
Multitask Learning using Nonparametrically Learned Predictor Subspaces

Piyush Rai and Hal Daumé III
School of Computing, University of Utah
{piyush,hal}@cs.utah.edu

Abstract

Given several related learning tasks, we propose a nonparametric Bayesian learning model that captures task relatedness by assuming that the task parameters (i.e., weight vectors) share a latent subspace. More specifically, the intrinsic dimensionality of this subspace is not assumed to be known *a priori*. We use an infinite latent feature model - the Indian Buffet Process - to automatically infer this number. We also propose extensions of this model where the subspace learning can incorporate (labeled, and additionally unlabeled if available) examples, or the task parameters share a *mixture* of subspaces, instead of sharing a *single* subspace. The latter property can allow learning nonlinear manifold structure underlying the task parameters, and can also help in preventing negative transfer from outlier tasks.

1 Introduction

We present a multitask learning model which assumes that the task parameters (i.e., weight vectors) share an underlying *basis space*, accounting for the task relatedness. Each task can then be represented as a linear combination of the set of basis tasks. We present two models to learn such a subspace: a) when all task parameters share a *single* subspace, and b) when they share a *mixture* of subspaces. Our subspace learning models are nonparametric (i.e., intrinsic dimensionality of the subspaces, and, in the mixture case, the intrinsic dimensionality *and* number of component subspaces need not to be known *a priori*). Here, we concentrate on probabilistic settings for prediction, namely Bayesian linear regression (for regression) and Bayesian logistic regression (for classification). The framework however is general enough and can accommodate a variety of different classification and regression tasks in probabilistic settings.

2 Latent Subspace Model for Task Parameters

To model task relatedness, we assume that the tasks have an underlying basis space and each actual task is a linear combination of the basis vectors (which act as “source” tasks). More specifically, suppose we have M tasks (regression/classification) represented by task parameters $\theta_1, \dots, \theta_M$ where $\theta_m \in \mathbb{R}^D$ is the task parameter for the m -th task. We assume the following generative model for each task parameter: $\theta_m = \mathbf{Z}\mathbf{A}_m + \epsilon_m$. Here $\mathbf{Z} \in \mathbb{R}^{D \times K}$ is a matrix in which each column is a D dimensional basis vector, and $\mathbf{A}_m \in \mathbb{R}^{K \times 1}$ is the set of coefficients for the m^{th} task parameter. The matrix \mathbf{Z} , consisting of the K basis vectors, defines the latent space underlying the set of predictors, and is shared across all tasks, justifying the task relatedness. The same generative model, with all task parameters grouped together in a matrix $\Theta = [\theta_1 \dots \theta_M] \in \mathbb{R}^{D \times M}$ can be written in a matrix form as $\Theta = \mathbf{Z}\mathbf{A}_\theta + \mathbf{E}$, where $\mathbf{A}_\theta = [\mathbf{A}_1 \dots \mathbf{A}_M]$.

Together, the matrix \mathbf{Z} of basis tasks, and the coefficients $[\mathbf{A}_1 \dots \mathbf{A}_M]$ give the task parameters a parsimonious representation where each $D \times 1$ task parameter vector is represented by a vector of size $K \times 1$, with $K \ll D$. Finally, each row e_m of the $D \times M$ matrix \mathbf{E} explains the task-specific idiosyncrasies and is assumed to be drawn from a multivariate Gaussian with a diagonal covariance matrix $\Psi = \text{diag}(\psi_{11}, \dots, \psi_{DD})$.

At a first blush, such a setup may seem like factor analysis [6]. However, unlike factor analysis, e.g., $\mathbf{X} = \mathbf{Z}\mathbf{A} + \mathbf{E}$ type of set-up where the data \mathbf{X} is *observed*, in this case the matrix Θ is *not* observed. So the “data” Θ is itself a latent variable in this model (others being \mathbf{Z} , \mathbf{A} , \mathbf{E} , and the associated hyperparameters). The goal of this model is to estimate all these using the data $\{(\mathbf{X}, \mathbf{Y}_1), \dots, (\mathbf{X}, \mathbf{Y}_M)\}$ available from the M tasks.

A crucial issue in the model is determining the intrinsic dimensionality and any underlying sparsity of the predictor subspace defined by \mathbf{Z} . We propose a nonparametric Bayesian model based on the recently proposed Indian Buffet Process (IBP) [3] to deal with both these issues. The dimensionality K of the latent space and the degree of sparsity of the basis space defined by \mathbf{Z} is automatically determined by the IBP prior. Note that the sparsity of \mathbf{Z} is akin to imposing an ℓ_1 -type regularization on \mathbf{Z} as in the lasso framework, or assuming a Laplace prior on the columns of \mathbf{Z} : $Z_k \sim \prod_{d=1}^D \text{LAPLACE}(0, \eta)$.

3 Indian Buffet Process

The Indian Buffet Process (IBP) [3] is a nonparametric Bayesian model that defines a distribution over sparse, infinite binary matrices (i.e., unbounded number of columns). The IBP was originally motivated by the need to model the latent feature structure of a given set of observations. The IBP, due to its flexibility, has been a model of choice in variety of nonparametric Bayesian applications, such as for factorial structure learning, learning causal structures, modeling dyadic data, modeling overlapping clusters, and others [3].

In the latent feature model, each observation can be thought of as consisting of a set of latent features. Given an $N \times D$ matrix \mathbf{X} of N observations having D features each, we can consider a decomposition of the form $\mathbf{X} = \mathbf{Z}\mathbf{A} + \mathbf{E}$ where \mathbf{Z} is an $N \times K$ binary feature-assignment matrix describing which features are present in each observation. $Z_{n,k}$ is 1 if feature k is present in observation n , and is otherwise 0. \mathbf{A} is a $K \times D$ matrix of feature scores, and the matrix \mathbf{E} consists of observation specific noise. A crucial issue in such models is the choosing the number K of latent features. The standard formulation of IBP lets us define a prior over the binary matrix \mathbf{Z} such that it can have an unbounded number of columns and thus can be a suitable prior in problems dealing with such structures. We use IBP to automatically determine the dimensionality of the task parameter subspace described in the previous section. Note that, in our formulation, the IBP prior is over the *latent* task parameters instead of the observations.

4 An Infinite Latent Subspace Model for Multitask Learning

In this exposition, we learn together several prediction tasks in the context of multi-label prediction, where each input \mathbf{x} is associated with multiple labels (our model is more general however and can be applied to settings when each task has its own source of input). Learning the prediction task for the m^{th} label amounts to learning the task parameter θ_m . Formally, given training data $\mathcal{D} = \{(\mathbf{x}_1, y_1^m), \dots, (\mathbf{x}_N, y_N^m)\}$ for task m where $\mathbf{x}_i \in \mathbb{R}^D$ and y_i^m is a real (for regression) or a binary valued (for classification) response, a learning task parameterized by θ_m , can be defined as $y_i^m \sim \text{Nor}(\theta_m^T \mathbf{x}_i, \rho^2)$ for regression, and $y_i^m \sim \text{Bin}(1/(1 + e^{-\theta_m^T \mathbf{x}_i}))$ for classification. To follow a more compact notation, we shall denote the inputs $[\mathbf{x}_1, \dots, \mathbf{x}_N]$ by the $N \times D$ matrix \mathbf{X} and the responses for all the M tasks by a $N \times M$ matrix \mathbf{Y} . With this notation, we can define the prediction setting as a probabilistic model $P(\mathbf{Y}|\Theta, \mathbf{X}) = \text{Nor}(\mathbf{Y}|\mathbf{X}^T \Theta, \rho^2 I)$ for regression (Bayesian linear regression), and $P(\mathbf{Y}|\Theta, \mathbf{X}) = \text{Bin}(1/(1 + e^{-\mathbf{X}^T \Theta}))$ for classification (Bayesian logistic regression).

Recall our original setup $\Theta = \mathbf{Z}\mathbf{A}_\theta + \mathbf{E}$. We model the matrix \mathbf{Z} using the Indian Buffet Process (IBP), thereby automatically choosing the intrinsic dimensionality of the task basis space defined by \mathbf{Z} . However since IBP defines a distribution over binary matrices and \mathbf{Z} a real-valued matrix, we model \mathbf{Z} as $\mathbf{B} \odot \mathbf{V}$, an element-wise product of a binary matrix \mathbf{B} and a real-valued matrix \mathbf{V} , both of size $D \times K$. We place an IBP prior over the binary matrix \mathbf{B} and a Gaussian prior over the real-valued matrix \mathbf{V} . Figure 1 shows the generative story of the model the corresponding graphical model(s). Error terms are not shown for the sake of brevity. Here Θ , which is itself a latent variable, acts as the “data” in the model and depends on other latent variables in the model (\mathbf{B} , \mathbf{V} , \mathbf{A}_θ , \mathbf{E}).

In the basic model (middle figure), the only available “data” for learning \mathbf{Z} is Θ (which is actually itself a latent variable under our probabilistic model). Given related but only a small number of tasks M , the $D \times M$ “data” matrix Θ may not be enough to reliably learn the basis \mathbf{Z} . This motivates our second model that also uses the inputs \mathbf{X} to improve the learning of \mathbf{Z} . Under this model (rightmost figure), it is assumed that the task parameters Θ and the inputs \mathbf{X} both shares the same basis space \mathbf{Z} , with different mixing matrices \mathbf{A}_θ and \mathbf{A}_x respectively. This model can be thought of as simultaneously discovering the task basis as well as doing dimensionality reduction for the data \mathbf{X} . In the latter model, one can simply use the learned task parameters Θ , or as well train a separate model with data $(\mathbf{A}_x, \mathbf{Y})$ in a reduced dimensional space (in our experiments, we just use the former approach). In addition, under this model, the data matrix \mathbf{X} need not only consist of

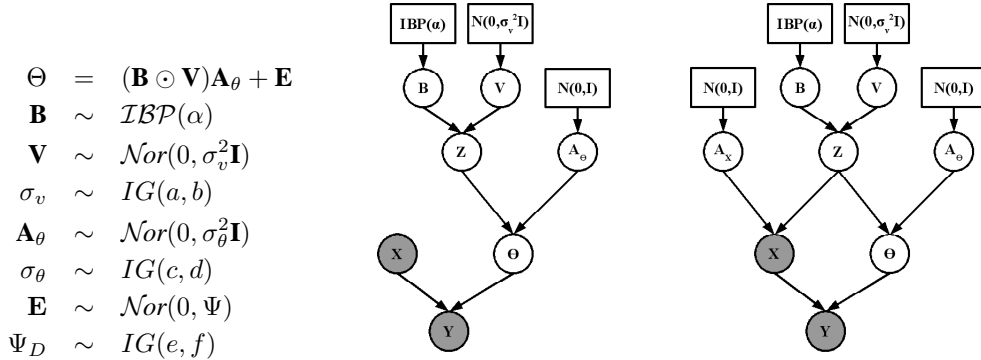


Figure 1: Generative Story and the Graphical Models (middle figure: basic model, rightmost figure: augmented model with inputs X)

labeled examples. So the matrix \mathbf{X} in the rightmost figure above can *additionally* also consists of unlabeled examples which are relatively easier to obtain.

5 Inference

We take a fully Bayesian approach for inference in these model. Inference is almost like the Gibbs sampler for the IBP [3], except that we also need to infer the latent variable Θ which acts as the “data” in our model and therefore needs to be sampled from its posterior $P(\Theta|\mathcal{D})$ where $\mathcal{D} = \{(\mathbf{x}_1, y_1^m), \dots, (\mathbf{x}_N, y_N^m)\}$, ($m = [1, \dots, M]$) denotes the actual data the model has access to.

Inference in our model is done using Gibbs sampling with a few Metropolis-Hastings steps. The samplers draws posterior samples of $\Theta, \mathbf{B}, \mathbf{V}, \mathbf{A}_\theta$ (also \mathbf{A}_x for the second model), and the remaining hyperparameters of the model. We skip the Gibbs sampling details here due to space limitation. For regression setting, posterior of Θ has a simple closed form so sampling is easily done by sampling it within the Gibbs sampler in the usual way. For classification setting, the posterior of Θ , unfortunately, does not admit a closed form so we embed an auxiliary variable sampling step [5] in our Gibbs sampler.

6 Prediction

Having learned the task parameters Θ , we use these to make predictions on the test data. For the test data \mathbf{x} of the m^{th} task, the prediction can be written as $p(y|\mathbf{x}) = \int p(y|\mathbf{x}, \theta_m)p(\theta_m|\mu_m, \Sigma_m)d\theta_m$ which is essentially averaging over the predictions made by each of the posterior samples of θ_m , where μ_m and Σ_m are the mean and covariance parameters of the m^{th} task parameter. Since the posterior averaging can be computationally expensive, it can also be replaced by $\hat{\theta}_m$, the MAP estimate of θ_m . Prediction for \mathbf{x} then simply requires plugging in the MAP estimate: $p(y|\mathbf{x}) = p(y|\mathbf{x}, \hat{\theta}_m)$.

7 Experiments

Here we demonstrate the effectiveness of our proposed approaches on two real-world multi-label classification datasets (Yeast and Scene) from the UCI repository. We compare both our models against Bayesian logistic regression trained separately on each label, and use overall accuracy, F1-Macro, F1-Micro, and AUC (Area Under ROC Curve) as the performance metrics. For the experiments, we ran for 1000 iterations both, the Bayesian logistic regression based on Gibbs sampling, and our models, and extract the MAP samples to make predictions. Results are shown in Table 1.

Model	Yeast				Scene			
	Acc	F1-macro	F1-micro	AUC	Acc	F1-macro	F1-micro	AUC
LR	0.5047	0.3415	0.3828	0.5049	0.7362	0.3132	0.3173	0.6153
Model-1	0.5212	0.3631	0.3901	0.5244	0.7756	0.3153	0.3242	0.6325
Model-2	0.5424	0.3946	0.4112	0.5406	0.7911	0.3214	0.3226	0.6416

Table 1: Comparison of Bayesian logistic regression, our basic model (model-1), and our augmented model (model-2), for two multilabel datasets. Bold face implies the best performance. Results are averaged over 10 runs with different initializations.

As the table shows, both our models do better than Bayesian logistic regression which completely ignores the task relatedness. Furthermore, our augmented model (model-2) does best overall sug-

gesting that incorporating the input data in learning the predictor subspace defined by \mathbf{Z} indeed helps in learning the task parameters even better, especially when the number of tasks is small (which is indeed the case with Yeast and Scene datasets).

8 A Mixture of Subspaces Model for Multitask Learning

Our factor analysis based predictor subspace model also admits natural extensions to more complex settings. Here, we briefly outline how a nonlinear manifold underlying the task parameters can be learned by assuming a *mixture* of subspaces model over the task parameters. In this setting, we do not assume that all task parameters share a *single* subspace, but instead there is a *mixture* of subspaces and each task parameter belongs to one of the subspaces. The mixture of subspaces model uses a collection of locally linear subspace, thereby effectively being able to model nonlinear manifolds [4]. The model also captures the notion of clustered tasks [7] in a way such that the tasks belonging to the same subspace are considered part of the same cluster. So, in essence, our framework captures task-relatedness on two levels: task clustering, followed by a subspace assumption in each cluster (and globally modeling a manifold structure underlying the task parameters). In addition, our model is fully nonparametric in that we do not need to assume an *a priori* fixed number of clusters, or a fixed dimensionality of the underlying subspaces in each cluster. The clustered structure of the task parameters also allows segregation of outlier tasks and can prevent negative transfer.

To be concrete, let us again assume that we are given M tasks $\theta_1, \dots, \theta_M$. We assume the following generative model for each task parameter θ_m : $p(\theta_m) = \sum_{i=1}^K \pi_i \text{Nor}(\mu_i, \mathbf{Z}_i \mathbf{Z}_i^T + \Psi_i)$. Here μ_i are the component means, \mathbf{Z}_i are the corresponding factor loadings, and π_i are the mixing proportions. This is essentially a mixture of factor analyzers (MFA) model originally proposed in [4] to learn the low-dimensional structure of a set of *observed* data points. However, in this model, the task parameters θ_m are actually *unobserved*. In our model, these are treated as random variables and are learned along with the rest of the model parameters for MFA. In order to make the model fully nonparametric, one can put a Dirichlet Process (DP) prior [1] on the mixing proportions π_i which nonparametrically determines K - the true number of mixture subspace components. To determine the dimensionality of each subspace, the Indian Buffet Process (IBP) prior [3] on each of the \mathbf{Z}_i just as we do for the single subspace model discussed in section 2. Inference in the model can be done by taking a fully Bayesian approach.

9 Conclusion

We have presented several flexible models for multitask learning, all of them based on the assumption that the task parameters lie on a subspace (linear, or nonlinear manifold), extending and generalizing prior work in this area [2, 8, 7]. In particular, our models proposed in section 4 extend the work of [8] making it fully nonparametric. Next, our MFA model proposed in section 8 generalizes both, the work of [7] which does only task clustering, and [8] which assumes only a fixed-dimensional and *linear* subspace for the task parameters. Thus it achieves information sharing on two levels - cluster and subspace. The mixture based model can also prevent the problem of *negative transfer* by segregating relevant tasks from the irrelevant or outlier tasks. Moreover, all our models are fully nonparametric, obviating the need of doing model selection.

References

- [1] C. E. Antoniak. Mixtures of dirichlet processes with applications to bayesian nonparametric problems. *The Annals of Statistics*, 2(6), 1974.
- [2] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *NIPS*, 2007.
- [3] Z. Ghahramani, T.L. Griffiths, and P. Sollich. Bayesian Nonparametric Latent Feature Models. In *Bayesian Statistics 8*. Oxford University Press, 2007.
- [4] Z. Ghahramani and G. E. Hinton. The EM algorithm for mixtures of factor analyzers. Technical report, 1997.
- [5] C. C. Holmes and L. Held. Bayesian Auxiliary Variable Models for Binary and Multinomial Regression. In *Bayesian Statistics 8*. Oxford University Press, 2006.
- [6] M. West. Bayesian Factor Regression Models in the “Large p, Small n” Paradigm. In *Bayesian Statistics 7*, 2003.
- [7] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task Learning for Classification with Dirichlet Process Priors. *JMLR*, 2007.
- [8] J. Zhang, Z. Ghahramani, and Y. Yang. Learning multiple related tasks using latent independent component analysis. In *NIPS*, 2006.