JMLR: Workshop and Conference Proceedings 27:145–154, 2012 Workshop on Unsupervised and Transfer Learning

Self-measuring Similarity for Multi-task Gaussian Process

Kohei Hayashi HAYASHI.KOHEI@GMAIL.COM
Takashi Takenouchi TTAKASHI@IS.NAIST.JP
Graduate School of Information Science,
Nara Institute of Science and Technology,
8916-5 Takayama, Ikoma, Nara, 630-0192, Japan

Ryota Tomioka TOMIOKA@MIST.I.U-TOKYO.AC.JP
Hisashi Kashima KASHIMA@MIST.I.U-TOKYO.AC.JP
Department of Mathematical Informatics,
The University of Tokyo,
7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan

Editor: I. Guyon, G. Dror, V. Lemaire, G. Taylor, and D. Silver

Abstract

Multi-task learning aims at transferring knowledge between similar tasks. The multi-task Gaussian process framework of Bonilla *et al.* models (incomplete) responses of C data points for R tasks (e.g., the responses are given by an $R \times C$ matrix) by using a Gaussian process; the covariance function takes its form as the product of a covariance function defined on input-specific features and an inter-task covariance matrix (which is empirically estimated as a model parameter). We extend this framework by incorporating a novel similarity measurement, which allows for the representation of much more complex data structures. The proposed framework also enables us to exploit additional information (e.g., the input-specific features) when constructing the covariance matrices by combining additional information with the covariance function. We also derive an efficient learning algorithm which uses an iterative method to make predictions. Finally, we apply our model to a real data set of recommender systems and show that the proposed method achieves the best prediction accuracy on the data set.

Keywords: Multi-task Gaussian process, conjugate gradient, collaborative filtering.

1. Introduction

Multi-task learning (Caruana, 1997) is a machine learning framework that aims to improve performance through the learning of multiple tasks at the same time, and sharing the information of each task. An application of multi-task learning is a recommender system. For example, a recommender system of movies makes recommendations of movies which a user may prefer based on the history of the user's preferences. Since the total number of movies is much larger than the number of movies which one user has watched in the past, one user's preferences are insufficient for accurate prediction. Multi-task learning enhances the prediction performance by regarding each user as a relevant task and by sharing the preferences information of the users whose preferences are similar to each other. Such techniques are called collaborative filtering and are widely applied in recommender systems.

Several methods for multi-task learning have been proposed (Pan and Yang, 2010), which includes a method based on a Gaussian process (GP) called the multi-task GP (Bonilla et al., 2008). GP models can make predictions as a distribution, and they provide not only a mean of the prediction but also a variance, which can be used as a reliability of the prediction. Another advantage of the GP approach is that the learning of a model and prediction making are consistently done within the Bayesian framework. As the GP models represent the similarities of each data sample as a covariance matrix, the multi-task GP parametrizes the similarities between tasks and the similarities between data points as two independent covariance matrices, which enables to transfer knowledge among different tasks and data points efficiently. The multi-task GP is a special case of a tensor GP (Yu et al., 2007), and it has various applications such as a robotic manipulation (Chai et al., 2009).

However, if either inputs or task-specific features are not provided, the multi-task GP cannot measure the similarities and thus we cannot construct the corresponding covariance matrix with kernel functions. Since task-specific features are generally difficult to obtain, the method proposed by Bonilla et al. estimates a full covariance matrix over tasks in an empirical Bayesian framework. They also provide a parametric estimation procedure for the covariance with low-rank constraint. The method proposed by Yu et al. (2007) directly estimates the outputs of the GP as parameters, which can be seen as a matrix-factorization-like approximation.

These parametric approaches work well when the model includes the true distribution of the observations, e.g., the observed responses have a low-rank structure. However, such modeling is sometimes too restrictive for real data. In addition, when given responses are sparse and have high dimensionality, i.e., the dimension of the covariance matrix is large compared to the number of the observations, the empirical estimation of the full covariance matrix would be unstable, which may cause a negative effect for prediction.

Another challenge is to improve scalability. The naive computation of the mean of the predictive distribution requires $\mathcal{O}(M^3)$ complexity, where M denotes the number of observed responses. When applying the multi-task GP to a large-scale data set, it is inevitable to introduce some approximations such as limitations of kernel functions (e.g., linear kernel) and low-rank approximations of the Gram matrix. The empirical Bayes approach also raises further the computational cost; it is only applicable when the number of tasks or data points is small.

In this paper, we propose a new GP framework for multi-task learning problems. Our main contributions are as follows:

Self-measuring similarities We use the responses themselves to measure the similarities between the tasks and the data points. The response-based similarities allow for the flexible representation of more complex data structures. Additional features (e.g., input-specific features) would enhance the prediction accuracy but that is not a requirement in our framework.

Efficient and exact inference scheme We propose an efficient algorithm which computes the predictive mean of the GP with $\mathcal{O}(RC(R+C))$ computational cost without any approximations, where R and C denote the number of tasks and data points, respectively.

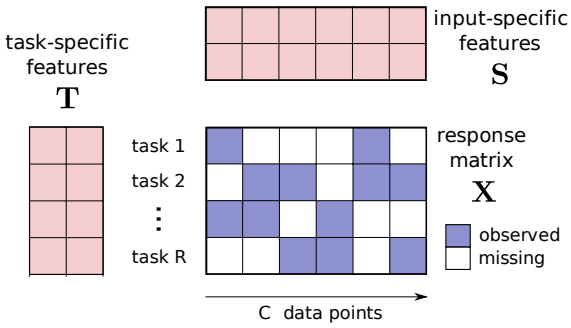

Figure 1: A problem setting of multi-task learning in this paper. Additional information $\mathbf{S}$ and $\boldsymbol{T}$ are not indispensable.

We evaluate our method in a collaborative filtering problem with a real data set and show that it attains the lowest prediction error.

2. Multi-task Gaussian Process

Suppose that we are given R tasks and the i-th task has incomplete outputs $\boldsymbol{x}_{i:} \in \mathbb{R}^C$ which may contain unknown values. Our purpose is to predict unobserved elements in a response matrix $\boldsymbol{X} = (\boldsymbol{x}_{1:}^\top, \ldots, \boldsymbol{x}_{R:}^\top)^\top \in \mathbb{R}^{R \times C}$ in which $\boldsymbol{x}_{i:}$ denotes the i-th row vector of $\boldsymbol{X}$. In some situations, additional information for data points $\mathbf{S} = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_C)$ and those for tasks $\boldsymbol{T} \equiv (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_R)$ are also given. Figure 1 is a summary of the problem setting, which is particularly called multi-label learning, a special case of multi-task learning.

2.1. Notations

We define the "vec" operator which creates a vector "vec $\boldsymbol{X}$" by stacking the column vectors of $\boldsymbol{X}$, i.e.,

$$\text{vec } \boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}_{:1} \\ \vdots \\ \boldsymbol{x}_{:C} \end{pmatrix}$$

where $\boldsymbol{x}_{:k}$ is the k-th column vector. We denote $\mathcal{I}$ as an index set of the observed elements, and we have $M = |\mathcal{I}|$ observations $\{x_{ik} | (i,k) \in \mathcal{I}\}$. We also denote $\boldsymbol{x}_\mathcal{I}$ as an M-dimensional vector which contains the observed elements of $\boldsymbol{X}$ in a certain order without overlapping. For later convenience, we introduce an observation matrix $\mathbf{P} \in \{0,1\}^{M \times RC}$ which removes the unobserved elements, i.e., $\boldsymbol{x}_\mathcal{I} = \mathbf{P}(\text{vec } \boldsymbol{X})$.

2.2. Modeling, Learning, and Predicting

The multi-task GP models $\boldsymbol{X}$ as

$$x_{ik} = m_{ik} + \mu + \varepsilon_{ik}, \quad \varepsilon_{ik} \sim N(0, \sigma^2) \tag{1}$$

where μ is a common bias and ε_{ik} is an i.i.d. spherical Gaussian noise. m_{ik} follows a tensor GP (Yu et al., 2007), which is defined as

$$m \sim \mathcal{GP}(0, \mathbf{\Sigma} \otimes \mathbf{\Omega}) \tag{2}$$

where $\otimes$ denotes the Kronecker product and both $\mathbf{\Sigma} \in \mathbb{R}^{C \times C}$ and $\mathbf{\Omega} \in \mathbb{R}^{R \times R}$ are symmetric and PSD. Each $\mathbf{\Sigma}$ and $\mathbf{\Omega}$ represents a covariance over columns and rows (i.e., data points and tasks), respectively. From the definition, the covariance between x_{ik} and x_{jl} is written as

$$\mathrm{cov}[x_{ik}, x_{jl}] = \Omega_{ij}\Sigma_{kl} + \delta_{ij}\delta_{kl}\sigma^2 \tag{3}$$

where δ_{ij} is the Kronecker's delta, i.e., $\delta_{ij} = 1$ if $i = j$ otherwise $\delta_{ij} = 0$. Equation (3) shows that the multi-task GP assumes the similarity (i.e., covariance) between the two responses x_{ik} and x_{jl} can be factorized into product of the task similarity Ω_{ij} and the data point similarity Σ_{kl}. The joint distribution of $\boldsymbol{X}$ is given by the Gaussian distribution

$$\int p(\boldsymbol{X} \mid \boldsymbol{M}, \mu, \sigma^2) p(\boldsymbol{M} \mid \mathbf{\Sigma}, \mathbf{\Omega}) \mathrm{d}\boldsymbol{M} = N(\mathrm{vec}\ \boldsymbol{X} \mid \boldsymbol{\mu}, \mathbf{g}) \tag{4}$$

where $\boldsymbol{\mu} = (\mu, \ldots, \mu) \in \mathbb{R}^{RC}$ and $\mathbf{g} = \mathbf{\Sigma} \otimes \mathbf{\Omega} + \sigma^2 \boldsymbol{I}$. Note that the multi-task GP is a special case of GP in which the covariance matrix has the structure of the Kronecker product.

By following the GP's framework, we need to determine the covariance matrices $\mathbf{\Sigma}$ and $\mathbf{\Omega}$ in a nonparametric way. On one hand, Bonilla et al. (2008) construct the covariance $\mathbf{\Sigma}$ via a covariance function $g(\cdot, \cdot)$ with the input-specific features $\mathbf{S}$ as

$$\Sigma_{kl} = g(\boldsymbol{s}_k, \boldsymbol{s}_l). \tag{5}$$

On the other hand, the task covariance $\mathbf{\Omega}$ is estimated as the empirical Bayesian method. We consider obtaining a maximum-likelihood solution $\hat{\mathbf{\Omega}}$ by maximizing the log-likelihood function of the observed elements, which is given by marginalizing out the unobserved elements from the joint distribution (4), i.e.,

$$\begin{aligned} &\ln \int N(\mathrm{vec}\ \boldsymbol{X} \mid \boldsymbol{\mu}, \mathbf{g}) \prod_{(i,k) \notin \mathcal{I}} \mathrm{d}x_{ik} \\ &= -\frac{1}{2}(\boldsymbol{x}_{\mathcal{I}} - \boldsymbol{\mu}_{\mathcal{I}})^\top \mathbf{g}_{\mathcal{I}}^{-1}(\boldsymbol{x}_{\mathcal{I}} - \boldsymbol{\mu}_{\mathcal{I}}) - \frac{1}{2}\ln\det|\mathbf{g}_{\mathcal{I}}| + \mathrm{const.} \end{aligned} \tag{6}$$

where $\boldsymbol{\mu}_{\mathcal{I}} = \mathbf{P}\boldsymbol{\mu}$ and $\mathbf{g}_{\mathcal{I}} = \mathbf{P}\mathbf{g}\mathbf{P}^\top \in \mathbb{R}^{M \times M}$ is a covariance matrix over the observed elements. The common bias μ is also estimated as a maximum-likelihood solution $\hat{\mu} = \frac{1}{M}\sum_{(i,k) \in \mathcal{I}} x_{ik}$.

Given the partially observed response matrix and the input-specific features $\mathcal{D} = \{\boldsymbol{X}, \mathbf{S}\}$, the multi-task GP predicts an unobserved response x_{ab} by a corresponding mean of the predictive distribution

$$\mathbb{E}[x_{ab} \mid \mathcal{D}, \hat{\mathbf{\Omega}}] = (\mathbf{g}_a \otimes \hat{\boldsymbol{\omega}}_b)_{\mathcal{I}}^\top \mathbf{g}_{\mathcal{I}}^{-1}(\boldsymbol{x}_{\mathcal{I}} - \hat{\boldsymbol{\mu}}_{\mathcal{I}}) + \hat{\mu} \tag{7}$$

where $\mathbf{g}_a = (k(\boldsymbol{s}_a, \boldsymbol{s}_1), \ldots, k(\boldsymbol{s}_a, \boldsymbol{s}_C))^\top$ and $\hat{\boldsymbol{\omega}}_b$ denotes the b-th row vector of $\hat{\mathbf{\Omega}}$. Since Equation (7) contains the inverse $\mathbf{g}_{\mathcal{I}}^{-1}$, the naive computational complexity of the predictive means is $\mathcal{O}(M^3)$.

3. Multi-task Gaussian Process with Self-measuring Similarity

Now we extend the multi-task GP model. First we introduce a way to construct the covariance matrices from the response matrix itself. Then we derive an efficient learning algorithm using the conjugate gradient method.

3.1. Self-measuring Similarities

We construct the covariance matrices by using the responses themselves:

$$\Omega_{ij} = k(\boldsymbol{x}_{i:}, \boldsymbol{x}_{j:}) \quad \text{and} \quad \Sigma_{kl} = g(\boldsymbol{x}_{:k}, \boldsymbol{x}_{:l}), \tag{8}$$

where k and g are arbitrary PSD kernel functions. As previously mentioned that $\boldsymbol{x}_{i:}$ denotes the i-th row vector and $\boldsymbol{x}_{:k}$ denotes k-th column vector of $\boldsymbol{X}$. We call this idea *Self-measuring Similarity*. The self-measuring similarity allows us to compute the covariance matrices without any additional information. The computation of the self-measuring similarity is simply done with evaluate the values of the covariance functions; it is much faster than the empirical Bayesian approach.

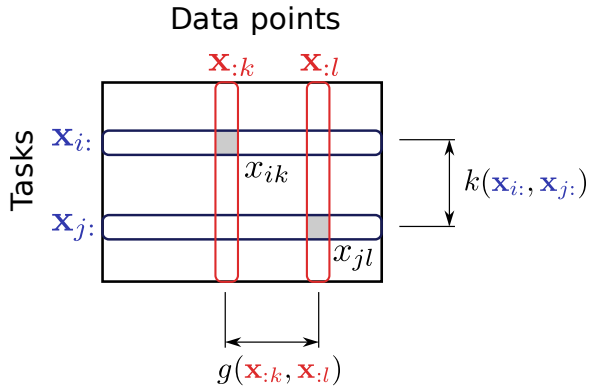

Figure 2: The idea of self-measuring similarities.

We introduce latent variables $\{z_{ik}\}$ for the missing elements $\{x_{ik}|(i,k) \notin \mathcal{I}\}$ to compute the kernel function in which the input contains missing values. After learning the latent variables, we use the completed matrix $\tilde{\boldsymbol{X}}$ as kernel inputs, where $\tilde{x}_{ik} = x_{ik}$ if $(i,k) \in \mathcal{I}$ otherwise $\tilde{x}_{ik} = z_{ik}$. We use the means of the predictive distribution as estimators of the latent variables, i.e., $\hat{z}_{ik} = \mathbb{E}[x_{ik}|\mathcal{D}]$ in a heuristic way.[1] Note that, however, the EM-like heuristic can be seen as an approximate marginalization (see Appendix A for more details).

Since the values of $\{\hat{z}_{ik}\}$ affect the predictive mean (7) through the kernel functions, the heuristic can be performed iteratively. Note that if the number of the missing elements (i.e. the latent variables) is larger than that of the observed elements, many iterations may cause over-fitting. We avoid this problem by early stopping of the iterations with a

1. One of the proper estimation methods for the latent variables is the empirical Bayesian. However, the optimization of the log-likelihood (4) with respect to $\{z_{ik}\}$ is computationally infeasible especially for large-scale data.

randomly picked validation set. For an initial value of $\hat{z}_{ik}$, we use a row-wise mean $\bar{x}_{i\cdot}$ or a column-wise mean $\bar{x}_{\cdot k}$:

$$\bar{x}_{i\cdot} = \frac{1}{M_{i\cdot}} \sum_{k,(i,k)\in\mathcal{I}} x_{ik}, \quad \bar{x}_{\cdot k} = \frac{1}{M_{\cdot k}} \sum_{i,(i,k)\in\mathcal{I}} x_{ik},$$

where $M_{i\cdot}$ and $M_{\cdot k}$ are the number of observed elements of $\boldsymbol{x}_{i:}$ and $\boldsymbol{x}_{:k}$, respectively.

If we have additional information such as input-specific features $\{\boldsymbol{s}_k | k = 1, \ldots, C\}$ for each data point, we exploit them by combining them with the self-measuring covariance function. For example, we extend the covariance function into a sum form

$$\Sigma_{kl} = g(\boldsymbol{x}_{:k}, \boldsymbol{x}_{:l}) + g'(\boldsymbol{s}_k, \boldsymbol{s}_l) \tag{9}$$

or a product form

$$\Sigma_{kl} = g(\boldsymbol{x}_{:k}, \boldsymbol{x}_{:l}) g'(\boldsymbol{s}_k, \boldsymbol{s}_l). \tag{10}$$

Note that if both g and g' are PSD, then the resulting kernel functions are still PSD (Rasmussen and Williams, 2006).

3.2. Computation for Prediction

As mentioned, we need to compute the inverse of $\mathbf{g}_{\mathcal{I}}$ in Equation (7) for the predictive means, and it requires $\mathcal{O}(M^3)$ computational cost. Instead, we solve the linear equation

$$\boldsymbol{x}_{\mathcal{I}} - \hat{\boldsymbol{\mu}}_{\mathcal{I}} = \mathbf{g}_{\mathcal{I}} \boldsymbol{\beta} \tag{11}$$

with respect to $\boldsymbol{\beta}$. Note that $\mathbf{g}_{\mathcal{I}}$ is positive definite when $\sigma^2 > 0$. After obtaining the solution $\hat{\boldsymbol{\beta}}$, we simply compute the predictive mean of x_{ab} as an inner product

$$\mathbb{E}[x_{ab} \mid \mathcal{D}, \hat{\boldsymbol{Z}}] = (\boldsymbol{g}_a \otimes \mathbf{g}_b)_{\mathcal{I}}^{\top} \hat{\boldsymbol{\beta}} + \hat{\mu}. \tag{12}$$

We solve Equation (11) using the conjugate gradient method (Shewchuk, 1994), which is an iterative method to solve a linear system in which the matrix is positive definite. Each iteration of the conjugate gradient needs to perform a matrix-vector multiplication; in our case that corresponds to the multiplication of $\mathbf{g}_{\mathcal{I}}$ and an M-dimensional vector, which requires $\mathcal{O}(M^2)$ computation and $\mathcal{O}(M)$ memory space.

The computational cost of the multiplication can still be reduced by exploiting a special structure in the matrix. Because of the structure of the Kronecker product in $\mathbf{g}_{\mathcal{I}}$, a multiplication of $\mathbf{g}_{\mathcal{I}}$ and an M-dimensional vector $\boldsymbol{v}$ is rewritten as

$$\begin{aligned} \mathbf{g}_{\mathcal{I}} \boldsymbol{v} &= \mathbf{P}(\boldsymbol{\Sigma} \otimes \boldsymbol{\Omega} + \sigma^2 \boldsymbol{I}) \mathbf{P}^{\top} \boldsymbol{v} \\ &= \mathrm{vec}\, \mathbf{P}(\boldsymbol{\Omega} \boldsymbol{V} \boldsymbol{\Sigma} + \sigma^2 \boldsymbol{V}) \end{aligned} \tag{13}$$

where $\boldsymbol{V} \in \mathbb{R}^{R \times C}$ is a matrix form of $\mathbf{P}^{\top} \boldsymbol{v}$, i.e., $\mathrm{vec}\, \boldsymbol{V} = \mathbf{P}^{\top} \boldsymbol{v}$. This technique, called *vec-trick* (Vishwanathan et al., 2007; Kashima et al., 2009), reduces the computational complexity from $\mathcal{O}(M^2)$ to $\mathcal{O}(RC(R + C))$.

Suppose we stop the iteration of the conjugate gradient when the ℓ_2 error of $\hat{\boldsymbol{\beta}}_l$ (i.e., between the solution at the l-th iteration and a true solution $\boldsymbol{\beta}_*$) is less than the error of the initial values $\hat{\boldsymbol{\beta}}_0$ with a tolerance ϵ, i.e., $||\boldsymbol{\beta}_* - \hat{\boldsymbol{\beta}}_l||_2 \leq \epsilon||\boldsymbol{\beta}_* - \hat{\boldsymbol{\beta}}_0||_2$. Then the maximum number of iterations is bounded

$$l \leq \frac{1}{2}\sqrt{\kappa}\log\left(\frac{2}{\epsilon}\right) \tag{14}$$

where κ is the condition number of $\mathbf{g}_{\mathcal{I}}$, which is defined as the ratio of the maximum and the minimum eigenvalue of $\mathbf{g}_{\mathcal{I}}$. The total cost for obtaining the solution $\hat{\boldsymbol{\beta}}$ is $\mathcal{O}(\sqrt{\kappa}RC(R+C))$.

With an observation rate α, the number of observations has a relation $M = \alpha RC$. If $\boldsymbol{X}$ is nearly square, i.e., $R \simeq C$, then the computational complexity can be rewritten as $\mathcal{O}((\sqrt{\kappa}/\alpha^{\frac{3}{2}})M^{\frac{3}{2}})$, which is much faster than the naive complexity $\mathcal{O}(M^3)$. Finally we summarize the entire algorithm as a pseudo code in Algorithm 1.

Algorithm 1: Computation of predictive means with conjugate gradient.

1. Initialize $\hat{\boldsymbol{Z}}_0$ with row-means or column-means of $\boldsymbol{X}$
2. For $l = 1$ to maximum number of iterations
   (a) Construct $\boldsymbol{\Sigma}$ and $\boldsymbol{\Omega}$ with $\hat{\boldsymbol{Z}}_{l-1}$ and additional features
   (b) Solve $\boldsymbol{x}_{\mathcal{I}} = \mathbf{g}_{\mathcal{I}}\boldsymbol{\beta}$ by the conjugate gradient with a tolerance ϵ
   (c) Compute predictive means $\{\mathbb{E}[x_{ab} \mid \mathcal{D}, \hat{\boldsymbol{Z}}_{l-1}] \mid (a,b) \notin \mathcal{I}\}$ of unobserved elements
   (d) Construct $\hat{\boldsymbol{Z}}_l$ from the predictive means
3. Return $\hat{\boldsymbol{Z}}_l$

4. Experimental Results

We evaluate the proposed method by applying it to a collaborative filtering problem. We use the Movielens 100k data set[2], which contains 100,000 ratings $x_{ik} \in \{1,2,3,4,5\}$ for 1,682 movies (data points) labeled by 943 users (tasks). The observation ratio α is $\simeq 0.06$. The data set contains user-specific features § (e.g., age, gender, ...) and movie-specific features $\boldsymbol{T}$ (release date, genre, ...). The data set provides 90,570 ratings for training and remaining 9,430 ratings for testing. After learning with the training data set, we evaluate the root-mean-squared-error (RMSE) for the testing data set. All experiments are done with a Xeon 2.93 GHz 8 core machine.

In the experiment, we prepare three forms of covariance functions: a covariance measured by the user-specific and the movie-specific features ("Feature"), a self-measuring covariance ("Self-measuring"), and a combination of them with the product form (10) ("Product"). Note that our model with "Feature" setting is equivalent to the kernel method proposed by Bonilla et al. (2007). We summarize the covariance functions in Table 1. We use the RBF kernel $k(\boldsymbol{x}, \boldsymbol{x}') = g(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\lambda\,||\boldsymbol{x} - \boldsymbol{x}'||^2)$ for the covariance functions. We

2. http://www.grouplens.org/node/73

Table 1: Settings of the covariance functions.

	Feature	Self-measuring	Product
Ω_{ij}	$k(\boldsymbol{s}_i, \boldsymbol{s}_j)$	$k(\boldsymbol{x}_{i:}, \boldsymbol{x}_{i:})$	$k(\boldsymbol{x}_{i:}, \boldsymbol{x}_{i:})k(\boldsymbol{s}_i, \boldsymbol{s}_j)$
Σ_{kl}	$g(\boldsymbol{t}_k, \boldsymbol{t}_l)$	$g(\boldsymbol{x}_{:k}, \boldsymbol{x}_{:l})$	$g(\boldsymbol{x}_{:k}, \boldsymbol{x}_{:l})g(\boldsymbol{t}_k, \boldsymbol{t}_l)$

Table 2: RMSEs on the Movielens 100k dataset. Lower RMSE indicates higher prediction performance. (Left) The RMSE behaviour of the EM-like heuristic. l indicates the number of iterations. (Right) Existing methods v.s. proposed method with early stopping.

l	Feature	Self-measuring	Product
1	1.0517	0.9431	0.9393
2	–	0.9276	0.9231
3	–	0.9329	0.9292
4	–	0.9439	0.9410

Method	RMSE	time
User-KNN	0.9507	7s
Movie-KNN	0.9354	42s
Matrix Factorization	0.9345	1m38s
Feature	1.0517	7m01s
Self-measuring	0.9308	16m22s
Product	**0.9256**	18m25s

choose the hyper-parameters as $(\sigma^2, \lambda) = (0.5, 0.001)$ for "Feature" and $(\sigma^2, \lambda) = (0.1, 0.1)$ for {"Self-measuring", "Product"}, selected by three-fold cross validation from candidates $\sigma^2 \in \{1, 0.5, 0.1, 0.05\}$ and $\lambda \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$. We set the tolerance of the conjugate gradient ϵ as 10^{-3}. As the initial values of $\{z_{ik}\}$ we use the row-mean for $\boldsymbol{\Sigma}$ and the column mean for $\boldsymbol{\Omega}$.

For fair comparison, the number of the EM-like iteration is determined by early stopping with a validation set randomly drawn 5% of the training set. We compare with standard methods for recommendation system which includes user- and movie-based K-nearest neighbour (KNN) with the Pearson correlation and matrix factorization (see Su and Khoshgoftaar (2009) for more details.) We use `MyMediaLite`[3] as these implementations and set the hyper-parameters by following the examples specially recommended for the Movielens 100k dataset.

We summarize the prediction errors in Table 2. The result shows that "Feature" is the worst performance; it suggests that the additional information-based similarity is not enough to capture the observed data structure, and the self-measuring similarity is much more important for the prediction. Nevertheless, the combination with the additional information ("Product") further improves the prediction performance compared to using the self-measuring alone ("Self-measuring"). The left panel of Table 2 shows the EM-like heuristic drastically improves the prediction accuracy at the second iteration, while the third or further iterations produce worse results. The best score ("Product") in the right panel of Table 2 is also the best over other 76 methods listed in mlcomp.org[4] as of May, 2011.

3. http://www.ismll.uni-hildesheim.de/mymedialite
4. http://mlcomp.org/datasets/341

5. Conclusion

In this paper, we have presented a new framework to solve multi-task problems by using a Gaussian process based on self-measuring similarities. We proposed the efficient algorithm based on the conjugate gradient method with the vec-trick. Our method achieved the best score of the Movielens 100k data set.

Acknowledgments

We thank the anonymous reviewers for insightful comments and Mauricio Alexandre Parente Burdelis for helpful advices.

References

Edwin V. Bonilla, Felix V. Agakov, and Christopher K. I. Williams. Kernel multi-task learning using task-specific features. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*. Omnipress, March 2007. URL aistats07.pdf.

Edwin V. Bonilla, Kian M. Chai, and Christopher K. I. Williams. Multi-task Gaussian process prediction. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*. MIT Press, Cambridge, MA, 2008. URL nips08.pdf.

Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, July 1997. ISSN 08856125. doi: 10.1023/A:1007379606734. URL http://dx.doi.org/10.1023/A:1007379606734.

Kian M. Chai, Christopher Williams, Stefan Klanke, and Sethu Vijayakumar. Multi-task gaussian process learning of robot inverse dynamics. In *NIPS 2008*, http://eprints.pascal-network.org/archive/00004640/, 2009.

Hisashi Kashima, Tsuyoshi Kato, Yoshihiro Yamanishi, Masashi Sugiyama, and Koji Tsuda. Link propagation: A fast semi-supervised learning algorithm for link prediction. In *SDM*, pages 1099–1110. SIAM, 2009.

Sinno J. Pan and Qiang Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, October 2010. ISSN 1041-4347. doi: 10.1109/TKDE.2009.191. URL http://dx.doi.org/10.1109/TKDE.2009.191.

Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, November 2006. ISBN 026218253X. URL http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/026218253X.

J. R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical report, Carnegie Mellon University, Pittsburgh, PA, USA, 1994. URL http://portal.acm.org/citation.cfm?id=865018.

Xiaoyuan Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009:1–19, January 2009. ISSN 1687-7470. URL http://dx.doi.org/10.1155/2009/421425.

S. V. N. Vishwanathan, Karsten M. Borgwardt, and Nicol N. Schraudolph. Fast computation of graph kernels. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1449–1456. MIT Press, Cambridge, MA, 2007.

Kai Yu, Wei Chu, Shipeng Yu, Volker Tresp, and Zhao Xu. Stochastic relational models for discriminative link prediction. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1553–1560. MIT Press, Cambridge, MA, 2007.

Appendix A. Interpretation of EM-like Heuristic

The EM-like heuristic can be interpreted as an approximation of the marginalization of the predictive distribution with respect to $\boldsymbol{Z}$. First, given estimated latent variables $\hat{\boldsymbol{Z}}_{l-1}$, we consider to use the predictive distribution $p(\boldsymbol{X}|\mathcal{D}, \hat{\boldsymbol{Z}}_{l-1})$ as a posterior of $\boldsymbol{Z}$, i.e., $p(\boldsymbol{Z}|\mathcal{D}) \approx p(\boldsymbol{X}|\mathcal{D}, \hat{\boldsymbol{Z}}_{l-1})$. If we further approximate the posterior distribution as a delta function $\Delta(\boldsymbol{Z} - \mathbb{E}[\boldsymbol{X}|\mathcal{D}, \hat{\boldsymbol{Z}}_{l-1}])$, then we have

$$\int \mathbb{E}[\boldsymbol{X} \mid \mathcal{D}, \boldsymbol{Z}] p(\boldsymbol{Z} \mid \mathcal{D}) \mathrm{d}\boldsymbol{Z} \approx \int \mathbb{E}[\boldsymbol{X} \mid \mathcal{D}, \boldsymbol{Z}] \Delta(\boldsymbol{Z} - \mathbb{E}[\boldsymbol{X}|\mathcal{D}, \hat{\boldsymbol{Z}}_{l-1}]) \mathrm{d}\boldsymbol{Z} = \hat{\boldsymbol{Z}}_l$$

Note that $(\hat{\boldsymbol{Z}}_l)_{ab}$ is also a predictive mean for an observation x_{ab}.