Multitask Learning for Protein Subcellular Location Prediction

Qian Xu, Sinno Jialin Pan, Hannah Hong Xue, and Qiang Yang

Abstract—Protein subcellular localization is concerned with predicting the location of a protein within a cell using computational methods. The location information can indicate key functionalities of proteins. Thus, accurate prediction of subcellular localizations of proteins can help the prediction of protein functions and genome annotations, as well as the identification of drug targets. Machine learning methods such as Support Vector Machines (SVMs) have been used in the past for the problem of protein subcellular localization, but have been shown to suffer from a lack of annotated training data in each species under study. To overcome this data sparsity problem, we observe that because some of the organisms may be related to each other, there may be some commonalities across different organisms that can be discovered and used to help boost the data in each localization task. In this paper, we formulate protein subcellular localization problem as one of *multitask learning* across different organisms. We adapt and compare two specializations of the multitask learning algorithms on 20 different organisms. Our experimental results show that multitask learning performs much better than the traditional single-task methods. Among the different multitask learning methods, we found that the multitask kernels and supertype kernels under multitask learning that share parameters perform slightly better than multitask learning by sharing latent features. The most significant improvement in terms of localization accuracy is about 25 percent. We find that if the organisms are very different or are remotely related from a biological point of view, then jointly training the multiple models *cannot* lead to significant improvement. However, if they are closely related biologically, the multitask learning can do much better than individual learning.

Index Terms—Protein subcellular localization; multitask learning.

1 Introduction

Organelles with different functions are specialized subunits in a cell. Most organelles are closed compartments separated by lipid membranes. The knowledge of the subcellular localization of proteins is important because it can 1) provide useful insights about their functions, 2) indicate how and in what kind of cellular environments they interact with each other and with other molecules, and 3) help us understand the intricate pathways that regulate biological process at the cellular level [1]. Thus, protein subcellular localization is crucial for genome annotations, protein function prediction, and drug discovery [2]. Proteins perform their appropriate functions as, and only when, they are located in the correct subcellular compartments. Take prokaryotic and eukaryotic proteins as examples. For prokaryotes, many proteins that are synthesized in the cytoplasm are ultimately found in noncytoplasmic locations [3], such as cell membranes or extracellular environments, while most eukaryotic proteins are encoded in the nuclear and transported to the cytosol for further synthesis. Due to the importance of protein subcellular localization, considerable attention has been drawn [4], [5], [6], [7], [8].

The annotations of protein subcellular localization can be detected by various biochemical experiments, such as cell fractionation, electron microscopy, and fluorescence microscopy. However, the purely experimental approaches are time-consuming and expensive, and as a result, available data are rare and sparse. Therefore, a large number of computational methods were developed in an attempt to predict protein subcellular locations accurately and automatically [1], [9], [10], [11], [12], [13], [14], [15]. Prediction-based techniques have a long history in bioinformatics, which, in many cases, can nicely supplement wet lab experiments. Examples of successful prediction techniques and their corresponding biological studies can be found, for example, in [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29]. A recent review summarized the state of the art in prediction-based methods for both basic research and applications [30].

However, using sparse and a small quantity of data for prediction can only give us low accuracy. In general, the lack of high-quality labeled data is a major problem in bioinformatics. According to the Swiss-Prot database version 50.0 released on 30 May 2006, the number of protein sequences with localization annotations occupies only about 14 percent of total eukaryotic protein entries [31]. Despite this difficulty, we observe that there exist protein databases with subcellular localization annotations from multiple organisms, some of which are more related to each other than others. These observations motivate us to explore whether it is possible to propagate the annotated knowledge across different organisms to benefit their

- Q. Xu is with the Bioengineering Program, Hong Kong University of Science and Technology (HKUST), Clearwater Bay, Kowloon, Hong Kong. E-mail: fleurxq@ust.hk.
- S.J. Pan and Q. Yang are with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology (HKUST), Clearwater Bay, Kowloon, Hong Kong. E-mail: {sinnopan, qyang}@ust.hk.
- H.H. Xue is with the Department of Biochemistry, Hong Kong University of Science and Technology (HKUST), Clearwater Bay, Kowloon, Hong Kong. E-mail: hxue@ust.hk.

Manuscript received 26 Nov. 2009; revised 1 Feb. 2010; accepted 7 Mar. 2010; published online 6 Apr. 2010.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-2009-11-0216.
Digital Object Identifier no. 10.1109/TCBB.2010.22.

1545-5963/11/$26.00 © 2011 IEEE Published by the IEEE CS, CI, and EMB Societies & the ACM

prediction. Note that proteins may simultaneously exist at or move between two or more different subcellular locations. Several web servers Hum-mPLoc [32], Euk-mPLoc [15], and Cell-PLoc [1] took multiplex proteins into account when predicting protein subcellular localization. However, in this paper, we do not consider the existence of multiplex proteins.[1]

Traditionally, classification models in machine learning are constructed based on the data from each organism individually. Take Cell-PLoc [1] as an example. This package contains the following six predictors: Euk-mPLoc, Hum-mPLoc, Plant-PLoc, Gpos-PLoc, Gneg-PLoc, and Virus-PLoc, which are specialized for eukaryotic, human, plant, Gram-positive bacteria, Gram-negative bacterial, and viral proteins, respectively. There is much common knowledge that are shared among them, especially species that are of the same types. In this work, we formulate the knowledge-sharing process under a multitask learning framework [33].

In machine learning community, it has been proved empirically and theoretically that learning tasks with few annotated data simultaneously can lead to better performance than learning the models independently, when the tasks are related to each other in some sense [34], [35], [36], [37], [38]. In this work, we answer two related questions:

1. Biologically, is it feasible to apply multitask learning to allow common knowledge in related species to benefit each other?
2. Computationally, which method in multitask learning (parameter sharing versus latent feature sharing) is more useful in subcellular localization?

In methodology, we examine two prominent multitask learning methods in the context of protein subcellular localization across different organisms. The first method is to find out the commonality among the parameters of different models for different data [39], and the second method is to discover common latent features that are shared among different tasks [40]. While each method has their own advantages, for the protein subcellular localization problem, it has not been clear which one is more advantageous. To highlight the biological significance, we test the belief that biologically related species are more likely to help each other in the subcellular localization task, which has been used as an intuition that has never been verified before. In this paper, we empirically compare these different methods under two multitask learning frameworks and other popular machine learning baselines, and evaluate the aforementioned hypotheses.

The rest of the paper is organized as follows: In Section 2, we will briefly review related works in the past. In Section 3, we introduce two multitask learning frameworks and their variations. In Section 4, we will describe the experimental design and analyze the experimental results. Finally, in Section 5, we summarize our results and suggest some future directions.

2 Related Works

In machine learning, researchers have found that in many situations, training statistical learning models on multiple related data is better than training models on each data set individually. For example, in financial forecasting, models for predicting the values of many possibly related indicators simultaneously are often required. In marketing, modeling the preferences of many individuals simultaneously is common practice [41], [42]. When there are relations between the different tasks, it can be advantageous to learn all tasks at the same time instead of the traditional approach of learning each task independently of others, because certain common knowledge can be applied to benefit the learning of each task. Learning multiple related tasks simultaneously has been empirically as well as theoretically shown to often significantly improve the performance relative to learning each task independently [34], [36], [37], [38].

There are various ways of relating multiple tasks in multitask learning. The multiple functions learned in different tasks can be related to each other through the sharing parameters or prior distributions of the hyperparameters of the models [43], [44], [45]. The common knowledge among tasks is encoded into the shared parameters or priors. Thus, by discovering the shared parameters or priors, knowledge can be transferred across tasks. Tasks may also be related in that they all share a common underlying representation [37], [38], [33], [46], [47]. The intuitive idea behind this case is to learn a "good" feature representation for the target domain. In this case, the knowledge used to transfer across domains is encoded into the learned feature representation. With the new feature representation, the performance of the target task is expected to improve significantly.

In the past few years, several multitask learning methods have been proposed to solve biological problems. Bickel et al. [48] studied the problem of predicting the HIV therapy outcomes of different drug combinations based on observed genetic properties of the patients, where each task corresponds to a particular drug combination. They proposed to jointly train models for different drug combinations by pooling data together for all tasks and use resampling weights to adapt the data for each particular task. Bi et al. [49] formulated the detection of different types of clinically related abnormal structures in medical images as multitask learning. Their method captured the task dependence via hierarchical Bayesian modeling such that the parameters of different classifiers share a common prior distribution, which was shown to be effective in eliminating irrelevant features and identifying discriminative features. To the best of our knowledge, few research has been done in multitask learning for subcellular localization.

3 Multitask Learning for Subcellular Localization

3.1 Problem Definition and Notation

We consider multitask learning for protein subcellular localization by learning across different organisms. We have T different organisms, each of which is considered as a task. To use the multitask framework, we first assume that all the data come from the same space of features $X \times Y$, where $X \subset \mathbb{R}^m$ are the problem features and $Y \subset \mathbb{R}$ are the

[1] We will address the problem of multiplex proteins in the future work.

class labels. Thus, for each task t ($t \in 1, 2, \dots, T$), we have n_t data points:

$$\{(\mathbf{x}_1^t, y_1^t), (\mathbf{x}_2^t, y_2^t), \dots, (\mathbf{x}_{n_t}^t, y_{n_t}^t)\},$$

where $\mathbf{x}_i^t$ represents a protein in an organism t and y_i^t is its corresponding location within a cell. The goal is to learn T functions $f_1, f_2, \dots, f_T$ simultaneously, such that $f_t(\mathbf{x}_i^t) = y_i^t$ and the learned function f_t can generalize well for future data.

In the past, multitask learning methods are designed based on different notions of *relatedness* among the tasks. Different assumptions often lead to different ways in which to model the shared information among different tasks. In this work, we consider two specialization of the frameworks of multitask learning: parameter sharing and latent feature space sharing.

Before delving into the methodological detail, we first introduce some notation used in the paper. In the sequel, $\mathbf{A}$ is used to denote vectors and matrices. Given any positive number p, the p-norm of a vector $\mathbf{w} \in \mathbb{R}^m$ is defined as $\|\mathbf{w}\|_p = (\sum_{i=1}^m |w_i|^p)^{\frac{1}{p}}$. For a matrix $\mathbf{A}$, we denote the ith row, jth column, and ijth entry of $\mathbf{A}$ by $\mathbf{a}^i$, $\mathbf{a}_j$, and a_{ij}, respectively. For any positive number p and q, the (q, p)-norm of an $n \times m$ matrix $\mathbf{A}$ is $\|A\|_{q,p} = (\sum_{i=1}^n \|\mathbf{a}^i\|_q^p)^{\frac{1}{p}}$, which is equal to the p-norm of an m-dimensional vector containing the q-norms of the rows of $\mathbf{A}$. We define $\mathbf{O}^n$ to be the set of $n \times n$ orthogonal matrices.

3.2 Multitask Learning by Sharing Model Parameters

We first assume that for each organism t, the predictive function f_t is a linear function $f_t(\mathbf{x}_i^t) = \mathbf{w}^{t\top}\mathbf{x}_i^t$, which can estimate the location y_i^t of $\mathbf{x}_i^t$. We further assume that if the organisms are related to each other, then their predictive functions f_ts may share a common parameter. As a result, for each organism, the objective linear function can be written as follows:

$$f_t(\mathbf{x}_i^t) = (\mathbf{w_t} + \mathbf{w_c})^\top \mathbf{x}_i^t, \tag{1}$$

where $\mathbf{w_c}$ is a common parameter shared by different tasks, which captures the relatedness among the organisms. $\mathbf{w_t}$ is a specific parameter for each task, which represents organism-specific properties of proteins. By encoding (1) into a formulation of SVMs, we aim at solving the following optimization problem [50]. Let $J(\mathbf{w_c}, \mathbf{w_t}, \xi_i^t)$ be $\sum_{t=1}^T \sum_{i=1}^{n_t} \xi_{it} + \frac{\lambda_1}{T}\sum_{t=1}^T \|\mathbf{w_t}\|^2 + \lambda_2\|\mathbf{w_c}\|^2$,

$$\begin{aligned} \min_{\mathbf{w_c}, \mathbf{w_t}, \xi_i^t} \quad & \{J(\mathbf{w_c}, \mathbf{w_t}, \xi_i^t)\} \\ \text{s.t.} \quad & \forall i \in \{1, 2, \dots, n_t\} \& \; \forall t \in \{1, 2, \dots, T\}, \\ & y_i^t(\mathbf{w_c} + \mathbf{w_t})^\top \mathbf{x}_i^t \geq 1 - \xi_i^t, \\ & \xi_i^t \geq 0, \end{aligned} \tag{2}$$

where ξ_i^ts are slack variables measuring the error that each of the final models $\mathbf{w}^t$ makes on the data. λ_1 and λ_2 are positive regularization coefficients to control the effect of the common parameter $\mathbf{w}_c$ and organism-specific parameter $\mathbf{w}_t$, respectively. Intuitively, for a fixed value of λ_2, a large value of the ratio $\frac{\lambda_1}{\lambda_2}$ tends to make the models the same, while for a fixed value of λ_1, a small value of the ratio $\frac{\lambda_1}{\lambda_2}$ tends to make them different and unrelated.

In [50], it was proved that solving the optimization problem (2) is equivalent to solving the optimization problem as follows, which is a standard optimization problem of SVMs:

$$\begin{aligned} \min_{\mathbf{w}, \xi_i} \quad & \left\{J(\mathbf{w}, \xi_i) := \sum_{i=1}^N \xi_i + \|\mathbf{w}\|^2\right\} \\ \text{s.t.} \quad & \forall i \in \{1, 2, \dots, N\} \& \; \forall t \in \{1, 2, \dots, T\}, \\ & y_i \mathbf{w}^\top \Phi(\mathbf{x}_i^t, t) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \end{aligned} \tag{3}$$

where $N = \sum_t n_t$ and the objective function becomes $f_t(\mathbf{x}_i^t) = F(\mathbf{x}_i^t, t) = \mathbf{w}^\top \Phi(\mathbf{x}_i^t, t)$, where $\mathbf{w} = (\sqrt{\mu}\mathbf{w_c}, \mathbf{w_1}, \mathbf{w_2}, \dots, \mathbf{w_T})$ and $\mu = \frac{T\lambda_2}{\lambda_1}$. Φ can be treated as a feature map defined by

$$\Phi(\mathbf{x}_i^t, t) = \left(\frac{\mathbf{x}_i^t}{\sqrt{\mu}}, \underbrace{\mathbf{0}, \dots, \mathbf{0}}_{t-1}, \mathbf{x}_i^t, \underbrace{\mathbf{0}, \dots, \mathbf{0}}_{T-t}\right), \tag{4}$$

where we denote by $\mathbf{0}$ the zero vector in $\mathbb{R}^m$. Thus, for each pair $(\mathbf{x}_i^t, t)$, Φ maps it to a large feature vector $\Phi(\mathbf{x}_i^t, t) \in \mathbb{R}^{m(T+1)}$ with only two nonzero parts, where the first one is common to all organisms and the second one is at an organism-specific position.

By using the kernel trick [51], it is easy to generalize the linear objective function $F(\cdot, \cdot)$ to the nonlinear case. We assume that $\Phi : X \times \{1, 2, \dots, T\} \rightarrow \mathcal{H}$ be a nonlinear feature map, where $\mathcal{H}$ is a Hilbert space:

$$\Phi(\mathbf{x}_i^t, t) = \left(\frac{\phi(\mathbf{x}_i^t)}{\sqrt{\mu}}, \underbrace{\mathbf{0}, \dots, \mathbf{0}}_{t-1}, \phi(\mathbf{x}_i^t), \underbrace{\mathbf{0}, \dots, \mathbf{0}}_{T-t}\right), \tag{5}$$

where $\phi : X \rightarrow \mathcal{H}$ is also a nonlinear feature map. Then, the kernel associated to Φ is defined by

$$\begin{aligned} K((\mathbf{x}_i^{t_i}, t_i), (\mathbf{x}_j^{t_j}, t_j)) &= \langle \Phi(\mathbf{x}_i^{t_i}, t_i), \Phi(\mathbf{x}_j^{t_j}, t_j)\rangle \\ &= \begin{cases} \left(1 + \frac{\lambda_1}{T\lambda_2}\right) k(\mathbf{x}_i^{t_i}, \mathbf{x}_j^{t_j}) & t_i = t_j, \\ \frac{\lambda_1}{T\lambda_2} k(\mathbf{x}_i^{t_i}, \mathbf{x}_j^{t_j}) & \text{otherwise}, \end{cases} \end{aligned} \tag{6}$$

where $t_i, t_j \in \{1, 2, \dots, T\}$ and $k(\cdot, \cdot)$ is a kernel associated to ϕ. Based on the *representor theorem* [35], we can learn the coefficients β_j for the function

$$F(x_i^t, t) = \sum_{j=1}^N \beta_j K((\mathbf{x}_i^{t_k}, t_k), (\mathbf{x}_j^{t_j}, t_j)),$$

by solving the standard dual problem with kernel K. For more general cases, we can rewrite the kernel K by a product of two kernels, as follows:

$$K((\mathbf{x}_i^{t_i}, t_i), (\mathbf{x}_j^{t_j}, t_j)) = K_{task}(t_i, t_j) K_{example}(\mathbf{x}_i, \mathbf{x}_j), \tag{7}$$

where K_{task} is a kernel defined on the tasks and $K_{example}$ is a kernel defined on the examples. In our case, K_{task} is the *organism kernel* that quantifies how information is shared between organisms, and $K_{example}$ is the *protein kernel* that quantifies similarity between the proteins. In (6),

$$K_{task}(t_i, t_j) = \begin{cases} 1 + \frac{\lambda_1}{T\lambda_2} & t_i = t_j, \\ \frac{\lambda_1}{T\lambda_2} & \text{otherwise.} \end{cases}$$

In the sequel, we call the kernel as defined above the *regularization kernel* $K_{regularization}$.

In [39], Jacob and Vert designed K_{task} for Epitope prediction. This corresponds to the approach of sharing parameters on K_{task}. In our work, the *organism kernels* K_{tasks} used in the experiments are summarized as follows:

$$\begin{aligned} K_{regularization}(t_i, t_j) &= \begin{cases} 1 + \frac{\lambda_1}{T\lambda_2} & t_i = t_j, \\ \frac{\lambda_1}{T\lambda_2} & \text{otherwise,} \end{cases} \\ K_{uniform}(t_i, t_j) &= 1 \quad \forall t_i, t_j \in \{1, 2, \dots, T\}, \\ K_{multitask}(t_i, t_j) &= \begin{cases} 2, & t_i = t_j, \\ 1, & \text{otherwise,} \end{cases} \\ K_{supertype}(t_i, t_j) &= \begin{cases} K_{multitask}(t_i, t_j) + 1, \\ \text{if } t_i \text{ and } t_j \text{ are in the same} \\ \text{supertype,} \\ K_{multitask}(t_i, t_j), \\ \text{otherwise.} \end{cases} \end{aligned}$$

For *protein kernels* $K_{example}$, we can use a linear kernel, a polynomial kernel, and an RBF kernel, which are widely used in many real-world applications. In our experimental setting, we conduct a series of experiments on different choices of the *organism kernel* and the *protein kernel*, as well as their different combinations.

3.3 Multitask Learning by Sharing Latent Features

The multitask learning method in the above section is based on sharing model parameters. In this section, we consider an alternative multitask learning framework based on sharing *latent features* across the tasks.

In [40], Argyriou et al. proposed a feature learning framework for multitask learning. In particular, this framework attempts to learn a low-dimensional feature representation shared by different tasks by minimizing the errors within each task while jointly regularizing the parameters of different models.

For simplicity, we first study the case of binary classification tasks for which the corresponding predictive functions are linear. Our goal is to learn T objective functions with the following form simultaneously:

$$f_t(\mathbf{x}_i^t) = \sum_{j=1}^{m} a_{jt} h_j(\mathbf{x}_i^t), \quad t \in \{1, 2, \dots, T\},$$

where $h_j : \mathbb{R}^m \rightarrow \mathbb{R}$ are feature maps that connect the original data to common features and $a_{jt} \in \mathbb{R}$ are model parameters. For simplicity, we focus on linear feature maps; that is, $h_j(\mathbf{x}_i^t) = \langle u_j, \mathbf{x}_i^t \rangle$. Thus, the objective functions can be rewritten as follows:

$$f_t(\mathbf{x}_i^t) = \sum_{j=1}^{m} a_{jt}(\mathbf{U}^\top \mathbf{x}_i^t), \quad t \in \{1, 2, \dots, T\},$$

where each column of $\mathbf{U}$ corresponds to a linear feature map.

To make the connection among tasks in the training process, Argyriou et al. [40] proposed to use a regularization term to model the common structure underlying the tasks. Thus, the final optimization problem for multitask learning can be rewritten as follows:

$$\begin{aligned} \min_{\mathbf{U},\mathbf{A}} \quad & \sum_{t=1}^{T} \sum_{i=1}^{n_t} \frac{1}{n_t} L(y_i^t, \langle \mathbf{a_t}, \mathbf{U}^\top \mathbf{x_i^t} \rangle) + \gamma \|\mathbf{A}\|_{\mathbf{2,1}}^{\mathbf{2}}, \\ \text{s.t.} \quad & \mathbf{U} \in \mathbf{O}^m, \quad \mathbf{A} \in \mathbb{R}^{m \times T}, \end{aligned} \tag{8}$$

where $L(\cdot, \cdot)$ is a loss function. The first term in (8) is the average of the empirical error across the tasks. The second term is a regularization term that penalizes the (2,1)-norm of the matrix $\mathbf{A}$, which aims to force the common features across the tasks to be sparse. More specifically, $\|\mathbf{A}\|_{2,1}^2$ first computes $\|\mathbf{a}^i\|_2$, the 2-norms of the rows of matrix $\mathbf{A}$, and then, computes the 1-norm of the vector $(\|\mathbf{a}^1\|_2, \dots, \|\mathbf{a}^m\|_2)$. This favors solutions in which entire rows of $\mathbf{A}$ are 0, which encourages selecting the features that are generally useful to all tasks. This formulation introduces dependency between the parameters of different tasks via the (2,1)-norm-based regularization, while the shared feature projection matrix $\mathbf{U}$ is learned based on the training data from all tasks. These are the key mechanisms that enabled different tasks to mutually enhance each other. If $\mathbf{U} = \mathbf{I}$, where $\mathbf{I}$ is an identity matrix, then the feature learning problem for multitask learning is reduced to a feature selection problem for multitask learning. The positive coefficient γ is to balance the importance between the error and the penalty.

In this paper, we apply this framework to solve the problem of protein subcellular localization across organisms. We encode logistic regression into multitask learning framework and extend it to solve multiclass problems (that is, prediction problem where the number of class labels is more than two) for protein subcellular localization. For each organism, the predictive function of logistic regression can be written as a parametric form of the conditional probability of y_i^t given $\mathbf{x}_i^t$:

$$f_t(\mathbf{x}_i^t) = P(y_i^t = 1 | \mathbf{a}_t, \mathbf{x}_i^t) = \frac{1}{1 + \exp(\mathbf{a}_t{}^\top \mathbf{x}_i^t)}, \tag{9}$$

where $\mathbf{a}_t$ is the model parameter vector. Typically, the parameter vector $\mathbf{a}_t$ can be estimated by using the maximum likelihood technique, which leads to solving the following optimization problem:

$$\begin{aligned} \min_{\mathbf{a}^t} \quad & \left\{ L(y_i^t, f_t(\mathbf{x_i^t})) = \sum_{\mathbf{i=1}}^{\mathbf{n_t}} \log(\mathbf{1} + \exp(-\mathbf{y_i^t a_t}^\top \mathbf{x_i^t})) \right\}, \\ \text{s.t.} \quad & \mathbf{a}_t \in \mathbb{R}^m. \end{aligned} \tag{10}$$

By substituting (10) into (8) appropriately, we can induce the optimization problem as follows:

$$\begin{aligned} \min_{\mathbf{U},\mathbf{A}} \quad & \sum_{t=1}^{T} \sum_{i=1}^{n_t} \frac{1}{n_t} \log(1 + \exp(-y_i^t \mathbf{a}_t{}^\top \mathbf{U}^\top \mathbf{x}_i^t)) + \gamma \|\mathbf{A}\|_{2,1}^2, \\ \text{s.t.} \quad & \mathbf{U} \in \mathbf{O}^m, \quad \mathbf{A} \in \mathbb{R}^{m \times T}. \end{aligned} \tag{11}$$

For solving the optimization problem proposed in (11), we extend the efficient algorithm proposed in [40] to our

TABLE 1
Statistics of Data Sets

organisms	No. of instances	No. of locations	organisms	No. of instances	No. of locations
archaea	635	4	bacteria	5264	5
bovine	277	6	dog	47	6
fish	114	6	fly	591	6
frog	191	6	fungi	2828	6
gneg	1296	8	gpos	452	5
human0	2396	14	human	3455	6
mouse	1808	6	pig	112	6
plant0	671	11	plant	912	7
rabbit	95	6	rat	716	6
virus0	112	4	virus	568	3

setting, which iteratively updates matrices **U** and **A** until the corresponding convergence condition holds.

We present our comparison results of the above two methods in the next section.

4 Experimental Results and Discussion

4.1 Experimental Hypotheses and Material

Above we have discussed two approaches in which we can apply multitask learning to the subcellular localization problem. In this section, we evaluate two hypotheses related to this problem:

- First, our intuition tells us that related species may help each other in making the classification better. But can we verify such results in real-data experiments?
- Second, above we have considered two potential ways to apply multitask learning to the subcellular localization problem. Which method is more suitable to the problem at hand? Again, we will answer this question through experiments.

We used 20 protein data sets with determined subcellular localization, obtained from 1) Cell-Ploc [1], including human, plant, gram-positive, gram-negative, and virus cells that are denoted by human0, plant0, gpos, gneg, and virus0 in the following experiment and analysis section, respectively; and 2) DBSubLoc [52], including archaea, bacteria, bovine, dog, fish, fly, frog, human, mouse, pig, rabbit, rat, fungi, plant, and virus, denoted by archaea, bacteria, bovine, dog, fish, fly, frog, human, mouse, pig, rabbit, rat, fungi, plant, and virus, respectively. Cutoff threshold of 25 percent is used for data sets extracted from Cell-Ploc to exclude those proteins that have equal to or greater than 25 percent sequence identity to others. We then set 60 percent threshold to exclude redundant proteins for the data sets extracted from DBSubLoc. The statistics and description list are given in Table 1.

When preprocessing these data sets, we exclude the human proteins with multiple locations extracted from Cell-Ploc [1]. The 2-gram protein encoding method is used to generate features of amino acid compositions, which is widely used in many existing protein subcellular localization systems [53]. We randomly sample 60 percent of each individual data set for training and use the rest 40 percent for testing. Among the independent data set test, subsampling (e.g., K-fold cross validation) test, and jackknife test, which are often used for examining the accuracy of a statistical prediction method [54], the jackknife test was deemed the most objective that can always yield a unique result for a given benchmark data set, as elucidated in [1] and demonstrated by (50) of [55]. Therefore, the jackknife test has been increasingly and widely adopted by investigators to test the power of various prediction methods (see, e.g., [19], [20], [24], [26], [56], [57], [58], [59], [60], [61], [62], [63], [64], [65], [66], [67], [68], [69]). To reduce the computational time, we repeat the five trials and report the average results in this study.

4.2 Baseline Multitask Learning Methods

In our experimental setting, we adopt standard SVMs with the linear kernel, polynomial kernel, and RBF kernel as baseline methods, which are denoted by *baseline1*, *baseline2*, and *baseline3*, respectively. Although there are many existing state-of-the-art methods and feature extraction approaches for subcellular localization prediction, our focus in this paper is to introduce a useful and strong learning framework "multitask" to address subcellular localization problem and illustrate the benefit of multitask learning comparing with single-task learning. Therefore, we choose simple amino acid compositions as input and standard single SVMs as baselines, which were used by [70], comparing with SVMs and other weak learners under multitask learning framework here. Actually, in the further study, we can extend the existing prediction methods under multitask learning framework in order to improve their prediction performance. We further denote the multitask learning method implemented based on the framework of "multitask learning by sharing model parameters" described in *method1*. Different combinations of organism kernels and protein kernels used in our experiments are summarized as follows:

$$\begin{aligned}&K_{regularization} \times K_{linear}, K_{regularization} \times K_{poly},\\&\quad K_{regularization} \times K_{RBF}, K_{uniform} \times K_{linear}, K_{uniform} \times K_{poly},\\&\quad K_{uniform} \times K_{RBF}, K_{multitask} \times K_{linear}, K_{multitask} \times K_{poly},\\&\quad K_{multitask} \times K_{RBF}, K_{supertype}, \times K_{linear}, K_{supertype} \times K_{poly},\end{aligned}$$

and $K_{supertype} \times K_{RBF}$. A standard SVM classifier is used for final prediction with these kernels. Finally, we denote by *method2* the multitask learning method implemented based on the framework of "multitask learning by sharing latent features." In *method2*, we have two settings, if **U** in (11) is not learned and $\mathbf{U} = \mathbf{I}$, where **I** is an identity matrix, then it is called "feature select;" otherwise, it is referred to as "feature learn."

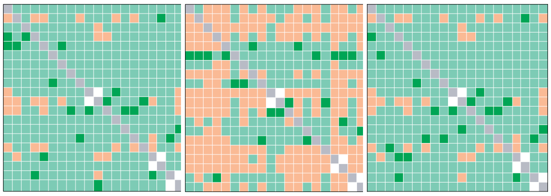


Fig. 1. K_{li} represents K_{linear}, summary of determined performances for *method1* using (a) $K_{multitask} \times K_{li}$, (b) $K_{uniform} \times K_{li}$, and (c) $K_{supertype} \times K_{li}$.

4.3 Performance Measure

We use the classification accuracy of the protein subcellular localization to evaluate the performance of different approaches. In our work, the metric *Accuracy* is defined as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}, \qquad (12)$$

where TP and TN denote the number of correctly classified positive and negative examples, and FP and FN denote the number of incorrectly classified positive and negative examples, respectively. Here, we use one versus the rest to define positive and negative examples.

4.4 Comparison with Single-Task Learning by Dual-Task Combinations

To answer the question "Can multitask learning generate more accurate classifiers than single-task learning?," we compare the accuracies on the test data among our proposed multitask learning methods and baselines. We conduct comparisons using the dual-task combinations by using arbitrary pairs of tasks. The results are summarized in Figs. 1, 2, 3, and 4. Fig. 1 illustrates the accuracies of multitask *method1* with the kernels $K_{multitask} \times K_{linear}$, $K_{uniform} \times K_{linear}$, and $K_{supertype} \times K_{linear}$, respectively, as well as the accuracies of standard SVM with linear kernel on the each task test data. Fig. 2 illustrates the accuracies of multitask *method1* with $K_{multitask} \times K_{poly}$, $K_{uniform} \times K_{poly}$, and $K_{supertype} \times K_{poly}$ kernel, respectively, as well as the accuracies of standard SVM with polynomial kernel on the each task test data. Fig. 3 illustrates the accuracies of multitask *method1* with $K_{multitask} \times K_{RBF}$, $K_{uniform} \times K_{RBF}$, and $K_{supertype} \times K_{RBF}$ kernel, respectively, as well as accuracies of standard SVM with RBF kernel on the each task test data. Fig. 4 shows the performance of "feature learning" in *method2* (Fig. 4a) and 'feature selection' in *method2* (Fig. 4b), respectively. The diagonal cells in Fig. 4 are obtained by *baseline1* (linear SVM). For tuning the parameters, we choose the parameters that give the best results. Generally, for all RBF kernels, we choose $\gamma = 0.0003$; for all polynomial kernels, we choose degree $= 3$; specifically, *method1* uses $\mu = \frac{T\lambda_2}{\lambda_1} = 1$ and *method2* (both "feature learning" and "feature selection") uses $\gamma = 2$. Due to the parameters determined above, $K_{regarlization}$ equals to $K_{multitask}$ in *method1*, we, therefore, only report $K_{multitask}$ here instead of $K_{regularization}$. In *method1*, we need to define "supertype" among organisms for using the kernel $K_{supertype}$. From conventional biological point of view, archaea and bacterial are categorized as two domains of prokaryote. Thus, organisms such as archaea, bacteria, gneg, and gpos can be considered belonging to the same supertype; organisms such as bovine, dog, fish, fly, frog, human0/human, mouse, pig, rabbit, rat, fungi, and plant0/plant can be categorized into the same supertype of eukaryote. Furthermore, to extend *method2* to deal with multiclass classification problems, we transform method 2 to multiple binary classification problems. The detailed results are given in the Appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2010.22.

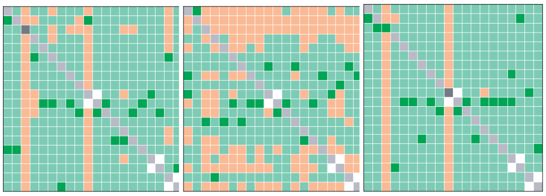


Fig. 3. Summary of determined performances for *method1* using (a) $K_{multitask} \times K_{RBF}$, (b) $K_{uniform} \times K_{RBF}$, and (c) $K_{supertype} \times K_{RBF}$.

We now explain Figs. 1, 2, 3, and 4 in detail. The columns from left to right and rows from up to down represent the organisms: archaea, bacteria, gneg, gpos, bovine, dog, fish, fly, frog, human0, human, mouse, pig, rabbit, rat, fungi, plant0, plant, virus0, and virus in order. Each cell C_{ij} in the figure is a average result over five random trails. More specifically, for C_{ij}, we jointly train models on the organism i and the organism j and use the trained model $f_j(\cdot)$ on the test data from the organism j. For diagonal cells C_{ii} (in gray), we train models on the training data of organism i only and evaluate on the test data of organism i as well. Thus, they correspond to the traditional supervised single-task learning, which we use as the baselines (*basline1* in

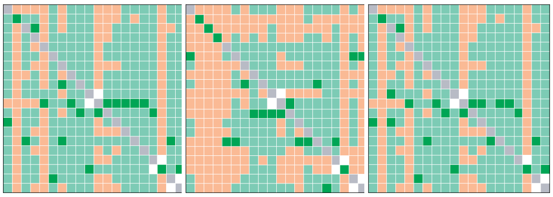


Fig. 2. Summary of determined performances for *method1* using (a) $K_{multitask} \times K_{poly}$, (b) $K_{uniform} \times K_{poly}$, and (c) $K_{supertype} \times K_{poly}$.

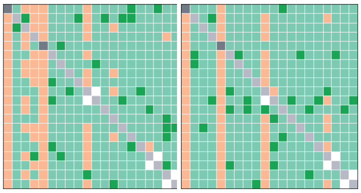


Fig. 4. Summary of determined performances for different settings of method2. (a) Feature learn. (b) Feature select.

Figs. 1 and 4, *baseline2* in Fig. 2, and *baseline3* in Fig. 3). The cells marked in red indicate that applying multitask learning methods gives worse performance than the baselines, whereas those in *light green* indicate that applying multitask learning methods gets better performance than the baselines. Furthermore, the cells in dark green or dark gray represent the best performance when evaluating on the test data from each column organism. Finally, the cells in white means that the performance result is missing, because some organisms that we used are overlapped, as in the case of human0 versus human, plant0 versus plant, and virus0 versus virus. Thus, we cannot conduct multitask learning experiments on these pairs.

From the above results, we can make the following observations:

1. Generally, *method1* using $K_{uniform} \times K_{linear}$, $K_{uniform} \times K_{poly}$, and $K_{uniform} \times K_{RBF}$ performs the worst. This means that using these kernels, dual-task combinations give little help for improving the performance. In many cases, using these kernels may even cause the performance to be worse. This may be because uniform kernel $K_{uniform}$ just pools data from different organisms simply together without considering the relatedness of different organisms. However, *method1* with other kernels and *method2* including both "feature learning" and "feature selection" indeed improve the performance as compared to single-task learning.
2. *method1* with RBF kernel achieves the best improvement. Nevertheless, *method1* with either $K_{multitask} \times K_{poly}$ or $K_{supertype} \times K_{poly}$ does not give promising results even though they still give a slight improvement.
3. We also note that *method1* using $K_{multitask} \times K_{RBF}$ and $K_{supertype} \times K_{RBF}$ works well for all dual-task combinations except for the one of gneg and human0.
4. By observing the tables shown in the Appendix section, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2010.22, we can find that the most significant improvement of using multitask learning strategy is about 25 percent. The performance of organisms plant, virus, and those belong to animals can be improved by around 10 percent by using multitask learning methods.
5. Interestingly, the columns from left to right and rows from up to down in the table represent organisms: archaea, bacteria, gneg, gpos, bovine, dog, fish, fly, frog, human0, human, mouse, pig, rabbit, rat, fungi, plant0, plant, virus0, and virus in order, which means that we arranged the tasks for learning on organisms in supertype order, for example, those organisms belonging to animal are put together. Moreover, better results are often obtained when approaching diagonals, while worse cases are often located in the cells far from diagonals. Therefore, the natural explanation is that the results in cells near the diagonals are obtained by training two relatively similar tasks like dog and fly, bacteria and archaea, and so on. As mentioned above, the organisms are listed based on their similarity. In contrast, the accuracy results in cells far from diagonals are obtained by training tasks in relatively low similarity, such as archaea and dog. Thus, we may conclude that multitask learning techniques generally help improve the prediction performance for protein subcellar localization in comparison with supervised single-task learning techniques. Furthermore, the relatedness of tasks may affect the final performance under the multitask learning framework.

4.5 Effect of Task Similarity in Terms of Prediction Accuracy

To answer the question "how do different task combinations affect the performance of multitask learning?" and the question "is there any correlation between the task relatedness and the final performance?," we conduct a series of experiments on eight different organism combinations for study. These organism combinations include:

1. bovine + dog + fish + fly + frog + human + mouse + pig + rabbit + rat;
2. bovine + dog + fish + fly + frog + human + mouse + pig + rabbit + rat + bacteria + archaea;
3. bovine + dog + fish + fly + frog + human + mouse + pig + rabbit + rat + virus;
4. bovine + dog + fish + fly + frog + human + mouse + pig + rabbit + rat + fungi;
5. bovine + dog + fish + fly + frog + human + mouse + pig + rabbit + rat + plant;
6. bacteria + archaea + virus;
7. bacteria + archaea + fungi; and
8. bacteria + archaea + plant which are abbreviated by *comb1*, *comb2*, *comb3*, *comb4*, *comb5*, *comb6*, *comb7*, and *comb8*, respectively.

comb1 is composed of animal organisms only; *comb2* consists of animal organisms belonging to eukaryote and bacteria and archaea; *comb3* involves animal and virus organisms; *comb4* includes animal and fungi organisms; *comb5* includes animal and plant organisms; *comb6*, *comb7*, and *comb8* contain organisms belong to bacteria and archaea together with virus, fungi, and plant, respectively, among which fungi and plant are also in eukaryote category but different from animal organisms. In this experimental setting, *method1* with multitask kernel as the *organism kernel* (denoted by *protein kernel* for convenience in the following tables), and *method2* and their corresponding kernel are used for comparison (*basline1* used to compare with *method2*). Similar to experimental setting described in previous section, all the results are obtained by averaging the results of five independent random trails. The detailed results are summarized in Tables 2, 3, 4, 5, 6, 7, 8, and 9. Note that the italic number in red in the tables indicates that the performance is worse than that of the corresponding baseline.

From these results, we can make the following observations:

1. It is clear that *comb1* achieves the best results among which all organisms evaluated by *method1* using RBF, polynomial, and linear kernels in multitask kernels have higher accuracies than the

TABLE 2
Results of *method1* Involving Multitask Kernel, *method2*, and the Baselines—the Training Set Is the Task *comb1* and the Test Set Is the Individual Task Listed in First Column

comb1	baseline3	K_{RBF}	baseline2	K_{poly}	baseline1	K_{linear}	feature learn	feature select
bovine	53.15	**75.68**	57.66	**63.24**	63.06	**63.24**	*61.62*	*58.56*
dog	63.16	**91.58**	52.63	**87.37**	57.89	**77.89**	72.63	69.47
fish	58.70	**78.26**	56.52	**66.96**	65.22	**68.26**	*56.09*	*56.52*
fly	60.17	**69.32**	48.31	**56.02**	54.66	56.53	63.48	**65.42**
frog	72.73	**84.68**	58.44	**72.99**	66.23	69.87	**74.55**	72.73
human	57.02	**62.66**	48.26	**49.46**	44.14	45.22	53.30	**54.82**
mouse	61.96	**71.04**	54.63	**60.33**	54.91	58.12	63.21	**63.76**
pig	40.00	**71.56**	48.89	**59.11**	53.33	**60.00**	55.56	56.89
rabbit	63.16	**82.11**	50.68	**77.37**	57.89	**71.05**	66.32	63.16
rat	65.38	**76.22**	44.74	**68.32**	63.29	68.60	**68.88**	67.41

TABLE 3
Results of *Method1* Involving Multitask Kernel, *method2*, and the Baselines—the Training Set Is the Task *comb2* and the Test Set Is the Individual Task Listed in First Column

comb2	baseline3	K_{RBF}	baseline2	K_{poly}	baseline1	K_{linear}	feature learn	feature select
bovine	53.15	**76.04**	57.66	**64.14**	63.06	**64.14**	*61.80*	*59.10*
dog	63.16	**89.47**	52.63	**88.42**	57.90	**76.84**	71.68	57.89
fish	58.70	**78.70**	56.52	**65.22**	65.22	**66.09**	*57.83*	*56.09*
fly	60.17	**69.15**	48.31	**56.02**	54.66	56.27	64.24	**65.09**
frog	72.73	**84.42**	58.44	**74.03**	66.23	70.39	**74.55**	72.99
human	57.02	**62.81**	48.26	**49.25**	44.14	44.78	53.39	**54.86**
mouse	61.96	**71.67**	54.63	**60.08**	54.91	57.87	63.29	**63.82**
pig	40.00	**70.67**	48.89	**59.11**	53.33	**57.33**	*48.00*	56.44
rabbit	63.16	**81.05**	50.68	**78.42**	57.89	**72.11**	65.79	63.16
rat	65.38	**76.99**	44.74	**66.99**	63.29	68.04	**69.65**	67.34
bacteria	86.94	**89.12**	85.28	*84.46*	83.76	*83.49*	**85.66**	84.63
archaea	78.74	**86.54**	79.13	*77.95*	79.92	*75.59*	*77.09*	*75.83*

TABLE 4
Results of *method1* Involving Multitask Kernel, *method2*, and the Baselines—the Training Set Is the Task *comb3* and the Test Set Is the Individual Task Listed in First Column

comb3	baseline3	K_{RBF}	baseline2	K_{poly}	baseline1	K_{linear}	feature learn	feature select
bovine	53.15	**75.14**	57.66	**62.88**	63.06	**63.60**	*61.44*	*58.38*
dog	63.16	**90.53**	52.63	**86.32**	57.89	**75.79**	68.42	69.47
fish	58.70	**80.00**	56.52	**66.09**	65.22	**67.83**	*56.09*	*56.52*
fly	60.17	**69.66**	48.31	**56.27**	54.66	56.69	63.73	**65.34**
frog	72.73	**84.42**	58.44	**71.95**	66.23	70.39	**74.55**	72.73
human	57.02	**62.53**	48.26	**49.42**	44.14	45.18	53.37	**54.80**
mouse	61.96	**71.45**	54.63	**60.08**	54.91	57.95	63.07	**63.79**
pig	40.00	**72.00**	48.89	**59.11**	53.33	**60.44**	56.00	56.00
rabbit	63.16	**82.11**	50.68	**76.32**	57.89	**70.00**	65.26	63.16
rat	65.38	**76.57**	44.74	**68.18**	63.29	68.46	**69.09**	67.34
virus	81.94	**85.99**	80.18	**84.23**	81.94	*78.33*	**85.11**	84.58

TABLE 5
Results of *method1* Involving Multitask kernel, *method2*, and the Baselines—the Training Set Is the Task *comb4* and the Test Set Is the Individual Task Listed in First Column

comb4	baseline3	K_{RBF}	baseline2	K_{poly}	baseline1	K_{linear}	feature learn	feature select
bovine	53.15	**74.59**	57.66	**62.70**	63.06	*61.62*	*62.16*	*58.38*
dog	63.16	**89.47**	52.63	**86.32**	57.89	**82.11**	72.63	69.47
fish	58.70	**78.26**	56.52	**63.91**	65.22	**70.00**	*56.09*	*56.09*
fly	60.17	**68.64**	48.31	**57.12**	54.66	55.76	63.56	**65.25**
frog	72.73	**84.68**	58.44	**74.03**	66.23	72.73	**74.81**	72.73
human	57.02	**62.68**	48.26	**49.10**	44.14	45.43	53.31	**54.73**
mouse	61.96	**71.73**	54.63	**61.36**	54.91	57.79	63.29	**63.96**
pig	40.00	**82.11**	48.89	**61.33**	53.33	**61.33**	56.44	56.89
rabbit	76.85	**82.11**	50.68	**76.84**	57.89	**73.16**	66.84	63.16
rat	65.38	**76.85**	44.74	**70.14**	63.29	68.46	**69.23**	67.34
fungi	62.42	**65.22**	51.02	**53.07**	46.60	48.58	60.74	**62.67**

TABLE 6
Results of *method1* Involving Multitask Kernel, *method2*, and the Baselines—the Training Set Is the Task *comb5* and the Test Set Is the Individual Task Listed in First Column

comb5	baseline3	K_{RBF}	baseline2	K_{poly}	baseline1	K_{linear}	feature learn	feature select
bovine	53.15	**75.68**	57.66	**62.52**	63.06	**64.50**	*62.34*	*58.38*
dog	63.16	**90.53**	52.63	**85.26**	57.89	**80.00**	71.58	69.47
fish	58.70	**79.57**	56.52	**66.96**	65.22	**67.39**	*56.09*	*56.96*
fly	60.17	**68.90**	48.31	**56.36**	54.66	56.27	63.81	**65.42**
frog	72.73	**84.68**	58.44	**73.25**	66.23	69.09	**74.81**	72.73
human	57.02	**62.60**	48.26	**49.48**	44.14	45.33	53.58	**54.82**
mouse	61.96	**71.26**	54.63	**60.66**	54.91	58.45	63.13	**63.82**
pig	40.00	**72.44**	48.89	**60.00**	53.33	**60.44**	56.00	56.44
rabbit	63.16	**83.16**	50.68	**77.37**	57.89	**71.58**	66.84	63.16
rat	65.38	**76.64**	44.74	**67.48**	63.29	67.90	69.30	67.48
plant	58.08	**62.03**	50.68	*45.37*	49.86	49.86	**55.51**	55.45

TABLE 7
Results of *method1* Involving Multitask Kernel, *method2*, and the Baselines—the Training Set Is the Task *comb6* and the Test Set Is the Individual Task Listed in First Column

comb6	baseline3	K_{RBF}	baseline2	K_{poly}	baseline1	K_{linear}	feature learn	feature select
bacteria	86.94	**89.08**	85.28	*84.62*	83.76	84.73	84.71	**84.81**
archaea	78.74	**85.59**	79.13	**80.39**	79.92	*79.53*	*75.83*	*76.06*
virus	81.94	**85.46**	80.18	**82.56**	81.94	**82.29**	*77.00*	*75.00*

TABLE 8
Results of *method1* Involving Multitask Kernel, *method2*, and the Baselines—the Training Set Is the Task *comb7* and the Test Set Is the Individual Task Listed in First Column

comb7	baseline3	K_{RBF}	baseline2	K_{poly}	baseline1	K_{linear}	feature learn	feature select
bacteria	86.94	**89.31**	85.28	*84.91*	83.76	83.91	84.67	**84.76**
archaea	78.74	**86.30**	79.13	79.13	79.92	*77.64*	*76.06*	*75.98*
fungi	62.42	**63.75**	51.02	**51.44**	46.60	**48.17**	*35.15*	*34.66*

TABLE 9
Results of *method1* Involving Multitask Kernel, *method2*, and the Baselines—the Training Set Is the Task *comb8* and the Test Set Is the Individual Task Listed in First Column

comb8	baseline3	K_{RBF}	baseline2	K_{poly}	baseline1	K_{linear}	feature learn	feature select
bacteria	86.94	**89.15**	85.28	**87.65**	83.76	**87.65**	84.86	84.74
archaea	78.74	**85.59**	79.13	*77.95*	79.92	*77.95*	*77.80*	*76.22*
plant	58.08	**60.99**	50.68	*18.52*	49.86	*18.52*	*41.10*	*41.81*

corresponding baselines. Moreover, *method1* using $K_{multitaks} \times K_{RBF}$ gives the best performance.

2. Overall, the generalization ability of *method2* as well as *method1* both with $K_{multitask} \times K_{linear}$ and $K_{multitask} \times K_{poly}$ kernel is weaker than that of *method1* with the kernel $K_{multitask} \times K_{RBF}$.
3. The most essential and interesting observation that we discovered is that *comb1* is only composed of tasks belonging to animal organisms, which are strongly related to each other, which reports a prediction accuracy improvement. However, when *comb1* is integrated with bacteria and archaea to become *comb2*, or when it integrates with virus to become *comb3*, the performance may get worse. Several worse results on *comb4* and *comb5* are caused when introducing fungi and plant, respectively, both of which are in eukaryote category as animal but are different from animal. Among *comb6*, *comb7*, and *comb8*, the side effect happens frequently, which can be observed in the case of archaea and virus in *comb6*, archaea and fungi in *comb7*, as well as archaea and plant in *comb8*. Thus, it can be concluded again that the relatedness of tasks may indeed affect the performance of multitask learning methods: the closer the tasks are related, the better the performance of the prediction. In contrast, jointly training distantly related tasks may not help improve the performance.

4.6 Discussion

As our experimental results have shown, the accuracy improvement can reach 25 percent in the best case. This illustrates that related tasks can help improve the performance of learning and prediction, which confirms our intuition. Of particular importance is the relatedness of tasks, which we have shown to indeed affect the performance of multitask learning methods. From a biological point of view, we showed that combining the learning problems of different related organisms can be beneficial, whereas learning for unrelated organisms together cannot lead to significant improvement. In many cases, unrelated tasks may even cause worse results.

Methodologically, we compared two methods: sharing parameters and sharing latent features. For protein subcellular localization, methods with multitask and supertype kernels under the framework of “multitask learning by sharing model parameters” performed better than methods under “multitask learning by sharing latent features.” “Multitask learning by sharing latent features” aims at learning a low-dimensional latent feature representation, shared by different tasks. However, since we use 2-gram to extract our features, features of each task are very sparse. On the one hand, it is difficult to learn feature representations that are shared across tasks based on sparse features for each task. On the other hand, multitask kernel and supertype kernel seem quite natural to apply to our problem, which places lower weight on different organisms, especially for organisms from different supertypes. This might explain why “multitask learning by sharing latent features” performs worse than multitask and supertype kernels under the framework of “multitask learning by sharing model parameter.”

5 Conclusions

In this paper, we have tackled the problem of data sparsity in subcellular localization by multitask learning so that models of multiple related organisms are trained together. We have shown empirically that multitask learning can indeed improve the performance. Furthermore, two multitask learning frameworks are compared on the problem of protein subcellular localization. Two kinds of experiments are conducted based on dual-task combinations and task combinations of similarity and dissimilarity. The parameter sharing approach is found to perform better.

In conclusion, we have strong belief that multitask learning techniques in machine learning can be used as a powerful and useful tool to alleviate the data scarceness problem, and improve the performance dramatically in protein subcellular localization. We also believe that this method can be extended to other biological problems. In the future, we wish to study how to introduce the unlabeled data into multitask learning for protein subcellular localization, which is considered the properties of biological data, in particular, protein data comprehensively and deeply. Furthermore, how to select similar organisms automatically is crucial and interesting.

Acknowledgments

The authors thank the support of HKUST projects HKUST4/CRF-SF/08, RPC06/07.EG09, and Joint Hong-Kong CERG/China-NSF Grant N_HKUST624/09, and also thank the Fok Ying Tung Foundation and the Science and Technology Bureau of Nansha, Guangzhou, Guangdong, China.

References

[1] K.C. Chou and H.B. Shen, “Cell-Ploc: A Package of Web Servers for Predicting Subcellular Localization of Proteins in Various Organisms,” *Nature Protocol*, vol. 3, pp. 153-162, 2008.

[2] E.C. Su, H.S. Chiu, A. Lo, J.K. Hwang, T.Y. Sung, and W.L. Hsu, “Protein Subcellular Localization Prediction Based on Compartment-Specific Feature and Structure Conservation,” *BMC Bioinformatics*, vol. 8, article no. 330, 2007.

[3] M. Claros, S. Brunak, and G. Heijne, “Prediction of N-Terminal Protein Sorting Signals,” *Current Opinion in Structural Biology*, vol. 7, pp. 394-398, 1997.

[4] H. Nakashima and K. Nishikawa, “Discrimination of Intracellular and Extracellular Proteins Using Amino Acid Composition and Residue-Pair Frequencies,” *J. Molecular Biology*, vol. 238, no. 1, pp. 54-61, 1994.

[5] K.C. Chou and D.W. Elrod, “Protein Subcellular Location Prediction,” *Protein Eng.*, vol. 12, no. 2, pp. 107-118, 1999.

[6] K.C. Chou and Y.D. Cai, “Using Functional Domain Composition and Support Vector Machines for Prediction of Protein Subcellular Location,” *J. Biology Chemistry*, vol. 277, no. 48, pp. 45765-45769, 2002.

[7] K.C. Chou, “Prediction of Protein Cellular Attributes Using Pseudo-Amino Acid Composition,” *Proteins*, vol. 43, no. 3, pp. 246-255, 2001.

[8] G.P. Zhou and K. Doctor, “Subcellular Location Prediction of Apoptosis Proteins,” *Proteins*, vol. 50, no. 1, pp. 44-48, 2003.

[9] K.C. Chou and H.B. Shen, “Predicting Protein Subcellular Location by Fusing Multiple Classifiers,” *J. Cellular Biochemistry*, vol. 99, no. 2, pp. 517-527, 2006.

[10] K.C. Chou and H.B. Shen, “Predicting Eukaryotic Protein Subcellular Location by Fusing Optimized Evidence-Theoretic K-Nearest Neighbor Classifiers,” *J. Proteome Research*, vol. 5, no. 8, pp. 1888-1897, 2006.

[11] K.C. Chou and H.B. Shen, “Hum-Ploc: A Novel Ensemble Classifier for Predicting Human Protein Subcellular Localization,” *Biochemical Biophysical Research Comm.*, vol. 347, no. 8, pp. 150-157, 2006.

[12] H.B. Shen and K.C. Chou, “Gpos-Ploc: An Ensemble Classifier for Predicting Subcellular Localization of Gram-Positive Bacterial Proteins,” *Protein Eng. Design Selection*, vol. 20, no. 1, pp. 39-46, 2007.

[13] H.B. Shen and K.C. Chou, “Nuc-Ploc: A New Web-Server for Predicting Protein Subnuclear Localization by Fusing Pseaa Composition and Psepssm,” *Protein Eng. Design and Selection*, vol. 20, no. 11, pp. 561-567, 2007.

[14] K.C. Chou and H.B. Shen, “Large-Scale Plant Protein Subcellular Location Prediction,” *J. Cellular Biochemistry*, vol. 100, no. 3, pp. 665-678, 2007.

[15] K.C. Chou and H.B. Shen, “Euk-Mploc: A Fusion Classifier for Large-Scale Eukaryotic Protein Subcellular Location Prediction by Incorporating Multiple Sites,” *J. Proteome Research*, vol. 6, no. 5, pp. 1728-1734, 2007.

[16] K.C. Chou, “A Novel Approach to Predicting Protein Structural Classes in a (20-1)-d Amino Acid Composition Space,” *Proteins: Structure, Function and Genetics*, vol. 21, no. 4, pp. 319-344, 1995.

[17] K.C. Chou and Y.D. Cai, “Predicting Protein Structural Class by Functional Domain Composition,” *Biochemical and Biophysical Research Comm.*, vol. 321, no. 4, pp. 1007-1009, 2004.

[18] K.D. Kedarisetti, L.A. Kurgan, and S. Dick, “Classifier Ensembles for Protein Structural Class Prediction with Varying Homology,” *Biochemical and Biophysical Research Comm.*, vol. 348, no. 3, pp. 981-988, 2006.

[19] X. Xiao, P. Wang, and K.C. Chou, “Predicting Protein Quaternary Structural Attribute by Hybridizing Functional Domain Composition and Pseudo Amino Acid Composition,” *J. Applied Crystallography*, vol. 42, pp. 169-173, 2009.

[20] K.C. Chou and H.B. Shen, “Foldrate: A Web-Server for Predicting Protein Folding Rates from Primary Sequence,” *Open Bioinformatics J.*, vol. 3, pp. 31-50, 2009.

[21] H.B. Shen, J.N. Song, and K.C. Chou, “Prediction of Protein Folding Rates from Primary Sequence by Fusing Multiple Sequential Features,” *J. Biomedical Science and Eng.*, vol. 2, pp. 136-143, 2009.

[22] K.C. Chou and H.B. Shen, “Memtype-2l: A Web Server for Predicting Membrane Proteins and Their Types by Incorporating Evolution Information through Pse-Pssm,” *Biochemical and Biophysical Research Comm.*, vol. 360, no. 2, pp. 339-345, 2007.

[23] H.B. Shen and K.C. Chou, “Ezypred: A Top-Down Approach for Predicting Enzyme Functional Classes and Subclasses,” *Biochemical and Biophysical Research Comm.*, vol. 364, no. 1, pp. 53-59, 2007.

[24] X. Xiao, P. Wang, and K.C. Chou, “Gpcr-Ca: A Cellular Automaton Image Approach for Predicting G-Protein-Coupled Receptor Functional Classes,” *J. Computational Chemistry*, vol. 30, pp. 1414-1423, 2009.

[25] K.C. Chou, “Prediction of G-Protein-Coupled Receptor Classes,” *J. Proteome Research*, vol. 4, no. 4, pp. 1413-1418, 2004.

[26] K.C. Chou and H.B. Shen, “Protident: A Web Server for Identifying Proteases and Their Types by Fusing Functional Domain and Sequential Evolution Information,” *Biochemical Biophysical Research Comm.*, vol. 376, no. 2, pp. 321-325, 2008.

[27] K.C. Chou, "A Vectorized Sequence-Coupling Model for Predicting hiv Protease Cleavage Sites in Proteins," *J. Biological Chemistry*, vol. 269, pp. 16938-16948, 1993.

[28] K.C. Chou, "Review: Prediction of hiv Protease Cleavage Sites in Proteins," *Analytical Biochemistry*, vol. 233, pp. 1-14, 1996.

[29] H.B. Shen and K.C. Chou, "Hivcleave: A Web-Server for Predicting hiv Protease Cleavage Sites in Proteins," *Analytical Biochemistry*, vol. 375, pp. 388-390, 2008.

[30] K.C. Chou and H.B. Shen, "Review: Recent Advances in Developing Web-Servers for Predicting Protein Attributes," *Natural Science*, vol. 2, pp. 63-92, 2009.

[31] H.B. Shen, J. Yang, and K.C. Chou, "Euk-Ploc: An Ensemble Classifier for Large-Scale Eukaryotic Protein Subcellular Location Prediction," *Amino Acids*, vol. 33, pp. 57-67, 2007.

[32] H.B. Shen and K.C. Chou, "Hum-mploc: An Ensemble Classifier for Large-Scale Human Protein Subcellular Location Prediction by Incorporating Samples with Multiple Sites," *Biochemical Biophysiacl Research Comm.*, vol. 355, no. 4, pp. 1006-1011, 2007.

[33] R. Caruana, "Multitask Learning: A Knowledge-Based Source of Inductive Bias," *Machine Learning*, vol. 28, pp. 41-75, 1997.

[34] B. Bakker and T. Heskes, "Task Clustering and Gating for Bayesian Multi-Task Learning," *J. Machine Learing Research*, vol. 4, pp. 83-99, 2003.

[35] T. Evgeniou, C.A. Micchelli, and M. Pontil, "Learning Multiple Tasks with Kernel Methods," *J. Machine Learing Research*, vol. 6, pp. 615-637, 2005.

[36] R.K. Ando and T. Zhang, "A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data," *J. Machine Learning Research*, vol. 6, pp. 1817-1853, 2005.

[37] J. Baxter, "A Model for Inductive Bias Learning," *J. Artifical Intelligence Research*, vol. 12, pp. 149-198, 2000.

[38] S. Ben-David and R. Schuller, "Exploiting Task Relatedness for Multiple Task Learning," *Proc. Ann. Conf. Computational Learning Theory*, 2003.

[39] L. Jacob and J.-P. Vert, "Efficient Peptide-mhc-I Binding Prediction for Alleles with Few Known Binders," *Bioinformatics*, vol. 3, pp. 358-366, 2008.

[40] E.A. Argyriou and M. Pontil, "Multitask Feature Learning," *Proc. Ann. Conf. Neural Information Processing System (NIPS)*, 2006.

[41] G.M. Allenby and P.E. Rossi, "Marketing Models of Consumer Heterogeneity," *J. Econometircs*, vol. 89, nos. 1/2, pp. 57-78, 1999.

[42] N. Arora, G.M. Allenby, and J.L. Ginter, "A Hierarchical Bayes Model of Primary and Secondary Demand," *Marketing Science*, vol. 17, no. 1, pp. 29-44, 1998.

[43] N.D. Lawrence and J.C. Platt, "Learning to Learn with the Informative Vector Machine," *Proc. 21st Int'l Conf. Machine Learning*, 2004.

[44] E. Bonilla, K.M. Chai, and C. Williams, "Multi-Task Gaussian Process Prediction," *Proc. 20th Ann. Conf. Neural Information Processing Systems*, 2008.

[45] A. Schwaighofer, V. Tresp, and K. Yu, "Learning Gaussian Process Kernels via Hierarchical Bayes," *Proc. 20th Ann. Conf. Neural Information Processing Systems*, 2005.

[46] A. Argyriou, C.A. Micchelli, M. Pontil, and Y. Ying, "A Spectral Regularization Framework for Multi-Task Structure Learning," *Proc. 20th Ann. Conf. Neural Information Processing Systems*, 2008.

[47] T. Jebara, "Multi-Task Feature and Kernel Selection for Svms," *Proc. 21st Int'l Conf. Machine Learnings*, 2004.

[48] S. Bickel, J. Bogojeska, T. Lengauer, and T. Scheffer, "Multi-Task Learning for hiv Therapy Screening," *Proc. 25th Int'l Conf. Machine Learning*, pp. 56-63, 2008.

[49] J. Bi, T. Xiong, S. Yu, M. Dundra, and R. Rao, "An Improved Multi-Task Learning Approach with Applications in Medical Diagnosis," *Machine Learning and Knowledge Discovery in Databases*, vol. 5211, pp. 117-132, 2008.

[50] T. Evgeniou and M. Pontil, "Regularized Multi-Task Learning," *Proc. ACM SIGKDD*, 2004.

[51] V.N. Vapnik, *Statistical Learning Theory*. Wiley, 1998.

[52] T. Guo, S. Hua, X. Ji, and Z. Sun, "Dbsubloc: Database of Protein Subcellular Localization," *Nucleic Acids Research*, vol. 32, pp. 122-124, 2004.

[53] J. Wang, W.-K. Sung, A. Krishnan, and K.-B. Li, "Protein Subcellular Localization Prediction for Gram-Negative Bacteria Using Amino Acid Subalphabets and a Combination of Multiple Support Vector Machines," *BMC Bioinformatics*, vol. 6, p. 174, 2005.

[54] K.C. Chou and C.T. Zhang, "Review: Prediction of Protein Structural Classes," *Critical Rev. Biochemistry and Molecular Biology*, vol. 30, no. 4, pp. 275-349, 1995.

[55] K.C. Chou and H.B. Shen, "Review: Recent Progresses in Protein Subcellular Location Prediction," *Analytical Biochemistry*, vol. 370, no. 1, pp. 1-16, 2007.

[56] X.B. Zhou, C. Chen, Z.C. Li, and X.Y. Zou, "Using Chou's Amphiphilic Pseudo-Amino Acid Composition and Support Vector Machine for Prediction of Enzyme Subfamily Classes," *J. Theoretical Biology*, vol. 248, no. 3, pp. 546-551, 2007.

[57] H. Lin, "The Modified Mahalanobis Discriminant for Predicting Outer Membrane Proteins by Using Chou's Pseudo Amino Acid Composition," *J. Theoretical Biology*, vol. 252, no. 2, pp. 350-356, 2008.

[58] G.Y. Zhang and B.S. Fang, "Predicting the Cofactors of Oxidoreductases Based on Amino Acid Composition Distribution and Chou's Amphiphilic Pseudo Amino Acid Composition," *J. Theoretical Biology*, vol. 253, no. 2, pp. 310-315, 2008.

[59] G.Y. Zhang, H.C. Li, and B.S. Fang, "Predicting Lipase Types by Improved Chou's Pseudo-Amino Acid Composition," *Protein and Peptide Letters*, vol. 15, pp. 1132-1137, 2008.

[60] X. Jiang, R. Wei, T.L. Zhang, and Q. Gu, "Using the Concept of Chou's Pseudo Amino Acid Composition to Predict Apoptosis Proteins Subcellular Location: An Approach by Approximate Entropy," *Protein and Peptide Letters*, vol. 15, pp. 392-396, 2008.

[61] F.M. Li and Q.Z. Li, "Predicting Protein Subcellular Location Using Chou's Pseudo Amino Acid Composition and Improved Hybrid Approach," *Protein and Peptide Letters*, vol. 15, pp. 612-616, 2008.

[62] H. Lin, H. Ding, F.B. Guo, A.Y. Zhang, and J. Huang, "Predicting Subcellular Localization of Mycobacterial Proteins by Using Chou's Pseudo Amino Acid Composition," *Protein and Peptide Letters*, vol. 15, pp. 739-744, 2008.

[63] T. Wang, J. Yang, H.B. Shen, and K.C. Chou, "Predicting Membrane Protein Types by the llda Algorithm," *Protein and Peptide Letters*, vol. 15, pp. 915-921, 2008.

[64] Y.S. Ding and T.L. Zhang, "Using Chou's Pseudo Amino Acid Composition to Predict Subcellular Localization of Apoptosis Proteins: An Approach with Immune Genetic Algorithm-Based Ensemble Classifier," *Pattern Recognition Letters*, vol. 29, no. 13, pp. 1887-1892, 2008.

[65] C. Chen, L. Chen, X. Zou, and P. Cai, "Prediction of Protein Secondary Structure Content by Using the Concept of Chou's Pseudo Amino Acid Composition and Support Vector Machine," *Protein and Peptide Letters*, vol. 16, pp. 27-31, 2009.

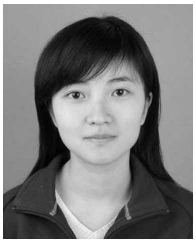
[66] H.B. Shen and K.C. Chou, "Quatident: A Web Server for Identifying Protein Quaternary Structural Attribute by Fusing Functional Domain and Sequential Evolution Information," *Biochemical Biophysical Research Comm.*, vol. 8, no. 3, pp. 1577-1584, 2009.

[67] H.B. Shen and K.C. Chou, "Identification of Proteases and Their Types," *Analytical Biochemistry*, vol. 385, no. 1, pp. 153-160, 2009.

[68] Y.S. Ding, T.L. Zhang, Q. Gu, P.Y. Zhao, and K.C. Chou, "Using Maximum Entropy Model to Predict Protein Secondary Structure with Single Sequence," *Protein and Peptide Letters*, vol. 16, pp. 552-560, 2009.

[69] H.B. Shen and K.C. Chou, "Predicting Protein Fold Pattern with Functional Domain and Sequential Evolution Information," *J. Theoretical Biology*, vol. 256, no. 3, pp. 441-446, 2009.

[70] S. Hua and Z. Sun, "Support Vector Machine Approach for Protein Subcellular Localization Prediction," *Bioinformatics*, vol. 7, pp. 721-728, 2001.



Qian Xu received the BSc degree from the Department of Computer Science and Technology, Nanjing University, China. She is currently working toward the PhD degree at the Bioengineering Program, Hong Kong University of Science and Technology.

Sinno Jialin Pan received the MS and BS degrees from the Applied Mathematics Department, Sun Yat-sen University, China, in 2003 and 2005, respectively. He is currently working toward the PhD degree at the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology. His research interests include transfer learning, semisupervised learning and their applications in pervasive computing, and web mining. He is a member of the AAAI. More information about his research can be found at http://www.cse.ust.hk/~sinnopan.

Hannah Hong Xue received the PhD degree in biochemistry from the University of Toronto, Canada. She was trained as a medical doctor in China and also did postdoctoral training in genetics from the University of Glasgow, United Kingdom. She joined the Faculty with the Department of Biochemistry in 1995, and currently serves as the director of Applied Genomics Center at The Hong Kong University of Science and Technology. She has served on the Board of Directors for International Society of Computational Biology and editorial panels and scientific reviews for several international journals. She has participated in the International HapMap Consortium and International Cancer Genome Consortium.

Qiang Yang received the bachelor's degree in astrophysics from Peking University, and the PhD degree in computer science from the University of Maryland, College Park. He is a faculty member in the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology. He is a fellow of the IEEE, a member of the AAAI and ACM, the editor in chief of the *ACM Transactions on Intelligent Systems and Technology*, a former associate editor for *IEEE Transactions on Knowledge and Data Engineering*, and a current associate editor for *IEEE Intelligent Systems*. His research interests include data mining and machine learning, AI planning, and sensor-based activity recognition. More information about his research can be found at http://www.cse.ust.hk/qyang.