Probabilistic Low-Rank Multitask Learning

Yu Kong, Member, IEEE, Ming Shao, Member, IEEE, Kang Li, and Yun Fu, Senior Member, IEEE

Abstract—In this paper, we consider the problem of learning multiple related tasks simultaneously with the goal of improving the generalization performance of individual tasks. The key challenge is to effectively exploit the shared information across multiple tasks as well as preserve the discriminative information for each individual task. To address this, we propose a novel probabilistic model for multitask learning (MTL) that can automatically balance between low-rank and sparsity constraints. The former assumes a low-rank structure of the underlying predictive hypothesis space to explicitly capture the relationship of different tasks and the latter learns the incoherent sparse patterns private to each task. We derive and perform inference via variational Bayesian methods. Experimental results on both regression and classification tasks on real-world applications demonstrate the effectiveness of the proposed method in dealing with the MTL problems.

Index Terms—Bayesian framework, low-rank modeling, multitask learning (MTL), sparse representation.

I. INTRODUCTION

In many real-world machine learning problems, learning several related tasks simultaneously rather than independently usually improves the generalization power of the learned models. The underlying reason is that valuable shared information among tasks can be easily identified and effectively utilized. For example, students’ exam scores from different schools (tasks) may share similar patterns. Thus, it would be better to identify these patterns and use them to train score predictors.

The goal of multitask learning (MTL) algorithms is to learn multiple related tasks simultaneously as opposed to single task algorithms that learn models independently. The underlying premise is that by learning tasks together, MTL algorithms can identify shared information, and share knowledge across tasks. Since these tasks are inherently related to each other, learning one task can benefit from utilizing the underlying shared information across these tasks. Therefore, better overall generalization performance can be achieved for all tasks. MTL is especially useful when training samples of each class are not sufficient or hard to obtain. In other words, learning multiple related tasks jointly can in some sense increase the sample size of each task and hence facilitate each task to gain better predictive performance.

In the past decade, researchers have approached this problem from various perspectives. Knowledge sharing has been imposed by sharing parameters [1]–[4], sharing of features [5]–[8], and sharing of structures (such as a few common basis vectors of the subspace) [9]–[11]. The flexibility of MTL makes it very useful in practical scenarios, but it also brings new challenges. First, it is desirable to incorporate multiple sharing perspectives so as to maximize the knowledge sharing instead of considering one particular factor at a time. Second, existing MTL methods put more emphasis on sharing, and however, the balance between sharing the knowledge and preserving discriminativeness of an individual task is also very important.

Existing work shows that the underlying predictive models may lie in a low-rank subspace. In the space, multiple models can be linked using some common information, such as basis vectors of the subspace. For example, the images of various object categories may share similar background, which is essentially a low-rank structure. In addition, there may be sufficient differences in multiple learning tasks, resulting in sparse discriminative features for each task. Therefore, identifying low-rank structure and sparse features are critical to MTL.

We propose a novel Bayesian MTL approach to address the aforementioned problems. In this paper, knowledge sharing is imposed by an autobalanced low-rank assumption and a sparsity constraint. These two components can well balance the useful information shared across tasks and discriminative information that is only available for an individual task. We assume that multiple related tasks lie in a low-rank subspace. The learned low-rank structure can explicitly capture the relationships among multiple tasks. In addition, inhomogeneous sparse patterns attributed to the features that are distinct to each task are also learned and preserved in each individual task. We formulate the MTL in a principled Bayesian probabilistic formulation. This allows us to learn the rank of the underlying structure and the sparsity level from data, and thereby facilitating the need for parameter tuning [11].

In our framework, we impose the low-rank constraint [12] by introducing a diagonal binary matrix to the singular value deposition (SVD) [13] of the low-rank matrix [14], where the elements of the diagonal matrix are drawn from a Beta-Bernoulli distribution. We induce sparsity in the incoherent sparse component also by assuming that the elements of a binary indicator matrix are drawn from a Beta-Bernoulli. We provide a variational inference approach.
to our probabilistic Bayesian formulation, which results in a more tractable inference compared with Markov chain Monte Carlo (MCMC) (utilized in Ding et al. [14]). Being Bayesian, our approach is able to automatically learn the rank of the underlying structure and the sparsity level, whereas in [11], they have to tune the balancing parameters.

An example of our method is shown in Fig. 1. In this figure, each task is to learn a classifier for a species of animal against the rest, and the weight matrix composed of all the classifier weight vectors can be modeled as the superposition of the low-rank and sparse components. The low-rank component explicitly models the relationship among different tasks, while the sparse one accounts for the features that are discriminative for classification of each task.

The second way for sharing knowledge is to share features. Optimization-based models impose feature sharing using $L_1$ norm regularization [5]. Joint feature selection from multiple tasks can also be made by regularization using $L_{2,1}$ norm, which encourages various learners to share similar sparsity patterns [6]. Kang et al. [30] clustered tasks and learned with whom each task should share features. A universal multitask kernel approach was investigated in [31] where the proposed approach learns related features in the kernel space. A kernelized approach was introduced in [32] for MTL. Lounici et al. [33] assumed that various task share the same sparsity patterns. Probability-based models impose feature sharing using sparsity inducing priors [34]. Archambeau et al. [7] used matrix-variate Gaussian scale mixtures and introduced a general family of group sparsity inducing priors to capture correlations between tasks.

The third way is to assume that the different tasks share common structures, for example, a few common basis vectors spanning a subspace. In optimization formulations, this is addressed by applying a low-rank constraint to the underlying predictive hypothesis structure among multiple related tasks [9]–[11], [35]. Low-rank matrix analysis considers the problem of recovering a low-rank matrix $W$ from a small random subset of its entries. Several methods were proposed to address this problem [36], [37], and they showed that the problem can be exactly solved under certain assumptions. The low-rank constraint allows MTL to extract and utilize the inherent relationships among tasks. Recently, Chen et al. [11] noted that tasks may also have sufficient differences where the discriminative features for each task is sparse; thus, they suggested learning both the low-rank subspace structure and the incoherent sparse patterns from multiple related tasks.
B. Low-Rank and Sparse Components

Tipping and Bishop [38] introduced a probabilistic formulation for principal component analysis (PCA), a common tool for extracting a low-dimensional subspace. Algorithms have been proposed in incorporating sparsity to PCA in a probabilistic framework [39]–[41]. Recently, there has been an increasing interest in robustifying PCA from noise and outliers via low-rank minimization methods [37], [42]. With the goal of making PCA robust, Ding et al. [14] provided a hierarchical Bayesian model for decomposing a matrix into low-rank and sparse components, where they infer the components through the MCMC analysis. Note here that we are addressing component decomposition in an MTL domain, which is quite different from robust PCA.

Recently, low rank and sparse modeling has been employed in regression problems for better performance and characteristics. In multivariate linear regression, the response matrix is often assumed to be low rank. To capture such property, a reduced rank ridge regression for multivariate linear regression and its kernel extension in reproducing kernel Hilbert space is proposed in [43]. The concept “Adaptive Nuclear Norm” is recently introduced to the multivariate linear regression problems to replace the prespecified weights of singular values to avoid the negative power [44]. In addition, it leads to a global optimal solution given the nonconvex regularizer. Most recently, both matrix rank and row sparsity are considered in regression problems to discover the intrinsic data structure, and select the most relevant samples [45]. The key difference between methods above and ours lies in the learning tasks: while ours attempts to find a projection matrix working on the column space, the regression models work on the row space.

C. Transfer Learning and Multitask Learning

MTL [4] has been identified as an important category of transfer learning [46]. A typical transfer learning problem includes source and target domains, and tasks therein [16]. It can be categorized by two factors: 1) availability of labels in both domains and 2) consistency of tasks. In MTL, we have different tasks, and get access to both source and target labels. In the learning stage, knowledge is transferred across different tasks, which helps jointly boost the model performance. A minor difference between conventional transfer learning and MTL is that transfer learning focuses on improving the performance of the target task, while MTL aims at improving all the tasks simultaneously.

Our work differs from previous work [11], [14], [45]. We focus on multitask feature learning, while [14] focused on learning low-dimensional features for reconstruction. In addition, the component decomposition in this paper is performed in the MTL domain, which essentially captures the correlations between tasks. Different from our work, the decomposition in [14] is performed in a single task domain, which aims to capture the intrinsic structure of the single task. Compared with [11], our method provides probabilistic explanation of data to model uncertainty, while their method does not. In addition, thanks to the Bayesian framework, the rank of the underlying structure and the sparsity level can be learned in this paper, while their method needs to be carefully tuned in [11], which is a time-consuming process. Our model is distinct from rank constrained group lasso (RCGL) method [45]. First, we model the relations between the low-rank and sparse terms using a “sum” operator, rather than a “product” operator in RCGL. Second, we use the Bayesian treatment in our model, and the estimate to the full posterior gives us a better understanding of the unknown variables. Third, our method explores the relationships between multiple tasks, while RCGL only focuses on one task.

III. PROBABILISTIC MULTITASK LEARNING

We provide a probabilistic framework for inferring the low-rank structure of the hypothesis space of multiple related tasks. We begin with formalizing our MTL problem and presenting our Bayesian framework to automatically learn the rank of the low-rank component [12] and the sparsity level of the sparse component. These two components capture the shared and private information among the different tasks, respectively. Then, a generative interpretation of the proposed framework is presented.

A. Problem Formulation

We first describe our model for regression task and will describe extensions to multiclass learning and multilabel learning models. Suppose we have m learning tasks. For each task, we need to learn a prediction function $f_i: \mathbb{R}^d \rightarrow \mathbb{R}$. The data set we have for each task is \( \{(x_{i1}^t, y_{i1}^t), \ldots, (x_{in_i}^t, y_{in_i}^t)\} \in \mathbb{R}^d \times \mathbb{R} \) \( (t = 1, \ldots, m) \), where \( n_i \) is the number of available training samples for the \( t \)th task, \( i \) indexes a sample in a single task, and \( d \) is the dimensionality of the feature vector of each sample. We assume a linear function \( f_i(x^t) = w_i^T x^t \), where \( w_i \in \mathbb{R}^d \) is the weight vector of the \( t \)th task. This gives us

$$ y_i^t = f_i(x_i^t) + \epsilon, \quad i = 1, \ldots, n_t $$

where \( \epsilon \) is sampled independently from Gaussian noise, \( \epsilon \sim \mathcal{N}(0, \rho^{-1}) \), and \( \rho \) is the noise’s precision. Therefore, we have \( p(y_i^t | x_i^t) = \mathcal{N}(w_i^T x_i^t, \rho^{-1}) \). Define the weight matrix as \( W = [w_1, \ldots, w_m] \in \mathbb{R}^{d \times m} \) by placing all the weight vectors learned from every task into a single matrix \( W \). Following our assumption that the tasks have a shared common underlying hypothesis subspace and a sparse component private to each task, we regard \( W \) as a superposition of a low-rank component \( L = [l_1, \ldots, l_m] \in \mathbb{R}^{d \times m} \) and a sparse part \( S = [s_1, \ldots, s_m] \in \mathbb{R}^{d \times m} \)

$$ W = L + S. $$

The low-rank component models the relation among tasks. If the tasks are related, we can expect that the hypothesis space is a low-rank subspace. The sparse component can sparsely select discriminative features [47].

B. Bayesian Framework

In this section, we propose to apply a hierarchical Bayesian framework for inferring the low-rank and sparse components
underlying model parameters in the MTL domain. We adopt a sparsity inducing prior on the SVD diagonal matrix of the low-rank component and the elements of the sparse component to achieve the desired rank-sparsity and elementwise sparsity. Details are given in the following.

1) Low-Rank Component $L$: After modifying the conventional SVD, the low-rank component [48] can be rewritten as

$$L = U(Z\Lambda)V$$

(3)

where $U \in \mathbb{R}^{d \times K}, Z \in \mathbb{R}^{K \times K}, \Lambda \in \mathbb{R}^{K \times K}, V \in \mathbb{R}^{K \times m}$, and $K$ is the largest possible rank we can infer for $L$.

The decomposition in (3) can be seen as a regular SVD process, with $U, V^T$ as the left and right singular vector matrices, respectively, and $Z\Lambda$ as the diagonal singular value matrix. Both $U$ and $V^T$ have orthogonal columns. Such decomposition is not unique when we have degenerated singular values, which have two linearly independent (left/right) singular vectors. If we infer the $U, V, \Lambda$, and $Z$ under the Bayesian framework in this paper, they still have similar mathematical properties. When $N$ and $K$ are large, the columns of $U$ and $V$ will be approximately orthogonal to each other. Analogously, the diagonal structure in $\Lambda$ and sparse induced term $Z$ plays the similar role as the singular value matrix with shrinkage. Also, the estimate to the full posterior is no longer a single solution, but infinite many ones with a different confidence.

The main difference between the decomposition in (3) and the SVD is that we introduce a diagonal indicator matrix $Z$ to learn the low-rank structure. Binary entries 0 or 1 are along the diagonal of $Z$: $z_{kk} \in \{0, 1\}$ ($k = 1, \ldots, K$), and thus, the rank of $L$ will be $\|Z\|_0$ and we expect this value to be as small as possible. We assume a Bernoulli distribution for the matrix $Z$ as each entry $z_{kk}$ in $Z$ is a binary variable

$$z_{kk} \sim \text{Bernoulli}(p_k), \quad (k = 1, \ldots, K).$$

(4)

Here, $p_k$ is the success probability that $z_{kk}$ takes the value 1. A Beta conjugate prior over $p_k$ is imposed to facilitate the inference

$$p_k \sim \text{Beta}(a_0, b_0)$$

(5)

where $a_0$ and $b_0$ are hyperparameters. By reducing the expectation for the success probability of the Bernoulli distribution, the probability of the diagonal elements taking value 1 is small. Hence, we can expect that the number of zero entries along the diagonal of matrix $Z$ will be as large as possible, imposing sparsity along the diagonal of matrix $Z$, and leading to the low-rank property of matrix $L$.

Diagonal entries $\lambda_{kk}$ of matrix $\Lambda$ are the singular values. We expect them to be smooth, and thus assume they are drawn from a normal-gamma distribution

$$\lambda_{kk} \sim \mathcal{N}(0, \tau^{-1}), \quad k = 1, \ldots, K$$

$$\tau \sim \text{Gamma}(a_0, b_0).$$

(6)

Here, a conjugate prior for $\tau$ is imposed using a Gamma distribution in order to penalize complex singular values in $\Lambda$. The marginal prior distribution over each $\lambda_{kk}$ has a heavier than normal tail and peaks at zero.

The column vectors of matrices $U = [u_1, \ldots, u_K]$ and $V = [v_1, \ldots, v_m]$ are assumed to be drawn from multivariate normal distributions in order to simplify the inference procedure

$$u_i \sim \mathcal{N}\left(0, \frac{1}{d}I_d\right), \quad (i = 1, \ldots, K)$$

$$v_j \sim \mathcal{N}\left(0, \frac{1}{K}I_K\right), \quad (j = 1, \ldots, m).$$

(7)

Their roles are the same as in SVD, but here we do not require orthonormality among the column vectors in $U$ and $V$. Therefore, without loss of generality, they have the Gaussian prior with zero mean. Note that their covariances are assigned in the way that they can span the space of principal vectors. Therefore, each column of $L$ can be written as

$$l_t = U(Z\Lambda)v_t = \sum_{k=1}^{K} z_{kk}\lambda_{kk} v_k u_k, \quad t = 1, \ldots, m.$$  

(8)

This equation is utilized in the inference section.

2) Sparse component $S$: The incoherent sparse component can be modeled as

$$S = B \circ E$$

(9)

where $B = [b_1, \ldots, b_m]$ is a binary matrix which is responsible for the sparsity of matrix $S$, and $\circ$ is the Hadamard pointwise product. We model the sparse term $S$ by a product, and in this way, we separate the learning of sparseness from the learning of values. As a result, the zero component in $S$ will be exactly zero. We call this “strong sparsity” in contrast to the conventional sparsity that only suppresses values in $S$.

Each binary element $b_t \in \{0, 1\}$ in $B$ can be drawn from a Bernoulli distribution

$$b_t \sim \prod_{j=1}^{d} \text{Bernoulli}(\pi_j)$$

$$\pi_j \sim \text{Beta}(\alpha_1, \beta_1).$$

(10)

Here, a conjugate prior on $\pi_j$ is imposed using a Beta distribution in order to control the sparsity level of matrix $S$. If we want to increase the sparsity of $S$, we need to decrease $\mathbb{E}[\pi_j]$.

We model the values in $E = [e_1, \ldots, e_m]$ as a normal-gamma distribution

$$e_i \sim \mathcal{N}(0, \gamma^{-1}I_d)$$

$$\gamma \sim \text{Gamma}(c_0, d_0).$$

(11)

A conjugate prior on $\gamma$ is adopted using the gamma distribution in order to reduce model complexity and increase robustness. Given $S = B \circ E$, $E$ is a Gaussian distribution with zero mean but unknown covariance. To facilitate the inference in estimating posterior, we assume the covariance’s prior as the inverse-Gamma, a popular approach in posterior estimation using variational Bayesian when the mean and covariance of Gaussian are unknown. In fact, there are extensive discussions on how to model sparsity in regression problem with Bayesian perspective, which can be seen in [15].
Bayesian model is given by

\[ p(Y_t, z_{kk}, \lambda_{kk}, u_k, v_k, e_t, \beta_t, \tau, \gamma, \pi) = p(Y_t | z_{kk}, \lambda_{kk}, u_k, v_k) \prod_k p(z_{kk} | \beta_k) p(p_k) \]

\[ \times \prod_k p(\lambda_{kk} | \tau) \prod_k p(u_k) \prod_k p(v_k) p(\tau) \]

\[ \times p(e_t | \gamma) p(\gamma) p(\beta_t) p(\pi). \]  

(12)

Notations are summarized in Table I and the graphical illustration is shown in Fig. 2.

The benefit of our Bayesian model is that it allows us to incorporate task-specific knowledge. In addition, our approach can automatically learn the rank of the underlying structure and the sparsity level without the need for manually specifying them, which requires expert knowledge [14].

### C. Extension to Multilabel and Multiclass Learning

In multilabel classification, a sample can belong to multiple classes at the same time. Our multitask framework can easily be recast to solve a multilabel problem by simply treating learning each label as a single learning task.

We can also recast our multitask framework to solve multiclass problems. We consider each task as a binary classification task of one class against all the others. Suppose we have \( m \) classes, then we will have \( m \) tasks. We consider a linear classifier, that is for each class, a linear function \( f_i(x) = w_i^T x \) is learned, where \( w_i \) is the weight vector for the \( t \)th task (class). The classification rule for a test sample \( x \) is

\[ H(x) = \arg \max_{i \in \{1,2,\ldots,m\}} w_i^T x. \]  

(13)

Given a set of labeled training samples, weight \( \{w_i\}_{i=1}^m \) can be jointly learned with our Bayesian multitask model.

Our model is also capable of making probabilistic decisions. Once the score \( w_i^T x \) is given by the linear function \( f_i(x) \), we will compute the posterior \( p(y_i^T | x) \) using a Gaussian distribution: \( p(y_i^T | x) = \mathcal{N}(w_i^T x, \rho^{-1}) \). Here, \( y \) is the regression or classification output, and \( \rho \) is the noise precision.

### IV. Variational Bayesian Inference

Exact Bayesian inference is generally intractable in many interesting latent variable models due to high-dimensional integrals associated with these models. Typically, approximation methods are applied to make them tractable. We utilize variational Bayesian approaches [49] to perform inference on the proposed model, which approximate the marginal likelihood by maximizing a lower bound, \( \mathcal{L}(q) \). Here, \( q(\Phi) \) (where \( \Phi \) represents the set of all parameters and latent variables) is the approximate distribution to the true posterior \( p(\Phi | Y) \).

We assume a distribution for \( q(\Phi) \) that factorizes over all the parameters.

The graphical model proposed in Fig. 2 is too complex to infer at first sight. If we know the value of \( z_{kk}, \lambda_{kk}, u_k, v_k \), we will immediately derive the value of \( l_i \) by Eq.(8). Similarly, if \( b_t, e_t \) are known, \( s_t \) will also be known since \( s_t = b_t \circ e_t \). From the graphical model, we have

\[ Y_t = X_t w_t + \mathcal{E}_t = X_t (l_i + s_t) + \mathcal{E}_t \]

\[ = X_t \left( \sum_{k=1}^K z_{kk} \lambda_{kk} u_k + b_t \circ e_t \right) + \mathcal{E}_t \]

(14)

for \( t = 1, \ldots, m \), where \( Y_t \) is the vector of all the targets of the \( t \)th task. \( X_t \) is all the inputs for the \( t \)th task, where every row of \( X_t \) is a data sample. \( \mathcal{E}_t \) is a Gaussian noise with zero mean, \( \mathcal{E}_t \sim \mathcal{N}(0, \rho^{-1} I) \). We consider the effect of both the low-rank part and sparse part in (14).

By mathematical deduction, we have

\[ X_t^+ Y_t = \sum_{k=1}^K z_{kk} \lambda_{kk} u_k + b_t \cdot e_t + X_t^+ \mathcal{E}_t \]  

(15)

where \( X_t^+ \) is \( X_t \) values Moore–Penrose pseudoinverse. The right-hand side of (15) has both the low-rank and sparse components coupled together, which is an obstacle for our inference task. Moreover, it has a sum of \( K \) items \( z_{kk} \lambda_{kk} u_k \), which requires more effort to break down. Here, we break it down and solve for the low-rank component and the sparse component alternately.
Bayesian Multitask Learning

\[ Y_t^k = X_t^+ Y_t - \sum_{k'=1, k' \neq k} z_{kk'} \lambda_{kk'} v_{kk'} u_{k'} - b_t \cdot e_t \]

\[ z_{kk} \lambda_{kk} v_{kk} u_k + X_t^+ e_t, \quad k = 1, \ldots, K. \] (16)

A. Low-Rank Component

Consider the low rank and keep only its \( k \)th term

\[ q(u_k) = \mathcal{N}(\mu, \Sigma), \text{ where } \mu = \sum_{i=1}^m (\langle \tilde{Y}_t^i \rangle z_{kk} v_{ki} u_k) G' \tilde{Y}_t^i \]

and \( \Sigma = \left( d D_d + \sum_{i=1}^m (\lambda_{kk} z_{kk} v_{ki}^2) G'^2 \right)^{-1} \). The other factors can be derived in a similar fashion.

B. Sparse Component

The same strategy can be applied to the sparse component

\[ Y_t^s = X_t^+ Y_s - \sum_{k=1}^K z_{kk} \lambda_{kk} v_{km} u_k \]

\[ = b_t \cdot e_t + X_t^+ e_t, \quad t = 1, \ldots, m. \] (19)

Here, \( Y_t^s \sim \mathcal{N}(b_t \cdot e_t, \rho^{-1} X_t^+ (X_t^+)^T) \). Similarly, regarding \( Y_t^s \) as the observations, the posterior of \( b_t, e_t, \gamma, \pi \) can be derived.

The joint distribution can be computed by

\[ p(Y_t^s, Z') = p(Y_t^s | b_t, e_t, \gamma, \pi) p(b_t | \pi) p(e_t | \gamma) p(\gamma) p(\pi). \]

Now, the latent variables become \( Z' = \{ b_t, e_t, \gamma, \pi \} \) and \( q(Z') \) can be factorized as \( q(Z') = q(b_t) q(e_t) q(\gamma) q(\pi) \). The update equations for the factors can be obtained as we did for the low-rank component. The main steps of model learning are provided in Algorithm 1.

The algorithm breaks the original inference task down to considering the low-rank and sparse components alternately. In each iteration, we first update the variational posterior distribution \( q(u_k), q(z_{kk}), q(\lambda_{kk}), q(v_{ki}), q(p_k) \) to derive the low-rank part and the required moments can be easily evaluated from these new distributions, which are then used for updating the next iteration. The same operations are performed for the sparse components. Note that in Algorithm 1, we monitor the variational lower bound to check the stopping criterion.

\[ \ln q^*(u_k) = \sum_{i=1}^m \left( E_{Z \setminus u_k} \left[ \ln (p(Y_t^i | Z)) \right] \right) + \ln p(u_k) + \text{const} \]

\[ = -\frac{1}{2} \sum_{i=1}^m (E_{Z \setminus u_k} \left[ (Y_t^i - z_{kk} \lambda_{kk} v_{ki} u_k)^T \right. \]

\[ \left. \cdot G' (Y_t^i - z_{kk} \lambda_{kk} v_{ki} u_k) \right] \]

\[ - \frac{1}{2} u_k^T u_k + \text{const} \] (18)

where \( G' = (\rho^{-1} X_t^+ (X_t^+)^T)^{-1} \). This is a quadratic form, and the distribution \( q^*(u_k) \) is Gaussian. Consequently, we can complete the square to identify the mean and covariance.

V. EXPERIMENTAL RESULTS

We evaluate the proposed model on both regression task and classification task. School data set and Parkinson data set are used for evaluating the performance of regression task, and four other data sets are adopted for classification task, including Extend YaleB data set, COIL object data set, Yeast data set, and MediaMill data set. We also show that the proposed method can be generalized to the multilabel error.
and multiclass tasks, and report the performance on Yeast data set and MediaMill data set. We follow previous work [14] to set hyperparameters $a_0 = b_0 = 1e - 6$ for hierarchical prior over $\tau$, $c_0 = d_0 = 1e - 6$ for hierarchical prior over $\gamma$, $a_0 = 1/K$ and $b_0 = (K - 1)/K$ for hierarchical prior over $p_k$, and $a_1 = 0.1d$ and $b_1 = 0.9d$ for hierarchical prior over $\pi$.

A. Regression

1) Data Sets and Settings: We test the regression performance of the proposed model on both the School data set and the Parkinson data set [50]. School data set consists of examination scores of 15,362 students from 139 secondary schools in London. There are 139 learning tasks, each of which corresponds to predicting student performance in each school. Each student is represented by 27 attributes, e.g., gender, VR band, and ethnic group. We take 75% of the samples from each task as training samples and the remaining as testing samples.

We also test our approach on the Parkinson data set, the second data set for regression task. Parkinson data set contains 5875 speech signals from 42 people captured by a telemonitoring device. Each speech sample is represented by 16 biomedical voice measures. The main goal of this data set is to predict the clinician’s Parkinson’s disease symptom score. We use the same split training scheme on this data set, and randomly sample 75% data instances as training samples.

We compare with convex multitask feature learning (CMTL) [6], least squares with trace norm regularization (Trace [9]), least squares with $l_1$-norm regularization (Lasso [5]), least squares with low-rank and sparse structures regularization (Sparse-LowRank [11]), and Robust MTL [10]. Experiments are repeated 20 times with randomly shuffled training data. The normalized mean squared error (nMSE) is calculated to measure their performance, which is defined as the mean squared error divided by the variance of the target vector.

2) Results: We report the regression performance of our model on both School data set and Parkinson data set.

a) School data set: In order to see how the number of tasks will affect the performance, we vary the number of tasks by setting it to 10, 30, 50, 70, 90, 110, and 130. The results of the nMSE with error bars in Fig. 3 show that our method achieves the lowest nMSE in all of the cases. Our method outperforms Sparse-LowRank and Robust MTL methods, which also use the assumption of low-rank and sparse (or structured sparse) components. Different from the two methods, we take underlying uncertainties into consideration. An interesting phenomenon is that our method, Sparse-LowRank and Robust MTL, is generally better than the other three methods, which either only have a trace norm regularization or only have a sparsity assumption of the model. This indicates that the modeling of both the underlying low-rank component and the sparse component can greatly help improve the performance. The low-rank component exploits the relationship of the tasks and the sparse component can tell the information specific to each task. The results also show that as the number of tasks increases, the error of most of the methods tend to increase. This is due to the increased complexity of more tasks being added.

We also test our method with different training ratios. A small training set is composed by randomly selecting 10%, 20%, and 30% samples from the entire training data of all the 139 tasks. The rest are used as testing set. We compare with Lasso, Trace, CMTL, Sparse-LowRank, and Robust MTL, and the experiment is repeated 20 times. The mean and the standard deviation of nMSE results in Table II indicate that our method achieves significantly better performance over all the other comparison methods.

We observe a similar phenomenon that ours, Robust MTL, and Sparse-LowRank generally perform better than the other three methods. Our method can achieve comparable or better performance than the optimization-based methods at different training ratio settings. It is reasonable to use the low-rank subspace assumption to model the relatedness among tasks, because the methods with this assumption can achieve relatively better performance than those without it (CMTL and Lasso). For example, Lasso and CMTL do not utilize a low-rank assumption. In most of the time, their nMSE is greater than Sparse-LowRank, Robust MTL, and ours, which all have the low-rank subspace assumption. However, even based on the same assumption, our method can perform better than those in the optimization camp, e.g., Sparse-LowRank. This is because the optimization-based methods circumvent their convex constraints by surrogating the nonconvex norms with convex envelopes, which will lead to a certain degree of precision loss, while our method models the generative process of the low-rank and sparse components based on the Bayesian framework. Another valuable observation is that as we increase the training ratio, the values of nMSE for all competing algorithms decrease. That is, with more training samples the generalization performance improves.

b) Parkinson data set: We compare with existing methods Robust MTL, CMTL, Lasso, Sparse-LowRank, and Trace on the Parkinson’s data set, and show the mean nMSE results with error bars in Fig. 4. The results demonstrate our method outperforms these comparison methods in a majority of cases. We show significant improvements by a large margin when the number of tasks equals to 5, 10, 15, 20, 25, and 30. Our method shows comparable results with Trace and Sparse...
LowRank when more tasks are jointly trained (number of tasks equals to 35 or 40). Our method utilizes the low-rank component to capture the intrinsic complex relationships among tasks, and the sparse component to identify task-specific information. Therefore, our method archives the best overall performance. Note that all the methods have high nMSEs when the number of tasks equals to 20. This is possibly due to the noise in the data, and some tasks are not related to other tasks.

B. Classification

1) Data Sets and Settings: We evaluate our model on both multiclass and multilabel classification tasks. Multiclass task is treated as a multitask problem where each task is a classification task of one class against the rest [7], [11], [25], [29]. Performance is reported on Extend YaleB face data set and COIL object data set. The multilabel problem can also be framed as a multitask problem where learning one label is regarded as a single task. Evaluations are conducted on Yeast data set and MediaMill. Statistics of the four data sets used for classification are shown in Table III. Multiclass linear SVM is adopted as the baseline on all the four data sets. The parameter in SVM is set to 1 for the ExtendedYaleB Face data set, COIL Object data set, and MediaMill data set, and 1000 for the Yeast data set through cross validation.

2) Multiclass Learning Results: We report the performance of our model on Extended YaleB Face data set and COIL object data set. Extended YaleB Face data set contains 2414 frontal-face images of 28 subjects captured under various laboratory-controlled lighting conditions. Each subject has about 64 images. The images are normalized to the size of 32 × 32. We randomly choose 30 and 40 training samples, respectively, from each subject and use the rest as testing samples. We compare with least squares with trace norm regularization (Trace [9]), least squares with low-rank and sparse structures regularization (Sparse-LowRank [11], Robust MTL [10]), MSDA [55], MTAE [56], and DAE [57]. Experiments are repeated 20 times with randomly shuffled training data for the two settings.

Table IV indicates that our method achieves the best accuracy in both settings. Notice that our improvement over Robust MTL is large when 30 training samples are used. This validates the idea that MTL is more effective when fewer training samples are available. Our method also outperforms deep learning approaches MSDA, MTAE, and DAE. Our method captures intrinsic structure between tasks using a low-rank matrix, while the structure information is not considered in MSDA and DAE. MTAE captures structure information between tasks by reconstructing a task using other tasks. This assumption may not work well when samples in various tasks (classes) differ significantly. We also perform a significance test with linear SVM and Robust MTL. The results in Table V show that our method is statistically better than linear SVM and Robust MTL. Our method significantly outperforms the Lasso and CMTL, which only achieve 10% accuracy (not shown in Table IV). This shows the superiority of low-rank assumption in our method.

a) COIL object data set: It contains images of 100 objects captured from various views under controlled lighting conditions. Each object has images of 72 equally spaced views, to provide a total of 7200 images. We use the first 20 objects and randomly choose 10, 20, and
30 samples, respectively, from each object as training samples and the rest as testing. We compare with Trace [9], Sparse-LowRank [11], Robust MTL [10], MSDA [55], MTAE [56], and DAE [57]. Experiments are repeated 20 times with randomly shuffled training data.

Table VI indicates that our method consistently outperforms other comparison methods in different training settings. When the number of training samples is 20 or 30, our method can achieve over 90% accuracy, better than all the other comparison methods. The underlying reason is that our method can well capture the correlations between tasks using the low-rank assumption. Our method consistently outperforms deep methods, MSDA, MTAE, and DAE, by about 2% in all three settings. Although the three deep methods contain multiple layers of features and are expressive in representing multitask data, they do not capture structure information between tasks very well. On the contrary, our method encourages a low-rank structure on the parameter matrix, and thus is capable of capturing the structure information between tasks.

We also perform a significance test comparing with linear SVM and sparse low-rank methods on this data set. The p-value results in Table VII show that our method is statistically different from linear SVM and sparse low-rank methods.

3) Multilabel Learning Results: Yeast data set and MediaMill data set are used to evaluate our method on the multilabel problem. 250 out of 1484 image instances and 800 out of 42177 videos are randomly chosen from Yeast data set and MediaMill data set as training set, respectively. The remaining are used as test set. We treat learning each label as a single task and explore the correlation among the multiple labels in the MTL approach. We compare with CMTL [6], Lasso [5], Trace [9], Sparse-LowRank [11], Robust MTL [10], MSDA [55], MTAE [56], and DAE [57] on the two data sets. Experiments are repeated 20 times with randomly shuffled training data. The average F1 scores (computed regardless of tasks) with the standard deviations of these methods are reported in Table VIII.

The results show that our method outperforms the other comparison methods. This indicates the superiority of using uncertainty in our model. Our method, Robust MTL, and Sparse-LowRank generally perform better than the CMTL, Trace, and Lasso methods on both data sets. This shows that it is beneficial to combine low-rank and sparse modeling. Our method significantly outperforms MSDA, MTAE, and DAE on the two data sets as we capture task correlations by enforcing a low-rank structure on the parameter matrix of tasks. By comparison, MTAE reconstructs a task using other tasks by the hidden features, which may not be able to capture large task variations in multiclass scenario. Note that the task structure is not exploited in MSDA and DAE.

a) Sensitivity to hyperparameters: We also evaluate the sensitivity of our model to hyperparameters \( a_0, b_0, c_0, d_0, \alpha_0, \alpha_1, \beta_1 \). We set hyperparameters \( a_0, b_0, c_0, d_0 \) to \( 10^{-6}, 10^{-4}, 10^{-2}, 1 \), respectively, and set \( a_0 \) to \( 0.1/K, 1/K, 10/K, 1 \). The F1 scores on the Yeast data set are reported in Table IX and X. The results indicate that our method is completely insensitive to these hyperparameters on Yeast data set. Our method achieves a consistent F1 score of 0.5279 using various parameter values of parameters \( a_0, b_0, c_0, d_0, a_0, \beta_0, a_1, \alpha_1 \). Therefore, parameter tuning is not an issue in our method.

We also test the sensitivity on MediaMill data set. Hyperparameters \( a_0, b_0, c_0, d_0 \) are set to \( 10^{-6}, 10^{-4}, 10^{-2}, 1 \), respectively, and \( a_1 \) is set to 0.01d, 0.1d, 1d, 10d. F1 scores shown in Tables XI and XII demonstrate that our method is also robust to these parameters on MediaMill data set. The proposed method archives a consistent F1 score of 0.5411 given various parameter values.

The underlying reason is that all these parameters are just hyperparameters of prior distributions in our model, whose influences will be finally reduced in multiple iterations in model training. This demonstrates the robustness of our model to these hyperparameters. However, the choices of probability distributions are particularly designed for the MTL task we address in this paper. They are carefully developed and used on the basis reasonable assumptions in our method. Other distributions are not appropriate for this task and may result in poor performance.

4) Discussion: From all the classification experiments, we see that our method can not only work for image classification and multilabel classification tasks, but can also achieve the

\( F_1 \) scores using various parameter values of parameters \( \beta_0, a_1, \beta_1 \) are not shown here but we obtain the consistent results in our experiments.

---

**TABLE VI**

**ACCURACY ON COIL DATA SET. THE NUMBER OF TRAINING SAMPLES PER CLASS N IS SET TO BE 10, 20, AND 30, RESPECTIVELY**

<table>
<thead>
<tr>
<th></th>
<th>N = 10</th>
<th>N = 20</th>
<th>N = 30</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear SVM</td>
<td>82% ± 2%</td>
<td>86% ± 1%</td>
<td>89% ± 1%</td>
</tr>
<tr>
<td>Trace [9]</td>
<td>42% ± 2%</td>
<td>77% ± 2%</td>
<td>84% ± 2%</td>
</tr>
<tr>
<td>Sparse-LowRank [11]</td>
<td>84% ± 1%</td>
<td>89% ± 1%</td>
<td>90% ± 1%</td>
</tr>
<tr>
<td>Robust MTL [10]</td>
<td>61% ± 2%</td>
<td>63% ± 1%</td>
<td>64% ± 1%</td>
</tr>
<tr>
<td>DAE [57]</td>
<td>84% ± 2%</td>
<td>88% ± 2%</td>
<td>90% ± 2%</td>
</tr>
<tr>
<td>MTAE [56]</td>
<td>83% ± 2%</td>
<td>89% ± 1%</td>
<td>91% ± 1%</td>
</tr>
<tr>
<td>MSDA [55]</td>
<td>84% ± 2%</td>
<td>89% ± 1%</td>
<td>91% ± 1%</td>
</tr>
<tr>
<td>Ours</td>
<td>86% ± 1%</td>
<td>91% ± 1%</td>
<td>93% ± 1%</td>
</tr>
</tbody>
</table>

**TABLE VII**

**SIGNIFICANCE ANALYSIS RESULTS BY T-TEST ON COIL DATA SET**

<table>
<thead>
<tr>
<th></th>
<th>N = 10</th>
<th>N = 20</th>
<th>N = 30</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear SVM</td>
<td>( 4.603 \times 10^{-12} )</td>
<td>( 6.578 \times 10^{-12} )</td>
<td>( 4.864 \times 10^{-12} )</td>
</tr>
<tr>
<td>Sparse-LowRank [11]</td>
<td>( 4.167 \times 10^{-12} )</td>
<td>( 1.093 \times 10^{-14} )</td>
<td>( 2.071 \times 10^{-13} )</td>
</tr>
</tbody>
</table>

**TABLE VIII**

**F1 SCORES OF COMPARISON ALGORITHMS ON THE YEAST AND MEDIAHILL DATA SET**

<table>
<thead>
<tr>
<th></th>
<th>Yeast</th>
<th>MediaMill</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear SVM</td>
<td>0.3970 ± 0.0316</td>
<td>0.5031 ± 0.0102</td>
</tr>
<tr>
<td>Robust MTL [10]</td>
<td>0.4557 ± 0.0089</td>
<td>0.5205 ± 0.0053</td>
</tr>
<tr>
<td>Sparse-LowRank [11]</td>
<td>0.4603 ± 0.0127</td>
<td>0.5070 ± 0.0062</td>
</tr>
<tr>
<td>CMTL [6]</td>
<td>0.4331 ± 0.0090</td>
<td>0.4734 ± 0.0098</td>
</tr>
<tr>
<td>Trace [9]</td>
<td>0.4425 ± 0.0443</td>
<td>0.4472 ± 0.0033</td>
</tr>
<tr>
<td>Lasso [5]</td>
<td>0.4382 ± 0.0443</td>
<td>0.4472 ± 0.0003</td>
</tr>
<tr>
<td>DAE [57]</td>
<td>0.4081 ± 0.0273</td>
<td>0.4507 ± 0.0017</td>
</tr>
<tr>
<td>MTAE [56]</td>
<td>0.3281 ± 0.0299</td>
<td>0.5100 ± 0.0074</td>
</tr>
<tr>
<td>MSDA [55]</td>
<td>0.4783 ± 0.0259</td>
<td>0.5100 ± 0.0074</td>
</tr>
<tr>
<td>Ours</td>
<td>0.5279 ± 0.0178</td>
<td>0.5411 ± 0.0053</td>
</tr>
</tbody>
</table>
Our method offers more improvement over others when fewer training examples are available. This validates that learning on several datasets from relatively simple to more complex data. Best results compared with other state-of-the-art methods on different classes in both Extended YaleB and COIL data sets are obtained. From the results on Yeast and MediaMill data sets, we see that MTL fits well for learning multiple labels, and it turns out the low-rank component can also model the task relatedness using a low-rank subspace based on a priori knowledge.

VI. CONCLUSION

In this paper, we propose a Bayesian framework to learn the sparse and low-rank patterns from multiple tasks. We model the task relatedness using a low-rank subspace based on a priori knowledge. We use variational inference to learn the latent variables. Our experimental results on both regression and classification tasks (including multilabel classification) demonstrate that the proposed method can achieve better performance over the state-of-the-art MTL methods. The results also show the effectiveness of the low rank and sparse components in our model. The performance of different methods on several data sets from relatively simple to more complex data suggests that the MTL methods can work well if different tasks have strong correlations.

| TABLE IX |
| F1 Scores of Our Method Given Various Values of Parameter $a_0$, $b_0$, $c_0$, and $d_0$ on Yeast Data Set |
| Parameter | $10^{-6}$ | $10^{-4}$ | $10^{-2}$ | 1 |
| $a_0, b_0, c_0, d_0$ |
| $F1$ score | 0.5279 | 0.5279 | 0.5279 |

| TABLE X |
| F1 Scores of Our Method Given Various Values of Parameter $a_0$ on Yeast Data Set |
| Parameter $a_0$ | $0.1/K$ | $1/K$ | $10/K$ | 1 |
| $F1$ score | 0.5279 | 0.5279 | 0.5279 |

| TABLE XI |
| F1 Scores of Our Method Given Various Values of Parameters $a_0$, $b_0$, $c_0$, and $d_0$ on MediaMill Data Set |
| Parameter | $10^{-6}$ | $10^{-4}$ | $10^{-2}$ | 1 |
| $a_0, b_0, c_0, d_0$ |
| $F1$ score | 0.5411 | 0.5411 | 0.5411 |

| TABLE XII |
| F1 Scores of Our Method Given Various Values of Parameter $a_1$ on MediaMill Data Set |
| Parameter $a_1$ | 0.01d | 0.1d | 1d | 10d |
| $F1$ score | 0.5411 | 0.5411 | 0.5411 |

 REFERENCES

[31] A. Caponnetto, A. C. Michielli, M. Pontil, and Y. Ying, “Univer-


[38] M. E. Tipping and C. M. Bishop, “Probabilistic principal compo-


[53] A. Elissonoff and J. Weston, “A kernel method for multi-labelled clas-


Yu Kong (M’15) received the B.Eng. degree in computer science, the B.S. degree in applied mathematics, and the M.E. degree in computer science from Beihang University, Beijing, China, in 2006, 2007, and 2010, respectively, and the Ph.D. degree in computer engineering from Northeastern University, Boston, MA, USA, in 2016. He is a Tenure-Track Assistant Professor affiliated with the College of Engineering, University of Massachusetts Dartmouth, Dartmouth, MA, USA, since 2016. His current research interests include sparse modeling, low-rank matrix analysis, deep learning, and applied machine learning on social media analytics.

Dr. Kong has served as a Peer Reviewer of the IEEE TRANSACTIONS, including the IEEE TRANSACTION ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE and the IEEE TRANSACTION ON CYBERNETICS.

Ming Shao (M’16) received the B.E. degree in computer science, the B.S. degree in applied mathematics, and the M.E. degree in computer science from Xi’an Jiaotong University, Xi’an, China, in 2006, 2007, and 2010, respectively, and the Ph.D. degree in computer engineering from Northeastern University, Boston, MA, USA, in 2015. He is currently an Interdisciplinary Faculty Member affiliated with the College of Engineering and the College of Computer and Information Science, Northeastern University, Boston, MA, USA, since 2012. His current research interests include machine learning, computational intelligence, big data mining, computer vision, pattern recognition, and cyber-physical systems.

Dr. Fu has extensive publications in leading journals, books/book chapters and international conferences/workshops. He serves as an Associate Editor, Chairs, PC Member, and Reviewer of many top journals and international conferences/ workshops. He received seven Prestigious Young Investigator Awards from the NAE, ONR, ARO, IEEE, INNS, UIUC, and Grainger Foundation; seven Best Paper Awards from the IEEE, IAPR, SPIE, and SIAM; three major Industrial Research Awards from Google, Samsung, and Adobe. He is currently an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS. He is a fellow of IAPR, a Lifetime Senior Member of ACM and SPIE, a Lifetime Member of AAOI, OSA, and Institute of Mathematical Statistics, a member of Global Young Academy, INNS and Beckman Graduate Fellow from 2007 to 2008.