasks demonstrate the effectiveness of GP_{T_*} on the sub-similarity capture and the transfer performance.

Index Terms—Transfer regression, transfer covariance function, Gaussian process.

1 INTRODUCTION

Transfer learning [1] has been proposed to deal with the problem in which training and test data are differently distributed. The idea is to learn a model that works well in a target domain (refers to the test data), by leveraging upon the knowledge from a different but related source domain (refers to the training data). With research efforts largely concentrated on transfer classification problems [2]–[4], an increasing amount of studies are dedicated to transfer regression problems. In this setting, various real-world applications arise, e.g. Wi-Fi localization [5], sentiment analysis [6], aerospace systems design [7], motion detection [8], and cyclical manufacturing [9]. In this work, we focus on the transfer regression problem. We use the *Gaussian process* (GP) as the base regression model so as to take advantage of its highly regarded capability to handle prior beliefs on the characteristics of the underlying functions through the use of kernels [10].

A brute-force transfer assuming that domains are mutually relevant is inadvisable as it may lead to negative transfer. Thus, the crucial point in transfer learning is to capture the domain similarity of the source and target domains, and adaptively transfer useful source knowledge to help target tasks. Generally, this is challenging due to the concurrence of consistences and divergences in real-world data. In practice, two domains may share similarities in some characteristics but conflict with each other in the others. For example, in sentiment analysis, we may seek to transfer sentiment knowledge between different product reviews, e.g., books and electronic devices. Both products exhibit the same sentiment on some words, e.g., “good” and “excellent”. However, they perform inconsistently on some other words, e.g., “noisy” and

“comprehensive”. The similarity heterogeneity of domains, which refers to the fact that different domains contain both consistences and divergences, is widespread in real-world applications [3], [11], [12]. Thus, transfer learning methods that are capable of tuning the strength of knowledge transfer to the similarity heterogeneity of domains are desirable.

A popular way of capturing inter-domain similarity through GP is to construct a *transfer covariance function*. Instead of treating all instance pairs equally, a transfer covariance function differentiates the covariance calculation of two instances based on the domains they come from. An early work was proposed in multi-task learning [13], and then a similar idea was adapted to transfer learning [14]. Note that multi-task learning differs from transfer learning in that the former aims at improving performance across all the tasks while the objective of the latter focuses on the target domain only. In this work, we focus on the transfer covariance function for transfer learning problems with one source domain and one target domain.

Generally, a transfer covariance function can be constructed by exploiting distinct functions for different instance pairs, i.e., using k_{dd} if two instances come from the same domain, and $k_{dd'}$ otherwise. Such a transfer covariance function is general in the sense that $k_{dd'}$ can be different from k_{dd} in all respects, and thus it enables flexible designs. For example, [14]–[16] set $k_{dd'} = \lambda k_{dd}$. Such a transfer covariance function, T_λ , takes advantage of a single coefficient λ to model the similarity of domains, which falls within the *intrinsic correlation model* from the geostatistics literature [17]. Using a similarity coefficient is simple and computationally cheap, but it penalizes the capability of precisely modeling the similarity heterogeneity of domains as it can only reflect a compromised relationship. In contrast, by following the *linear model of coregionalization* from geostatistics [17], we may adopt a more flexible transfer covariance function using multiple kernels [18]. In this case, it is assumed that k_{dd} and $k_{dd'}$ lie in the same kernel function space and share the same base kernels. However, k_{dd} and $k_{dd'}$ may perform differently on

• ¹School of Computing, National University of Singapore. ²Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore. ³School of Computer Science and Engineering, Nanyang Technological University, Singapore. * The work was done when the authors worked at Nanyang Technological University.

the base kernels. They may present a divergence on the same base kernel, that is, the coefficients of the same base kernel may be dramatically different between k_{dd} and $k_{dd'}$. They may also have a consistence on the same base kernel, which means k_{dd} and $k_{dd'}$ share the same base kernel coefficient. Such coefficients diversities play a role in modeling the similarity heterogeneity of domains. We denote such a transfer covariance function as T_* , and a GP model using T_* as GP_{T_*} .

Even though a transfer covariance function permits certain design flexibility, it must be a valid kernel function so that the resulting GP model is applicable for any data. This is equivalent to guaranteeing that the corresponding covariance matrix is always positive semi-definite (PSD). In some scenarios, this can be achieved by exploiting prior knowledge on the application domain [19], [20]. However, in many real-world applications, such knowledge is unavailable or cannot be used in a feasible way. These cases demand a black-box learning approach that (i) accounts for the complete set of valid covariance functions (to avoid overlooking the adequate ones), (ii) enables interpretable domain similarities (so that problem insights can be inferred), and (iii) is computationally feasible (as the learning of GP is prone to computational issues). Although partial solution approaches exist in the literature, methods spanning all the three aspects are not well studied. Regarding T_λ , [13] and [15] focus on the Cholesky decomposition or an EM strategy to enforce a PSD but “free-form” similarity matrix. To regularize the similarities and ease the optimization, [14] proposes a sufficient condition for the similarity coefficient to handle the PSD problem. With this condition, T_λ can also be semantically associated with different transfer schemes. Likewise, for T_* , we are interested in exploring a practical condition with which the PSD problem is efficiently solved and T_* is semantically regularized. To be complete, we are dedicated to a necessary and sufficient condition.

In this paper, we investigate the transfer covariance function T_* which can model the similarity heterogeneity of domains. As stated above, T_* allows for both divergence and consistence on the coefficients of base kernels. The coefficient vectors of k_{dd} and of $k_{dd'}$ characterize T_* . The element-wise distance of these two coefficient vectors indicates the similarity of the source and target domains on the corresponding base kernel (we call it *sub-similarity of two domains*). With the black-box learning, our aim is to precisely capture the heterogeneous sub-similarities of two domains. We first propose a necessary and sufficient condition for T_* that embraces all the three aspects described above in a simple way. This condition describes a necessary and sufficient (and hence complete) relationship between the coefficient vectors of k_{dd} and $k_{dd'}$ which readily delivers an interpretation in terms of sub-similarities. Thus, we can exploit this relationship as a computationally inexpensive constraint in the learning of GP_{T_*} . Two instantiations of GP_{T_*} are developed, namely, $GP_{T_*}^c$ and $GP_{T_*}^p$. $GP_{T_*}^c$ uses a set of predefined constant base kernels, while $GP_{T_*}^p$ exploits a set of learnable parametric base kernels. We have conducted extensive experiments using artificial data and four real-world datasets to verify our investigations. More specifically, we first utilize 36 synthetic transfer tasks to demonstrate the capability of GP_{T_*} in the modeling of domain sub-similarities. We then show the transfer effectiveness of GP_{T_*} as compared to other state-of-the-art baselines on 12 real-world transfer tasks.

A preliminary version of this work [21] was published in ICDM 2018. Compared with the conference version, this work makes the following new contributions. Firstly, we add more

details to the conference version, including a more comprehensive related work, a new underlying intuition of our model, more complete technical details, and more analyses on the experimental studies. Secondly, we propose a new model with learnable parametric base kernels, which is significantly different from the one in the conference paper. We compare the new model with the conference one, and highlight the advantages of the new model. Finally, we conduct extensive experimental studies to investigate the transfer performance of our new model on 4 real-world datasets. We also analyze how the number of base kernels affect the transfer performance.

The rest of the paper is organized as follows. We discuss related work in Section 2. After that, we introduce the problem setting and some preliminaries in Section 3. Then, we elaborate on the transfer covariance function T_* in Section 4. In Section 5, we present a sufficient and necessary condition for GP_{T_*} in the black-box scenario. With the condition, we instantiate $GP_{T_*}^c$ and $GP_{T_*}^p$, and propose an inexpensive model learning strategy for both of them. The experimental study is described in Section 6. We finalize with some conclusions.

2 RELATED WORK

Covariance functions akin to T_* (T_λ) have been widely used for modeling problems in a variety of practical fields, such as geo-statistics [22], computer emulation in engineering [23], industrial robot dynamics [15], human physiology [24], sensor networks [19] or natural language processing [20]. One of the main challenges in the learning of such models is to ensure the validity of the covariance function. Note that any Gram matrix from T_* can be written as a sum of Schur products, each involving a matrix $\mathbf{K}_d$ for domains similarities and a matrix $\mathbf{K}_x$ for the input locations. Once the model has been learned, the $\mathbf{K}_d$ in each product term is kept fixed. Many approaches take advantage of this formulation.

In some cases, the learning task has been made easier by exploiting available problem information to confine the form of the transfer covariance function. For instance, in [19], a function similar to T_λ is used for a setting with multiple environmental domains, and a spherical parameterization is allotted to the modeling of $\mathbf{K}_d$. Alternatively, one can presume specific similarity values hold in such a way that positive semi-definiteness is guaranteed. This may be however at the expense of making the suitability of the represented domain dependencies more uncertain. So, in [20], different formulations of similarity coefficients offer mixed results on a T_λ -like function to model the judgements by multiple experts on the quality of text translations. In particular, making domains interdependent just through the sharing of the base kernel k (i.e. $\mathbf{K}_d$ is an identity matrix) performs better or worse than assuming a full correlation ($\mathbf{K}_d$ is an all-ones matrix) depending on whether *per-task* noise is adopted or not in the model. In this work, we cover scenarios where no reasonable assumptions can be made a-priori, nor enough information from the problem domain is available to determine the form of a valid T_* (T_λ). As described in the Introduction, this calls for learning approaches that consider the whole set of such valid functions, lead to interpretable domain similarities and do not pose computational overhead.

A number of works follow a constraint handling strategy which is explicitly applied during the optimization. More precisely, one direction is to formulate a constrained optimization problem which is then addressed with a metaheuristic algorithm. So, in [22], a T_* -like function minimizing a distance measure to an

empirical correlation matrix is sought with simulated annealing. To check the feasibility of a candidate solution, each $\mathbf{K}_d$ term in the associated sum of Schur products is tested for positive semidefiniteness. Provided that the covariance function on the inputs is PSD, this procedure guarantees the validity of the resulting transfer covariance function. However, the computation of the PSD tests can be cumbersome, e.g., typical approaches based on eigenvalues are subject to ill-conditioning issues and, if the number of domains is large, can be rather time consuming (for two domains, this should pose no problems though). Also, a common concern with metaheuristic approaches is that, often, their performance in terms of solution quality or runtime is greatly affected by how the various algorithm components are tuned. This is evinced in [22] in the disparity of fittings obtained with three cooling schedules which govern the stopping condition of the search. An alternative direction is taken in [13], [15], [25], where the constraint handling is embedded in the search by means of an Expectation Maximization algorithm. At each step, the estimates for all the base kernels in T_* (T_λ) are decoupled, and in each base kernel, the estimates for the expected data likelihood can be computed separately: first for the hyperparameters of the covariance function on the inputs, and then for the $\mathbf{K}_d$ matrix given the former. Further, the $\mathbf{K}_d$ is given in closed form and is PSD. Thus, the formulation is quite convenient, although the implementation of the expectations is somewhat involved. Another concern is that this method does not account for the interpretability of the $\mathbf{K}_d$ matrices being learned. Neither do the metaheuristic-based approaches, though they could be augmented with explicitly defined constraints for this. In this work, by contrast, we provide a characterization of the source-target similarities in T_* that readily delivers an interpretation of their relations with regard to each kernel component of the model. Moreover, as we demonstrate in the experiments, this characterization serves as a computationally inexpensive constraint that can be used with numerical optimization methods.

The direction followed by other works is to define an unconstrained parameterization of the $\mathbf{K}_d$ matrices that enforces their positive semidefiniteness, and subsequently apply the usual methods for optimization (e.g. maximizing log-likelihood). Since by using the Cholesky factorization, any PSD $\mathbf{K}_d$ can be written as $\mathbf{K}_d = \mathbf{L}\mathbf{L}^T$, where $\mathbf{L}$ is lower triangular, a popular choice is to reparameterize $\mathbf{K}_d$ on L [13], [24], [26]. Notice that this ensures the resulting transfer covariance function is valid, however it does not explain us whether all such valid functions are being considered or not (i.e. $\mathbf{K}_d$ could be non-PSD and the Schur product of $\mathbf{K}_d$ and a PSD $\mathbf{K}_x$ still be PSD). Also, a drawback of the Cholesky parameterization is the lack of a clear-cut relationship between a given $\mathbf{L}$ and the elements in $\mathbf{K}_d$. This hampers any control on the semantic meaning of the similarities during learning. Moreover, the problem is exacerbated by the fact that the factorization of a given $\mathbf{K}_d$ matrix may not be unique. Cao et al. [14], by contrast, enable an unconstrained parameterization for T_λ such that $\lambda \in [-1, 1]$, which facilitates its interpretation in terms of correlations between the source and target domains. However, only a sufficient condition for a PSD covariance function is provided, leaving the behavior of the model for other λ values unclear. Further, as opposed to T_* , T_λ cannot deal with the similarity heterogeneity of domains, which may jeopardize its capability of handling complex real-world transfer tasks. Recently, Wei et al. [6] investigate the extension of T_λ into the multiple-source transfer scenario and demonstrate a similar pathology. Wagle et

al. [27] further incorporate T_λ with cross-domain matrices [28] to construct a new PSD transfer kernel. In [29], the authors propose a fast and scalable GP model based on T_λ for the large source scenario. The characterization for T_* that we infer in this work is readily interpretable, and complete in that it provides a necessary and sufficient condition on the individual PSD $\mathbf{K}_d$ matrices.

3 PROBLEM SETTING AND PRELIMINARIES

We consider the transfer regression setting with one source domain and one target domain. We denote the source domain by $\mathcal{S}$ and assign to it a set of n_S data points $\mathbf{X}^S \in \mathbb{R}^{n_S \times \dim}$ with corresponding labels $\mathbf{y}^S \in \mathbb{R}^{n_S}$. Likewise, we represent the target domain as $\mathcal{T}$. Typically, only few labeled data points are available in $\mathcal{T}$. We denote the target data matrix by $\mathbf{X}^T \in \mathbb{R}^{n_T \times \dim}$ and the label vector by $\mathbf{y}^T \in \mathbb{R}^{n_T}$. Note that $n_T \ll n_S$ and $n_S + n_T = n$. Our objective is to utilize $\mathbf{X} = [\mathbf{X}^S; \mathbf{X}^T] \in \mathbb{R}^{n \times \dim}$ and $\mathbf{y} = [\mathbf{y}^S; \mathbf{y}^T] \in \mathbb{R}^n$ to build a model for $\mathcal{T}$.

In the standard regression setting, we assume an underlying latent function between the input $\mathbf{x}$ and the output y , and a Gaussian noise term with variance σ^2 . This latent function can be represented as an arbitrarily long vector $\mathbf{f}$ of values at given input data locations. A GP model defines a Gaussian distribution over the functions, $\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$, with mean vector $\boldsymbol{\mu}$ and covariance matrix $\mathbf{K}$. $\mathbf{K}$ must be *positive semi-definite* (PSD, or equivalently denoted as $\mathbf{K} \succeq \mathbf{0}$), and is given by a parameterized covariance function over the input space. Usually, $\boldsymbol{\mu} = \mathbf{0}$ is assumed, and so the GP model is completely specified by the covariance function.

A GP model for the transfer learning setting defines a joint Gaussian distribution over the space of source and target functions. Thus, the representation of the dependencies between the functions is ruled by the form of the transfer covariance function. The following transfer covariance function is used in [13], [15] for multi-task learning (*mtl*):

$$k_{mtl} = k_r(d, d')k(\mathbf{x}, \mathbf{x}'), \quad (1)$$

where $k_r(d, d')$ represents the similarity of domains, and $k(\cdot, \cdot)$ is any valid covariance function. To regularize $k_r(d, d')$, a transfer covariance function for two domains, T_λ , is proposed in [14]:

$$k_\lambda(\mathbf{x}, \mathbf{x}') = \begin{cases} \lambda k(\mathbf{x}, \mathbf{x}'), & \mathbf{x} \in \mathbf{X}^S \text{ \& \& } \mathbf{x}' \in \mathbf{X}^T \\ & \text{or } \mathbf{x} \in \mathbf{X}^T \text{ \& \& } \mathbf{x}' \in \mathbf{X}^S, \\ k(\mathbf{x}, \mathbf{x}'), & \text{otherwise,} \end{cases} \quad (2)$$

where λ is a coefficient measuring the similarity of the source and target domains. Compared with k_{mtl} in (1), k_λ in (2) fixes $k_r(d, d')$ to be 1 when $d = d'$. Further, $\lambda \in [-1, 1]$ enables a PSD T_λ while exposing more rationales. For the case $|\lambda| = 1$, T_λ can model the extreme transfer scenario in which the source and target domains are consistently related, and so all the source data is fully used in transfer. For the case $|\lambda| = 0$, T_λ models another extreme transfer scenario in which the source and target domains are completely unrelated, and thus only target data is used to avoid negative transfer. For the case $|\lambda| < 1$, T_λ fine-tunes the impact the source domain has on the target domain.

4 A TRANSFER COVARIANCE FUNCTION T_*

Even if semantically engaging, relying on a single λ hampers the capture of the similarity heterogeneity of domains.

To overcome the above, we may consider a general extension of (2) by using two different kernel functions:

$$k_g(\mathbf{x}, \mathbf{x}') = \begin{cases} k_{dd'}(\mathbf{x}, \mathbf{x}'), & \mathbf{x} \in \mathbf{X}^S \& \mathbf{x}' \in \mathbf{X}^T \\ & \text{or } \mathbf{x} \in \mathbf{X}^T \& \mathbf{x}' \in \mathbf{X}^S, \\ k_{dd}(\mathbf{x}, \mathbf{x}'), & \text{otherwise,} \end{cases} \quad (3)$$

where k_{dd} and $k_{dd'}$ can be different functions. Ideally, we would like to specify k_{dd} and $k_{dd'}$ so that k_g is a valid covariance function for any data and, at the same time, can model the similarity heterogeneity of domains. In this paper, we assume that k_{dd} and $k_{dd'}$ are in the same kernel function space expanded by a set of base kernels $k_i, i = 1, 2, \dots, N$ where k_i is a valid covariance function and N can be infinite. Such a transfer covariance function, T_* , falls within the *linear model of coregionalization* in geostatistics [17]:

$$k_*(\mathbf{x}, \mathbf{x}') = \begin{cases} \sum_i \beta_i k_i(\mathbf{x}, \mathbf{x}'), & \mathbf{x} \in \mathbf{X}^S \& \mathbf{x}' \in \mathbf{X}^T \\ & \text{or } \mathbf{x} \in \mathbf{X}^T \& \mathbf{x}' \in \mathbf{X}^S, \\ \sum_i \alpha_i k_i(\mathbf{x}, \mathbf{x}'), & \text{otherwise,} \end{cases} \quad (4)$$

where $\beta_i, i = 1, 2, \dots, N$ ($\alpha_i \geq 0, i = 1, 2, \dots, N$) are the combination coefficients for $k_{dd'}$ (k_{dd}). Note that we use the conical combination coefficients for the α_i s (i.e., $\alpha_i \geq 0$). We choose to do so as they are the similarity coefficients for the case that two domains are the same. Regarding the β_i s, we derive the condition that they should satisfy later on.

The intuition behind T_* is that the two different sets of coefficient variables, α_i s and β_i s, allow diverse similarities on the base kernels, and thus represent the heterogeneous sub-similarities between the source and target domains. Specifically, if β_i is close to α_i , the source domain is closely related to the target domain on the base kernel k_i . On the contrary, if β_i is very dissimilar to α_i , it means that the two domains differ considerably on the base kernel k_i . The similarity of β_i and α_i indicates how much knowledge can be transferred across domains on k_i .

5 TRANSFER MODEL USING T_*

With T_* we can model the similarity heterogeneity of domains. However, the learned GP_{T_*} should be a valid GP model for any data. This prevents β_i s and α_i s from taking arbitrary values. In this section, we provide a necessary and sufficient condition on the relationship of β_i s and α_i s that makes T_* (and hence GP_{T_*}) valid. We then introduce a learning rule for GP_{T_*} that exploits this relationship. Two instantiations of GP_{T_*} are also proposed.

5.1 On the Validity of GP_{T_*} for Any Data

To make T_* valid we must guarantee that a covariance matrix derived from it will be PSD. Theorem 1 facilitates this.

Theorem 1 A Gram matrix

$$\mathbf{K}_* = \begin{bmatrix} \sum_{i=1}^N \alpha_i \mathbf{K}_{SS}^i & \sum_{i=1}^N \beta_i \mathbf{K}_{ST}^i \\ \sum_{i=1}^N \beta_i \mathbf{K}_{TS}^i & \sum_{i=1}^N \alpha_i \mathbf{K}_{TT}^i \end{bmatrix}$$

with $\alpha_i > 0$ is PSD for any covariance matrices $\mathbf{K}^i$, in the form

$$\mathbf{K}^i = \begin{bmatrix} \mathbf{K}_{SS}^i & \mathbf{K}_{ST}^i \\ \mathbf{K}_{TS}^i & \mathbf{K}_{TT}^i \end{bmatrix},$$

if and only if $|\beta_i| \leq \alpha_i$ for $i = 1, 2, \dots, N$.

Proof 1 Necessary condition: We first rewrite $\mathbf{K}_*$ as:

$$\mathbf{K}_* = \sum_{i=1}^N \alpha_i \mathbf{K}_*^i, \quad (5)$$

where

$$\mathbf{K}_*^i = \begin{bmatrix} \mathbf{K}_{SS}^i & \beta_i / \alpha_i \mathbf{K}_{ST}^i \\ \beta_i / \alpha_i \mathbf{K}_{TS}^i & \mathbf{K}_{TT}^i \end{bmatrix}. \quad (6)$$

Since $\mathbf{K}_*$ is PSD for *any* covariance matrices $\mathbf{K}^i, i = 1, 2, \dots, N$ and $\alpha_i > 0$, we can derive

$$\mathbf{K}_*^i \succeq \mathbf{0}, i = 1, 2, \dots, N.$$

According to the Schur complement theorem [30], we derive:

$$\mathbf{K}_{TT}^i - (\beta_i / \alpha_i)^2 (\mathbf{K}_{ST}^i)^\top (\mathbf{K}_{SS}^i)^{-1} \mathbf{K}_{ST}^i \succeq \mathbf{0}. \quad (7)$$

Since $\mathbf{K}^i \succeq \mathbf{0}, i = 1, 2, \dots, N$, we have:

$$\mathbf{K}_{TT}^i - (\mathbf{K}_{ST}^i)^\top (\mathbf{K}_{SS}^i)^{-1} \mathbf{K}_{ST}^i \succeq \mathbf{0}.$$

We then rewrite Eq. (7):

$$\begin{aligned} & \mathbf{K}_{TT}^i - (\beta_i / \alpha_i)^2 \mathbf{K}_{TT}^i + (\beta_i / \alpha_i)^2 \mathbf{K}_{TT}^i \\ & - (\beta_i / \alpha_i)^2 (\mathbf{K}_{ST}^i)^\top (\mathbf{K}_{SS}^i)^{-1} \mathbf{K}_{ST}^i \succeq \mathbf{0}. \end{aligned}$$

By denoting $\mathbf{M} = \mathbf{K}_{TT}^i - (\mathbf{K}_{ST}^i)^\top (\mathbf{K}_{SS}^i)^{-1} \mathbf{K}_{ST}^i$, we have:

$$\mathbf{M} \succeq \mathbf{0}, \quad (8)$$

$$(1 - (\beta_i / \alpha_i)^2) \mathbf{K}_{TT}^i + (\beta_i / \alpha_i)^2 \mathbf{M} \succeq \mathbf{0}, \quad (9)$$

Since $\mathbf{K}_{TT}^i \succeq \mathbf{0}$ and Eq. (9) must hold for all $\mathbf{K}^i$, we induce:

$$(1 - (\beta_i / \alpha_i)^2) \geq 0, i = 1, 2, \dots, N,$$

and thus we have: $|\beta_i| \leq \alpha_i$ for $i = 1, 2, \dots, N$.

Sufficient condition: Similarly, we rewrite $\mathbf{K}_*$ as Eq. (5) with Eq. (6). We then consider:

$$\begin{aligned} \mathbf{N} &= \mathbf{K}_{TT}^i - (\beta_i / \alpha_i)^2 (\mathbf{K}_{ST}^i)^\top (\mathbf{K}_{SS}^i)^{-1} \mathbf{K}_{ST}^i \\ &= \mathbf{K}_{TT}^i - (\mathbf{K}_{ST}^i)^\top (\mathbf{K}_{SS}^i)^{-1} \mathbf{K}_{ST}^i \\ &+ (1 - (\beta_i / \alpha_i)^2) (\mathbf{K}_{ST}^i)^\top (\mathbf{K}_{SS}^i)^{-1} \mathbf{K}_{ST}^i \end{aligned}$$

Since $\mathbf{M} \succeq \mathbf{0}$ and $|\beta_i| \leq \alpha_i$ for $i = 1, 2, \dots, N$, we obtain $\mathbf{N} \succeq \mathbf{0}$. Considering $\mathbf{K}_{SS}^i \succeq \mathbf{0}$, we obtain $\mathbf{K}_*^i \succeq \mathbf{0}$ based on the Schur complement theorem. As the conical sum of PSD matrices is also PSD, we obtain that $\mathbf{K}_*$ is PSD.

To sum up, we conclude that, for $i = 1, 2, \dots, N$, if $\mathbf{K}_*$ is PSD with $\alpha_i > 0$ for any $\mathbf{K}^i$, the coefficients β_i s should satisfy $|\beta_i| \leq \alpha_i$.

Note that the proof of PSD $\mathbf{K}_*^i$ in the sufficient condition can also be obtained by using Theorem 1 in [14]. To be complete, we give the full proof. Moreover, when $\alpha_i = 0$, the corresponding base kernel k_i does not contribute to k_{dd} , and thus no knowledge on k_i can be transferred across domains. In this case, the coefficient for $k_{dd'}$, β_i , must be 0 too.

With Theorem 1, the PSD requirement is resolved and, at the same time, a semantic interpretation of T_* is readily obtained. Looking at each base kernel, Theorem 1 states that $|\beta_i|$ never goes beyond α_i . It is reasonable as α_i alludes to k_{dd} , which is used to calculate the distance of two instances from the same domain, and thus implies the maximum transfer capacity when the domains are different. The extreme case $|\beta_i| = \alpha_i$ indicates a full knowledge transfer on the base kernel k_i , while the opposite case $\beta_i = 0$ implies no knowledge is transferred on k_i . When

considering all the base kernels, we can see that T_* enables rich and diverse sub-similarities between the source and target domains. The extreme case $|\beta_i| = \alpha_i, i = 1, \dots, N$, only happens when the source and target domains are identical. The extreme case $|\beta_i| = 0, i = 1, \dots, N$, only happens when the two domains are completely unrelated. For other cases $|\beta_i| < \alpha_i, i = 1, \dots, N$, T_* gives a fine-grained modeling of the nuances between the source and target domains.

5.2 Further Intuition behind GP_{T_*}

We now investigate more intuitions behind GP_{T_*} . Given the inputs $\mathbf{X}$, GP_{T_*} defines a Gaussian distribution over the functions, $\mathbf{f}_* \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_*)$. Also, GP_{T_*} is a linear combination of several GPs, $GP_{T_*}^i, i = 1, 2, \dots, N$. Each $GP_{T_*}^i$ specifies $\mathbf{f}_*^i \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_*^i)$ with $\mathbf{K}_*^i$ defined in Eq. (6), and the corresponding combination coefficient is α_i . Comparing $GP_{T_*}^i$ with GP_{T_λ} , we can observe that $GP_{T_*}^i$ is a particular GP_{T_λ} with k_i as the kernel function and β_i/α_i as the similarity coefficient. As each $GP_{T_*}^i$ is modeling the sub-similarity for the base kernel k_i , GP_{T_*} captures the similarity heterogeneity of domains on these base kernels.

Furthermore, we may seek to improve the flexibility of T_λ by using the multiple kernel paradigm on the $k(\cdot, \cdot)$ as follows ($T_{\lambda_{mk}}$):

$$k_{\lambda_{mk}}(\mathbf{x}, \mathbf{x}') = \begin{cases} \lambda \sum_i \eta_i k_i(\mathbf{x}, \mathbf{x}'), & \mathbf{x} \in \mathbf{X}^S \text{ \& \& } \mathbf{x}' \in \mathbf{X}^T \\ & \text{or } \mathbf{x} \in \mathbf{X}^T \text{ \& \& } \mathbf{x}' \in \mathbf{X}^S, \\ \sum_i \eta_i k_i(\mathbf{x}, \mathbf{x}'), & \text{otherwise,} \end{cases} \quad (10)$$

where $k_i, i = 1, 2, \dots, N$ are base kernels in T_* and $\eta_i, i = 1, 2, \dots, N$ are the combination coefficients. From Eq. (10), we can infer that a GP using $T_{\lambda_{mk}}$ ($GP_{T_{\lambda_{mk}}}$) is also equivalent to a linear combination of several GPs, denoted as $GP_{T_{\lambda_{mk}}}^i, i = 1, 2, \dots, N$, where each $GP_{T_{\lambda_{mk}}}^i$ is a particular GP_{T_λ} with the kernel function as k_i and the similarity coefficient as λ . However, unlike GP_{T_*} , $GP_{T_{\lambda_{mk}}}$ assigns the same similarity coefficient λ to all $GP_{T_{\lambda_{mk}}}^i$ s. Thus, it fails to capture the similarity heterogeneity of the source and target domains. Actually, $T_{\lambda_{mk}}$ is a special case of T_* with $\beta_i = \lambda\alpha_i, i = 1, 2, \dots, N$.

5.3 Two instantiations of GP_{T_*}

In this section, we propose two instantiations of GP_{T_*} . Their main difference lies in the design of the base kernels. In the first instantiation, we predefine base kernels which are kept constant, i.e., for each base kernel k_i , the hyper-parameters inside, e.g., the length scale of the radial basis functions (RBF) kernel, are prefixed. Thus, the similarity coefficients, α_i and β_i , are the only free variables associated with k_i that need to be learned. We denote this instantiation of GP_{T_*} as $GP_{T_*}^c$.

In the second instantiation, we use a set of parametric base kernels from some primitive kernel families with simple functional forms. Specifically, we focus on the following three widely used primitive kernel families:

- Linear kernel: $LIN(\mathbf{x}, \mathbf{x}') = \sigma^2 \mathbf{x}^\top \mathbf{x}'$,
- Polynomial kernel: $POLY(\mathbf{x}, \mathbf{x}') = \sigma^2 (c + \mathbf{x}^\top \mathbf{x}')^d$,
- RBF kernel: $RBF(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2l^2})$,

as they express important structural motifs for GP [10]. Each kernel family is associated with a set of hyper-parameters, and instantiating the hyper-parameters gives a specific kernel. We treat these hyper-parameters as free variables, and jointly learn

them with the similarity coefficients α_i s and β_i s. Note that we allow multiple duplicates of the primitive kernel so that different hyper-parameters can be instantiated. We denote such a parametric instantiation of GP_{T_*} as $GP_{T_*}^p$. Compared with $GP_{T_*}^c$, $GP_{T_*}^p$ adds more flexibility on the selection of the base kernels, and thus may better capture the characteristics of the training data. We also note that $GP_{T_*}^p$ adds more free variables, i.e., the hyper-parameters of the base kernels, than $GP_{T_*}^c$, which may add to the computational complexity. However, as evidenced by the experimental studies, $GP_{T_*}^p$ can achieve better results than $GP_{T_*}^c$ by using less base kernels. In other words, $GP_{T_*}^p$ can outperform $GP_{T_*}^c$ by optimizing less similarity coefficients and base kernel hyper-parameters. This results in a superiority of $GP_{T_*}^p$ over $GP_{T_*}^c$ on both the transfer performance and the computation cost. In real-world applications, we suggest to use $GP_{T_*}^p$, especially when prior domain knowledge is not given in the initial stage.

5.4 Learning of GP_{T_*}

In this section, we concentrate on the learning of GP_{T_*} . Note that the learning procedure is applicable for both $GP_{T_*}^c$ and $GP_{T_*}^p$. By using Theorem 1 GP_{T_*} becomes valid for any data. Herein, we demonstrate how this theorem can be easily exploited for the learning of GP_{T_*} . Following [14], we only focus on the target task, and optimize the conditional distribution $p(\mathbf{y}^T | \mathbf{X}, \mathbf{y}^S)$. By denoting δ_S^2 and δ_T^2 as the noise variance for the source and target domains respectively, we have $p(\mathbf{y}^T | \mathbf{X}, \mathbf{y}^S) \sim \mathcal{N}(\boldsymbol{\mu}_*, \mathbf{C}_*)$ where

$$\begin{aligned} \boldsymbol{\mu}_* &= \left(\sum_{i=1}^N \beta_i \mathbf{K}_{TS}^i \right) \left(\sum_{i=1}^N \alpha_i \mathbf{K}_{SS}^i + \delta_S^2 \mathbf{I} \right)^{-1} \mathbf{y}^S, \\ \mathbf{C}_* &= \left(\sum_{i=1}^N \alpha_i \mathbf{K}_{TT}^i + \delta_T^2 \mathbf{I} \right) \\ &\quad - \left(\sum_{i=1}^N \beta_i \mathbf{K}_{TS}^i \right) \left(\sum_{i=1}^N \alpha_i \mathbf{K}_{SS}^i + \delta_S^2 \mathbf{I} \right)^{-1} \left(\sum_{i=1}^N \beta_i \mathbf{K}_{ST}^i \right). \end{aligned}$$

Then, the log-likelihood is defined as:

$$\mathcal{L}(\boldsymbol{\Theta}) = -\frac{1}{2} \ln |\mathbf{C}_*| - \frac{1}{2} (\mathbf{y}^T - \boldsymbol{\mu}_*)^\top \mathbf{C}_*^{-1} (\mathbf{y}^T - \boldsymbol{\mu}_*) - \frac{n}{2} \ln(2\pi)$$

where $\boldsymbol{\Theta} := \{\alpha_1, \dots, \alpha_N, \beta_1, \dots, \beta_N, \delta_S^2, \delta_T^2\}$ for $GP_{T_*}^c$ and $\boldsymbol{\Theta} := \{\alpha_1, \dots, \alpha_N, \beta_1, \dots, \beta_N, \boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_N, \delta_S^2, \delta_T^2\}$ for $GP_{T_*}^p$. Note that $\boldsymbol{\theta}_i$ represents the hyper-parameters associated with the base kernel k_i . We can use the standard conjugate gradient descent to do the optimization. The derivatives of $\mathcal{L}(\boldsymbol{\Theta})$ with respect to each variable is given by:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \theta_i} &= -\frac{1}{2} \text{tr}(\mathbf{C}_*^{-1} \frac{\partial \mathbf{C}_*}{\partial \theta_i}) \\ &\quad + \frac{1}{2} (\mathbf{y}^T - \boldsymbol{\mu}_*)^\top \mathbf{C}_*^{-1} \frac{\partial \mathbf{C}_*}{\partial \theta_i} \mathbf{C}_*^{-1} (\mathbf{y}^T - \boldsymbol{\mu}_*) \\ &\quad + \left(\frac{\boldsymbol{\mu}_*}{\partial \theta_i} \right)^\top \mathbf{C}_*^{-1} (\mathbf{y}^T - \boldsymbol{\mu}_*). \end{aligned}$$

The complete derivatives list can be found in the Appendix A. Note that we also need to take the relationship of α_i s and β_i s into account in the optimization. So, we use $\alpha_i \geq 0, i = 1, \dots, N$ and $|\beta_i| \leq \alpha_i, i = 1, \dots, N$ as constraints. Specifically, in the l -th iteration during learning, we calculate the updated α_i^l and use $\max(0, \alpha_i^l)$ to avoid negative values of α_i^l . Next we calculate the

Table 1
Model descriptions

Model	Description
GP_{T_*}	The general GP with the transfer covariance function T_* in Eq. (1).
$GP_{T_*}^c$	The instantiation of GP_{T_*} with predefined constant base kernels.
$GP_{T_*}^p$	The instantiation of GP_{T_*} with learnable parametric base kernels.
$GP_{T_{ar}}$	GP trained only using target labelled data.
GP_{All}	GP trained using the combination of source and target labelled data.
$GP_{T_{mk}}$	GP with the transfer covariance function T_{mk} in Eq. (10).
$GP_{T_{mk}^{ar}}$	GP with the transfer covariance function proposed in [27].

updated β_i^l , and if the constraint $|\beta_i| \leq \alpha_i$ is violated, we project β_i^l to a value close to the feasible boundary. Specifically, we set

$$\beta_i^l = \begin{cases} \alpha_i^l - \epsilon_i^l, & \text{if } \beta_i^l > \alpha_i^l, \\ -\alpha_i^l + \epsilon_i^l, & \text{if } \beta_i^l < -\alpha_i^l, \end{cases}$$

where ϵ_i^l is a variable with very small random value guaranteed to be within $[0, 2\alpha_i^l]$. We use ϵ_i^l to explore more diverse values of β_i^l in the search. Once we obtain the (sub)optimal Θ^{opt} , we can use them to predict the output value for a new test target point. The inference process is the same as in standard GPs. The prediction for the new target point $\mathbf{x}_{te}^T$ can be obtained by

$$y_{te}^T = [\mathbf{k}_{xS}^{opt}, \mathbf{k}_{xT}^{opt}] \mathbf{C}_*^{opt-1} [\mathbf{y}^S; \mathbf{y}^T],$$

where $\mathbf{k}_{xS}^{opt}$ is calculated using $\mathbf{x}_{te}^T$ and $\mathbf{X}^S$ with the optimal β_i^{opt} , $i = 1, 2, \dots, N$, and $\mathbf{k}_{xT}^{opt}$ is calculated using $\mathbf{x}_{te}^T$ and $\mathbf{X}^T$ with the optimal α_i^{opt} , $i = 1, 2, \dots, N$.

6 EXPERIMENTAL STUDIES

In the following experimental studies, we aim at (1) assessing the capability of GP_{T_*} to capture the sub-similarities between domains, and (2) evaluating its predictive effectiveness on new target test data. All the GP models herein are optimized using the conjugate gradient implementation from the *gpm1* package [31]. For each search, we empirically allow a maximum of 200 iterations. The reported results correspond to the model providing the best objective function value over 10 independent runs. Root mean squared error (RMSE) is used as the prediction evaluation metric. We test on one synthetic dataset and four real-world datasets. To be clear, Table 1 lists out the description of all the GP models used in the experiments.

Synthetic dataset. We artificially generate function values for the source and target domains based on a predefined T_* . We first generate target function values, and then generate the source function values given the generated target ones. To start, we predefine a set of base kernels k_i s, a set of ground-truth combination coefficients α_i^{gt} s for k_{dd} , and a set of ground-truth combination coefficients β_i^{gt} s for $k_{dd'}$. Given a set of inputs $\mathbf{X}$, we generate the target function values $\mathbf{f}_{tar}$ from a GP with zero mean and covariance matrix calculated using $k_{dd} = \sum_i \alpha_i^{gt} k_i$. Given $\mathbf{f}_{tar}$, we then generate the source function values $\mathbf{f}_{src}$ drawn from $p(\mathbf{f}_{src} | \mathbf{f}_{tar}, \mathbf{X})$. Note that $p(\mathbf{f}_{src} | \mathbf{f}_{tar}, \mathbf{X}) \sim \mathcal{N}(\boldsymbol{\mu}', \mathbf{C}')$ where $\boldsymbol{\mu}'$ and $\mathbf{C}'$ have the same form with $\boldsymbol{\mu}_*$ and $\mathbf{C}_*$ respectively. By varying α_i^{gt} s and β_i^{gt} s, we can control the ground-truth sub-similarity $sim_i = \beta_i^{gt} / \alpha_i^{gt}$ of domains on each base kernel k_i .

Wine Quality. The wine quality dataset includes red and white wine samples [32]. Physicochemical variables, e.g. volatile acidity, are used as features and the sensory variable is used as label.

Figure 1. (a) The red ‘o’s are the function values generated from GP using the exponential kernel $k_1 = 0.1 * \exp(-\frac{x^2}{2*1^2})$; the black ‘△’s are the function values generated from GP using the exponential kernel $k_2 = 0.01 * \exp(-\frac{x^2}{2*0.1^2})$; the blue ‘□’s are the function values generated from GP using covariance function $k = k_1 + k_2$. (b) The four special cases of source function values with $(\beta_1^{gt}, \beta_2^{gt})$ equaling to (1, 1), (1, 0), (0, 1), (0, 0), in which the source function values of $\beta_1^{gt} = 1, \beta_2^{gt} = 1$ are exactly the same with the target function values.

We use red wine (**R**) and white wine (**W**) as two domains, and construct two transfer tasks with the form $\mathcal{S} \rightarrow \mathcal{T}$: **R-W** and **W-R**.

Appliance Energy Prediction. This dataset includes the appliance energy use in different months [33]. The house conditions data, e.g. humidity, monitored by the sensors are used as features, and the amount of appliance energy use is used as label. We select data from two months (January and May) as two domains, and then form two transfer tasks **J-M** and **M-J**.

Air Quality. The air quality dataset [34] contains responses of gas multisensor devices from different seasons. The averaged responses from the metal oxide chemical sensors are used as features, and the ground-truth provided by a co-located reference certified analyzer is used as label. We use data from two seasons (Summer and Winter) as two domains, and then form two transfer tasks **S-W** and **W-S**.

UJIIndoorLoc. The building location dataset covers three buildings of Universitat Jaume I [35]. The received signal strength intensity from 520 wireless access points is used as the features, and the location represented by the latitude and longitude is taken as label. We build 3 domains by taking the location data from each building as a domain, denoted as $\mathbf{B}_0, \mathbf{B}_1, \mathbf{B}_2$. By pairing up two domains, we construct 6 transfer tasks, denoted as $\mathbf{B}_i - \mathbf{B}_j$, $i, j \in \{0, 1, 2\}$.

For the four real-world datasets, each domain has two sets of data points, set1 and set2, which are randomly selected. When a domain is selected as source, set1 is used for the source train. When a domain is selected as target, set2 is used for the target train and set1 is used for target test. In UJIIndoorLoc, set1 and set2 are of size 1000 and 25 respectively. In the rest of the datasets, set1 and set2 are of size 500 and 10 respectively.

6.1 Domain Sub-Similarity Capture

Firstly, we elucidate the capability of GP_{T_*} to capture the heterogeneous sim_i s between domains. In other words, we aim to analyze the distance between the learned sub-similarity $\beta_i^{opt} / \alpha_i^{opt}$ to the ground-truth similarity sim_i for all the base kernels. As in both $GP_{T_*}^c$ and $GP_{T_*}^p$, the coefficient of each base kernel serves to model the corresponding sub-similarity, we restrict ourselves to $GP_{T_*}^c$ in this set of experiments. We test on synthetic datasets and construct a variety of problems covering a broad spectrum of transfer problems. Specifically, we generate 500 data points as the

input $\mathbf{X} \in \mathbb{R}^{500 \times 1}$. We predefine two base kernels (k_1 and k_2) using two exponential covariance functions with different amplitudes and length scales. As shown in Figure 1(a), k_1 and k_2 have distinct characteristics. k_1 decides the ‘big trend’ while k_2 depicts the ‘small perturbation’. Next, we generate target and source function values based on k_* in Eq. (4). We fix $\alpha_1^{gt} = 1$ and $\alpha_2^{gt} = 1$, but vary β_1^{gt} and β_2^{gt} in a range set $\mathcal{V} = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$. Thus, the corresponding sim_i also varies in $\mathcal{V}$. We then generate $\mathbf{f}_{tar}$ from a GP using the covariance function $k_1 + k_2$. From Figure 1(a), we can see that the generated target function values possess the characteristics of both kernels. Based on $\mathbf{f}_{tar}$, for each combination $(\beta_1^{gt}, \beta_2^{gt})$ we generate source function values using $p(\mathbf{f}_{src} | \mathbf{f}_{tar}, \mathbf{X}) \sim \mathcal{N}(\boldsymbol{\mu}', \mathbf{C}')$. Figure 1(b) shows four special cases with $(\beta_1^{gt}, \beta_2^{gt})$ equaling to $(1, 1)$, $(1, 0)$, $(0, 1)$, $(0, 0)$. In total, we can generate $C_6^1 \times C_6^1 = 36$ different source functions, and thus we construct 36 transfer problems. For these 36 transfer problems, we use 380 data points as the source train data, 20 data points as the target train data, and the rest 100 data points as the target test data. Note that the function values of the target train and test are the same for the 36 transfer problems, but the function values of source train data vary in different transfer problems. We test $GP_{T_*}^c$ on these 36 transfer problems. Table 2 shows the learned sub-similarity $(\beta_1^{opt}/\alpha_1^{opt}, \beta_2^{opt}/\alpha_2^{opt})$ and the transfer performance. The result of GP using target train data only (GP_{Tar}) and results of $GP_{T_{\lambda mk}}$ are also shown as a reference. Note that GP_{Tar} is also implemented using multiple kernels.

Firstly, we pay attention to the domain sub-similarities capture. From Table 2, we can see that, overall, $GP_{T_*}^c$ successfully recover sim_i s on different problems. We show the distance of $(\beta_1^{opt}/\alpha_1^{opt}, \beta_2^{opt}/\alpha_2^{opt})$ to (sim_1, sim_2) more clearly in Figure 2(a). By setting a distance threshold as half of the step size of $\mathcal{V}$, which is 0.1, we obtain that for 32 out of the 36 problems, the distance is below the threshold. In the figure, the four cases in which the distance is above the threshold are pinpointed using the red arrow. This outcome indicates the effectiveness of $GP_{T_*}^c$ in capturing the heterogeneous sub-similarities of domains. On the other side, the single similarity coefficient learned by $GP_{T_{\lambda mk}}$ can accurately reflect the similarity value for the problems where $sim_1 = sim_2$. We highlighted these cases in bold in Table 2. However, for the problems $sim_1 \neq sim_2$, the learned similarity coefficient λ is usually between sim_1 and sim_2 . This indicates that $GP_{T_{\lambda mk}}$ fails to capture the similarity heterogeneity of two domains, and it can only learn a compromised similarity.

We now turn to the transfer performance. First of all, from Table 2 we observe that both $GP_{T_{\lambda mk}}$ and $GP_{T_*}^c$ can achieve adaptive transfer as they are not worse than the no-transfer method GP_{Tar} . To display the relative differences of $GP_{T_{\lambda mk}}$ and $GP_{T_*}^c$, we redraw their RMSE in Figure 2(b). As it can be noted, the circle diameters of the two methods are becoming smaller, which means the transfer performances are becoming better, from the left side to the right side. A similar trend is observed from the bottom to the top. This indicates that the two methods are transferring more knowledge as the two domains are becoming similar. The adaptiveness of the two methods is thus verified. Furthermore, we find that $GP_{T_*}^c$ achieves better results than $GP_{T_{\lambda mk}}$ as the red circles are lying inside the blue circles. In particular, $GP_{T_{\lambda mk}}$ yields similar results with $GP_{T_*}^c$ in the problems where $sim_1 = sim_2$. This corresponds to the diagonal cases in Figure 2(b). Relative significant improvements of $GP_{T_*}^c$ to $GP_{T_{\lambda mk}}$ can be observed on those non-diagonal cases. This indicates that $GP_{T_*}^c$ can better

Table 2
Comparison results of GP_{T_*} with $GP_{T_{\lambda mk}}$ ¹

Problem (sim ₁ ,sim ₂)	Learned Similarity		RMSE	
	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$
(0,0)	(0.057,0.008)	0.042	0.279	0.280
(0,0.2)	(0.087,0.174)	0.114	0.264	0.266
(0,0.4)	(0.015,0.443)	0.080	0.254	0.280
(0,0.6)	(0.080,0.623)	0.155	0.216	0.300
(0,0.8)	(0.079,0.805)	0.227	0.171	0.293
(0,1)	(0.015,1)	0.838	0.122	0.152
Problem (sim ₁ ,sim ₂)	Similarity		RMSE	
	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$
(0.2,0)	(0.247,0.056)	0.136	0.247	0.271
(0.2,0.2)	(0.190,0.231)	0.198	0.237	0.237
(0.2,0.4)	(0.195,0.364)	0.210	0.221	0.225
(0.2,0.6)	(0.214,0.616)	0.080	0.181	0.302
(0.2,0.8)	(0.2,0.763)	0.516	0.162	0.202
(0.2,1)	(0.184,1)	0.875	0.112	0.188
Problem (sim ₁ ,sim ₂)	Similarity		RMSE	
	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$
(0.4,0)	(0.375,0.039)	0.261	0.235	0.267
(0.4,0.2)	(0.438,0.172)	0.301	0.220	0.236
(0.4,0.4)	(0.420,0.446)	0.433	0.175	0.176
(0.4,0.6)	(0.383,0.644)	0.539	0.124	0.146
(0.4,0.8)	(0.410,0.763)	0.515	0.117	0.151
(0.4,1)	(0.686,1)	0.907	0.108	0.137
Problem (sim ₁ ,sim ₂)	Similarity		RMSE	
	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$
(0.6,0)	(0.644,0.078)	0.498	0.212	0.247
(0.6,0.2)	(0.611,0.173)	0.488	0.199	0.230
(0.6,0.4)	(0.580,0.413)	0.537	0.164	0.173
(0.6,0.6)	(0.578,0.565)	0.580	0.111	0.110
(0.6,0.8)	(0.583,0.760)	0.849	0.109	0.153
(0.6,1)	(0.741,1)	0.969	0.025	0.039
Problem (sim ₁ ,sim ₂)	Similarity		RMSE	
	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$
(0.8,0)	(0.793,0.032)	0.468	0.191	0.208
(0.8,0.2)	(0.807,0.254)	0.543	0.152	0.206
(0.8,0.4)	(0.757,0.426)	0.556	0.125	0.156
(0.8,0.6)	(0.792,0.611)	0.678	0.093	0.105
(0.8,0.8)	(0.798,0.800)	0.806	0.060	0.061
(0.8,1)	(0.768,1)	0.951	0.037	0.083
Problem (sim ₁ ,sim ₂)	Similarity		RMSE	
	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$	$GP_{T_*}^c$	$GP_{T_{\lambda mk}}$
(1,0)	(0.999,0.062)	0.826	0.074	0.120
(1,0.2)	(1,0.094)	0.801	0.076	0.153
(1,0.4)	(0.999,0.364)	0.984	0.066	0.081
(1,0.6)	(1,0.457)	0.937	0.068	0.075
(1,0.8)	(1,0.812)	0.981	0.041	0.048
(1,1)	(0.995,0.999)	1.000	0.001	0.001
GP_{Tar} RMSE			0.296	

¹ There are 6 sub-tables. In each sub-table, Column 1 shows 6 transfer problems with the ground-truth sub-similarities (sim_1, sim_2) . Column 2 shows $(\beta_1^{opt}/\alpha_1^{opt}, \beta_2^{opt}/\alpha_2^{opt})$ learned by $GP_{T_*}^c$, and column 3 shows λ learned by $GP_{T_{\lambda mk}}$. The last two columns show the transfer performance (RMSE) of $GP_{T_*}^c$ and $GP_{T_{\lambda mk}}$.

deal with complex transfer tasks than $GP_{T_{\lambda mk}}$. By checking the exact values of the results for the two methods in Table 2, we find that $GP_{T_*}^c$ consistently outperforms $GP_{T_{\lambda mk}}$. We believe that the performance improvement of $GP_{T_*}^c$ to $GP_{T_{\lambda mk}}$ is due to its capability of precisely capturing the sub-similarities of domains. More intuitively, as discussed in Section 5.2, both GP_{T_*} and $GP_{T_{\lambda mk}}$ are equivalent to a summation of GPs ($GP_{T_*}^i$ and $GP_{T_{\lambda mk}}^i$). As

Figure 2. (a) The red ‘o’s are (sim_1, sim_2) on k_1 and k_2 , and the blue ‘□’s are $(\beta_1^{opt}/\alpha_1^{opt}, \beta_2^{opt}/\alpha_2^{opt})$ learned by GP_{T^*} . (b) The diameter of the circle indicate the value of RMSE. The red ones are the results of GP_{T^*} , and the blue ones are the results of $GP_{T_{\lambda_{mk}}}$.

β_i/α_i is the ground-truth similarity on k_i , $GP_{T^*}^i$ should be no worse than $GP_{T_{\lambda_{mk}}}^i$ on the transfer performance. Thus, GP_{T^*} , more specifically $GP_{T^*}^c$ in these synthetic experiments, should be also no worse than $GP_{T_{\lambda_{mk}}}^p$.

6.2 Performance on Real-World Datasets

We now test the transfer performance of GP_{T^*} on the real-world transfer tasks. We compare GP_{T^*} with baselines GP_{Tar} , GP_{All} , $GP_{T_{\lambda_{mk}}}$, and $GP_{T_{\lambda_{var}}}$ [27]. Note that GP_{Tar} only utilizes the target training data to learn a model. On the contrary, GP_{All} fully uses all the source and target training data in the model learning. $GP_{T_{\lambda_{mk}}}$ is a GP with the transfer covariance function $T_{\lambda_{mk}}$ shown in Eq. (10). Note that we also implement GP_{Tar} and GP_{All} with multiple kernels. $GP_{T_{\lambda_{var}}}$ refers to a variant of the GP model in [14], and we follow the authors’ default setting. Regarding our GP_{T^*} , we implement both $GP_{T^*}^c$ and $GP_{T^*}^p$. For $GP_{T^*}^c$, we predefine 10 constant base kernels including 1 linear kernel, 2 polynomial kernels with degrees 2 and 3, and 7 RBF kernels with the length scale taking values in the set $\{0.01, 0.1, 1, 5, 10, 15, 20\}$. For $GP_{T^*}^p$, we allow 1 parametric linear kernel, 1 parametric polynomial kernel, and 3 parametric RBF kernels, in which the kernel hyper-parameters are to be learned. The evaluations are done on the 4 real-world datasets stated above, which amounts to 12 transfer tasks. The comparison results are shown in Figure 3.

From Figure 3, we can observe that both $GP_{T^*}^c$ and $GP_{T^*}^p$ consistently outperforms the other baselines on the transfer performance across all the transfer tasks. This showcases the effectiveness of GP_{T^*} to adaptively transfer knowledge in various

Figure 3. Comparison results on four real-world datasets.

real-world problems. More specifically, we find that GP_{Tar} is the worst in most transfer tasks as it does not transfer any knowledge from source domain. However, we also note that GP_{All} is even worse than GP_{Tar} in tasks **W-S** and **S-W**. This indicates us such a brute-force way of transferring all the source knowledge may even cause negative transfer. In contrast, $GP_{T_{\lambda_{mk}}}$ and $GP_{T_{\lambda_{var}}}$ consistently achieve positive transfer in all the transfer tasks. This is because both of them take the domain similarity into account in transfer, and thus can tune the shared knowledge and avoid negative transfer. However, they can only achieve a trade off in similarity heterogeneity. $GP_{T^*}^c$ and $GP_{T^*}^p$, by contrast, can explicitly model different sub-similarities, outperforming the other methods in all the transfer tasks.

Comparing $GP_{T^*}^c$ with $GP_{T^*}^p$, both of them select base kernels from the same set of primitive kernel families, i.e., Linear, Polynomial, and RBF. However, $GP_{T^*}^c$ uses 10 base kernels, while $GP_{T^*}^p$ only uses 5. With less base kernels, we observe that $GP_{T^*}^p$ generally performs better than $GP_{T^*}^c$. The performance difference stems from their different capacities on the data characteristic modelling. $GP_{T^*}^c$ predefines a limited set of constant base kernels, and each constant base kernel can only capture one specific type of data characteristic. Therefore, the data characteristic capture capacity of $GP_{T^*}^c$ is highly constrained by the number of constant base kernels used. In contrast, $GP_{T^*}^p$ utilizes parametric base kernels that are learned jointly with the transfer process, and thus adds the flexibility of the transfer covariance function in $GP_{T^*}^p$ on the modelling of data covariance. We add another level of flexibility by allowing 3 duplicates on the RBF family. Next, we discuss how the number of duplicate RBF kernels affects the final performance.

6.3 Performance with duplicate base kernels

We investigate how the transfer performance of $GP_{T^*}^p$ varies with the number of RBF parametric kernels used, i.e., n_{rbf} . We test $GP_{T^*}^p$ with $n_{rbf} = 1, \dots, 7$ on three transfer tasks, namely, **M-J**, **S-W**, and **B₁-B₂**. We also take the results of $GP_{T^*}^c$ with 10 constant base kernels as a reference. The results are shown in Figure 4.

As can be seen in Figure 4, the transfer performance of $GP_{T^*}^p$ generally becomes better with the increase of n_{rbf} as more base kernels allows more flexibility. However, the dependency is not strictly monotonic, but diversified among different transfer tasks. It is worth noting that $GP_{T^*}^p$ outperforms $GP_{T^*}^c$ from $n_{rbf} = 2$ on **M-J** and **B₁-B₂**, and sweeps on **S-W**. This indicates that, compared with $GP_{T^*}^c$, $GP_{T^*}^p$ can achieve better results with less free optimization variables, which demonstrates the superiority of

Figure 4. The RMSE results of $GP^P_{T_*}$ and $GP^C_{T_*}$ with the increase of the number of RBF base kernels used on four transfer tasks.

$GP^P_{T_*}$ to $GP^C_{T_*}$ on both transfer performance and computation complexity.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we focus on GPs using a transfer covariance function for regression in a black-box learning scenario. We investigate a family of general transfer covariance functions, T_* , that are capable of modeling the similarity heterogeneity of domains through multiple kernel learning. A necessary and sufficient condition that encodes the relationship of the coefficients for the base kernels is presented. With this condition, T_* is guaranteed to be a valid covariance function and is semantically regularized. We then instantiate $GP^C_{T_*}$ and $GP^P_{T_*}$. The former uses a set of predefined constant base kernels and the latter uses a set of learnable parametric base kernels. We propose a computationally inexpensive constraint for the learning of $GP^C_{T_*}$ and $GP^P_{T_*}$ with which we can explicitly capture the sub-similarities between domains. Extensive experiments on 36 synthetic transfer tasks and 12 real-world transfer tasks show the effectiveness of GP_{T_*} on capturing sub-similarities and on the transfer adaptiveness.

Note that we only consider different covariance calculations depending on whether two instances come from the same domain or not in the design of T_* . A more general way of taking the specific domain information into account, that is, differentiating covariance calculation using source-source, target-target, and source-target coefficients, is a promising research direction. Moreover, here GP is used as the base regression model because it can seamlessly be augmented with a transfer covariance function. Incorporating a transfer covariance function to some other regression models, e.g., SVR, linear regression, or LSTM, is also a research direction worth of further study.

ACKNOWLEDGMENTS

This research is partially supported by the National Research Foundation (NRF), Singapore, under the Singapore Data Science Consortium (SDSC) International Research Collaboration Grant No. SDSC-2020-004. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of NRF and SDSC. Yew-Soon Ong acknowledges the support of the Singtel Cognitive and Artificial Intelligence Lab for Enterprises (SCALE@NTU), which is a collaboration between Singapore Telecommunications

Limited (Singtel) and Nanyang Technological University (NTU) that is funded by the Singapore Government through the Industry Alignment Fund - Industry Collaboration Projects Grant.

APPENDIX A

THE DERIVATIVES OF LIKELIHOOD

The log-likelihood is:

$$\mathcal{L}(\Theta) = -\frac{1}{2} \ln |\mathbf{C}_*| - \frac{1}{2} (\mathbf{y}^T - \boldsymbol{\mu}_*)^T \mathbf{C}_*^{-1} (\mathbf{y}^T - \boldsymbol{\mu}_*) - \frac{n}{2} \ln(2\pi)$$

The derivatives of $L(\Theta)$ with respect to each variable are given below:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \Theta_i} &= -\frac{1}{2} \text{tr}(\mathbf{C}_*^{-1} \frac{\partial \mathbf{C}_*}{\partial \Theta_i}) \\ &\quad + \frac{1}{2} (\mathbf{y}^T - \boldsymbol{\mu}_*)^T \mathbf{C}_*^{-1} \frac{\partial \mathbf{C}_*}{\partial \Theta_i} \mathbf{C}_*^{-1} (\mathbf{y}^T - \boldsymbol{\mu}_*) \\ &\quad + (\boldsymbol{\mu}_*)^T \mathbf{C}_*^{-1} (\mathbf{y}^T - \boldsymbol{\mu}_*), \\ \frac{\partial \mathbf{C}_*}{\partial \alpha_i} &= \mathbf{K}_{\mathcal{T}\mathcal{T}}^i + \sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{T}\mathcal{S}}^i (\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I})^{-1} \mathbf{K}_{\mathcal{S}\mathcal{S}}^i \\ &\quad (\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I})^{-1} \sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{S}\mathcal{T}}^i, \\ \frac{\partial \mathbf{C}_*}{\partial \beta_i} &= -\mathbf{K}_{\mathcal{T}\mathcal{S}}^i (\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I})^{-1} \sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{S}\mathcal{T}}^i \\ &\quad - \sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{T}\mathcal{S}}^i (\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I})^{-1} \mathbf{K}_{\mathcal{S}\mathcal{T}}^i, \\ \frac{\partial \mathbf{C}_*}{\partial \delta_{\mathcal{S}}^2} &= \sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{T}\mathcal{S}}^i ((\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I})^{-1})^2 \sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{S}\mathcal{T}}^i, \\ \frac{\partial \boldsymbol{\mu}_*}{\partial \alpha_i} &= -\sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{T}\mathcal{S}}^i (\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I})^{-1} \mathbf{K}_{\mathcal{S}\mathcal{S}}^i \\ &\quad (\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I})^{-1} \mathbf{y}^{\mathcal{S}}, \\ \frac{\partial \boldsymbol{\mu}_*}{\partial \beta_i} &= \mathbf{K}_{\mathcal{T}\mathcal{S}}^i (\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I})^{-1} \mathbf{y}^{\mathcal{S}}, \\ \frac{\partial \boldsymbol{\mu}_*}{\partial \delta_{\mathcal{S}}^2} &= -\sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{T}\mathcal{S}}^i ((\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I})^{-1})^2 \mathbf{y}^{\mathcal{S}}. \end{aligned}$$

The derivative of $\mathbf{C}_*$ with respect to $\delta_{\mathcal{T}}^2$ is an identity matrix and the derivative of μ_* with respect to $\delta_{\mathcal{T}}^2$ is 0. Moreover, the derivatives with respect to the kernel hyper-parameters are:

$$\begin{aligned}\frac{\partial \mu_*}{\partial \theta_i} &= \left(\sum_{i=1}^N \beta_i \frac{\partial \mathbf{K}_{\mathcal{T}\mathcal{S}}^i}{\partial \theta_i} \right) \left(\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I} \right)^{-1} \mathbf{y}^{\mathcal{S}} \\ &\quad - \left(\sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{T}\mathcal{S}}^i \right) \left(\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I} \right)^{-1} \sum_{i=1}^N \alpha_i \frac{\partial \mathbf{K}_{\mathcal{S}\mathcal{S}}^i}{\partial \theta_i} \\ &\quad \left(\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I} \right)^{-1} \mathbf{y}^{\mathcal{S}} \\ \frac{\partial \mathbf{C}_*}{\partial \theta_i} &= \sum_{i=1}^N \alpha_i \frac{\partial \mathbf{K}_{\mathcal{T}\mathcal{T}}^i}{\partial \theta_i} - \left(\sum_{i=1}^N \beta_i \frac{\partial \mathbf{K}_{\mathcal{T}\mathcal{S}}^i}{\partial \theta_i} \right) \left(\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I} \right)^{-1} \\ &\quad \left(\sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{S}\mathcal{T}}^i \right) + \left(\sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{T}\mathcal{S}}^i \right) \left(\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I} \right)^{-1} \\ &\quad \sum_{i=1}^N \alpha_i \frac{\partial \mathbf{K}_{\mathcal{S}\mathcal{S}}^i}{\partial \theta_i} \left(\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I} \right)^{-1} \left(\sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{S}\mathcal{T}}^i \right) \\ &\quad - \left(\sum_{i=1}^N \beta_i \mathbf{K}_{\mathcal{T}\mathcal{S}}^i \right) \left(\sum_{i=1}^N \alpha_i \mathbf{K}_{\mathcal{S}\mathcal{S}}^i + \delta_{\mathcal{S}}^2 \mathbf{I} \right)^{-1} \left(\sum_{i=1}^N \beta_i \frac{\partial \mathbf{K}_{\mathcal{S}\mathcal{T}}^i}{\partial \theta_i} \right)\end{aligned}$$

REFERENCES

- [1] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.
- [2] H. S. Bhatt, A. Rajkumar, and S. Roy, "Multi-source iterative adaptation for cross-domain classification," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, ser. IJCAI'16. AAAI Press, 2016, pp. 3691–3697.
- [3] P. Wei, Y. Ke, and C. K. Goh, "Domain specific feature transfer for hybrid domain adaptation," in *Proceedings of International Conference on Data Mining*, 2017, pp. 1–5.
- [4] Z. Luo, Y. Zou, J. Hoffman, and L. F. Fei-Fei, "Label efficient learning of transferable representations across domains and tasks," in *Advances in Neural Information Processing Systems*, 2017, pp. 164–176.
- [5] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *Neural Networks, IEEE Transactions on*, vol. 22, no. 2, pp. 199–210, 2011.
- [6] P. Wei, R. Sagarna, Y. Ke, Y.-S. Ong, and C.-K. Goh, "Source-target similarity modelings for multi-source transfer gaussian process regression," in *Proceedings of the 34th International Conference on Machine Learning*. PMLR, 2017, pp. 3722–3731.
- [7] A. Tan Wei Min, R. Sagarna, A. Gupta, Y.-S. Ong, and C.-K. Goh, "Knowledge transfer through machine learning in aircraft design," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 48–60, 2017.
- [8] X. Xie, G. Liu, Q. Cai, P. Wei, and H. Qu, "Multi-source sequential knowledge regression by using transfer rnn units," *Neural Networks*, vol. 119, pp. 151–161, 2019.
- [9] W. Zellinger, T. Grubinger, M. Zwick, E. Lughofer, H. Schöner, T. Natschlager, and S. Saminger-Platz, "Multi-source transfer learning of time series in cyclical manufacturing," *Journal of Intelligent Manufacturing*, vol. 31, no. 3, pp. 777–787, 2020.
- [10] C. E. Rasmussen, *Gaussian Processes for Machine Learning*. MIT press, 2006.
- [11] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola, "Correcting sample selection bias by unlabeled data," in *Advances in Neural Information Processing Systems*, 2007, pp. 601–608.
- [12] S. Sun, "A survey of multi-view machine learning," *Neural Computing and Applications*, vol. 23, no. 7-8, pp. 2031–2038, 2013.
- [13] E. V. Bonilla, K. M. Chai, and C. Williams, "Multi-task gaussian process prediction," in *Advances in Neural Information Processing Systems 20*, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds. Curran Associates, Inc., 2008, pp. 153–160.
- [14] B. Cao, S. J. Pan, Y. Zhang, D.-Y. Yeung, and Q. Yang, "Adaptive transfer learning," in *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, ser. AAAI'10. AAAI Press, 2010, pp. 407–712.
- [15] K. M. Chai, C. Williams, S. Klanke, and S. Vijayakumar, "Multi-task gaussian process learning of robot inverse dynamics," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds. MIT Press, 2009, pp. 265–272.
- [16] P. Sollich and S. Ashton, "Learning curves for multi-task gaussian process regression," in *Advances in Neural Information Processing Systems*, 2012, pp. 1781–1789.
- [17] H. Wackernagel, *Multivariate Geostatistics: An Introduction with Applications*. Springer Science & Business Media, 1998.
- [18] M. Gönen and E. Alpaydm, "Multiple kernel learning algorithms," *Journal of Machine Learning Research*, vol. 12, no. 7, pp. 2211–2268, 2011.
- [19] M. A. Osborne, A. Rogers, S. D. Ramchurn, S. J. Roberts, and N. R. Jennings, "Towards real-time information processing of sensor network data using computationally efficient multioutput gaussian processes," in *Proceedings of the 7th International Conference on Information Processing in Sensor Networks*. IEEE, 2008, pp. 109–120.
- [20] T. Cohn and L. Specia, "Modelling annotator bias with multi-task gaussian processes: An application to machine translation quality estimation," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, ser. ACL, 2013, pp. 32–42.
- [21] P. Wei, R. Sagarna, Y. Ke, and Y. S. Ong, "Uncluttered domain sub-similarity modeling for transfer regression," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 1314–1319.
- [22] R. M. Lark and A. Papritz, "Fitting a linear model of coregionalization for soil properties using simulated annealing," *Geoderma*, vol. 115, pp. 245–260, 2003.
- [23] J. McFarland, S. Mahadevan, V. Romero, and L. Swiler, "Calibration and uncertainty analysis for computer simulations with multivariate output," *AIAA Journal*, vol. 46, no. 5, pp. 1253–1265, 2008.
- [24] R. Dürichen, M. A. F. Pimentel, L. Clifton, A. Schweikard, and D. A. Clifton, "Multitask gaussian processes for multivariate physiological time-series analysis," *IEEE Transactions on Biomedical Engineering*, vol. 62, no. 1, pp. 314–322, 2015.
- [25] H. Zhang, "Maximum-likelihood estimation for multivariate spatial linear coregionalization models," *Environmetrics*, vol. 18, no. 2, pp. 125–139, 2007.
- [26] K. M. Chai, "Multi-task learning with gaussian processes," Ph.D. dissertation, The University of Edinburgh, 2010.
- [27] N. Wagle and E. W. Frew, "Forward adaptive transfer of gaussian process regression," *Journal of Aerospace Information Systems*, vol. 14, no. 4, pp. 214–231, 2017.
- [28] A. Melkumyan and F. Ramos, "Multi-kernel gaussian processes," in *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011, pp. 1408–1403.
- [29] B. Da, Y.-S. Ong, A. Gupta, L. Feng, and H. Liu, "Fast transfer gaussian process regression with large-scale sources," *Knowledge-Based Systems*, vol. 165, pp. 208–218, 2019.
- [30] F. Zhang, *The Schur Complement and Its Applications*. Springer Science & Business Media, 2005, vol. 4.
- [31] C. E. Rasmussen and H. Nickisch, "Gaussian processes for machine learning (GPML) toolbox," *Journal of Machine Learning Research*, vol. 11, pp. 3011–3015, Dec. 2010.
- [32] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems*, vol. 47, no. 4, pp. 547–553, 2009.
- [33] L. M. Candanedo, V. Feldheim, and D. Deramaix, "Data driven prediction models of energy use of appliances in a low-energy house," *Energy and Buildings*, vol. 140, pp. 81–97, 2017.
- [34] S. De Vito, G. Fattoruso, M. Pardo, F. Tortorella, and G. Di Francia, "Semi-supervised learning techniques in artificial olfaction: A novel approach to classification problems and drift counteraction," *IEEE Sensors Journal*, vol. 12, no. 11, pp. 3215–3224, 2012.
- [35] J. Torres-Sospedra, R. Montoliu, A. Martínez-Usó, J. P. Avariento, T. J. Arnau, M. Benedito-Bordonau, and J. Huerta, "UJIIndoorloc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems," in *2014 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE, 2014, pp. 261–270.

Pengfei Wei received the Ph.D. degree in 2019, School of Computer Science and Engineering, Nanyang Technological University, Singapore. He is currently a research fellow in the school of Computing, National University of Singapore. His research interests include data analysis, transfer learning, domain adaptation, and reinforcement learning.

Ramon Sagarna received the Ph.D. degree in computer science from the University of the Basque Country, Spain. After working as senior research fellow at Nanyang Technological University, Singapore, he is currently a research scientist at the Institute for Infocomm Research (A*STAR), Singapore. His current research interests include transfer learning, metalearning, and algorithm selection for optimization and machine learning problems.

Ke Yiping is an assistant professor at the School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore. She obtained her Ph.D. degree in Computer Science from the Hong Kong University of Science and Technology in 2008 and B.Sc. degree in Computer Science from Fudan University in 2003. Her research lies broadly in Data Mining, Artificial Intelligence, and Data Management, with a focus on Graph Learning, Graph Analytics, Graph Mining, and Large Graph Data Manage-

ment. She has successfully rolled out technologies that advance the best practice in a variety of domains including engineering design, education, biology, etc.

Ong Yew-Soon received a Ph.D. degree in artificial intelligence for complex design from the University of Southampton, U.K., in 2003. He is a President's Chair Professor in Computer Science and serves as Director of the Singtel-NTU Cognitive & Artificial Intelligence Joint Lab at Nanyang Technological University. His research interest is in artificial intelligence. He serves as Chief Artificial Intelligence Scientist of the Agency for Science, Technology and Research Singapore and is founding Editor-in-Chief of the

IEEE Transactions on Emerging Topics in Computational Intelligence. He is listed as a Thomson Reuters highly cited researcher and among the World's Most Influential Scientific Minds.