

Bayesian Supervised Dimensionality Reduction

Mehmet Gönen

Abstract—Dimensionality reduction is commonly used as a preprocessing step before training a supervised learner. However, coupled training of dimensionality reduction and supervised learning steps may improve the prediction performance. In this paper, we introduce a simple and novel Bayesian supervised dimensionality reduction method that combines linear dimensionality reduction and linear supervised learning in a principled way. We present both Gibbs sampling and variational approximation approaches to learn the proposed probabilistic model for multiclass classification. We also extend our formulation toward model selection using automatic relevance determination in order to find the intrinsic dimensionality. Classification experiments on three benchmark data sets show that the new model significantly outperforms seven baseline linear dimensionality reduction algorithms on very low dimensions in terms of generalization performance on test data. The proposed model also obtains the best results on an image recognition task in terms of classification and retrieval performances.

Index Terms—Dimensionality reduction, Gibbs sampling, handwritten digit recognition, image recognition, image retrieval, multiclass classification, subspace learning, variational approximation.

I. INTRODUCTION

DIMENSIONALITY reduction algorithms have three main goals: 1) removing the inherent noise to improve prediction performance; 2) obtaining low-dimensional visualizations for exploratory data analysis; and 3) reducing space and time complexities for testing. In this paper, we consider only linear dimensionality reduction algorithms, which basically project the data points to a low-dimensional subspace using a projection matrix. *Principal component analysis* (PCA) [1] and *Fisher discriminant analysis* (FDA) [2] are two well-known algorithms for unsupervised and supervised dimensionality reduction, respectively.

PCA seeks to maximize the explained variance of the data in the projected feature space and performs a linear unsupervised dimensionality reduction by calculating a projection matrix from the eigenvectors of the covariance matrix. There are also other unsupervised algorithms that try to preserve the neighborhood structure in the original space instead of maximizing the explained variance [3], [4]. However, these unsupervised

algorithms generally perform badly for classification problems due to their unsupervised nature.

FDA is a linear supervised dimensionality reduction method that jointly minimizes the within-class variance and maximizes the between-class variance. FDA has two main limitations: 1) The dimensionality of the projected subspace can be at most $K - 1$, where K is the number of classes, and 2) it assumes that each class follows a unimodal distribution, which may not always hold. Sugiyama [5] proposes a local FDA algorithm to preserve the local neighborhood structure in the data again by solving a generalized eigenvalue problem using the between-class scatter and within-class scatter matrices, which are calculated locally with the help of an affinity matrix. Similarly, Zhang *et al.* [6] construct local neighborhood patches to better impose discriminative information while learning the projection matrix. Zhang and Yeung [7] also modify the original FDA algorithm by replacing the between-class scatter and within-class scatter matrices with more suitable ones for classification tasks. Tao *et al.* [8] extend FDA by maximizing the geometric mean of the divergences between different pairs of classes and obtain better projections in terms of class separation for multiclass problems. Bian and Tao [9] propose to maximize the minimum separation between classes in the low-dimensional subspace, leading to better performance in classification tasks. Instead of considering the separations between classes, Parrish and Gupta [10] follow a different approach by minimizing an approximation to the leave-one-out training error of a local quadratic discriminant analysis classifier.

Sufficient dimension reduction (SDR) methods aim to find a subspace such that the original data points and the target outputs become conditionally independent given the low-dimensional representations [11], [12]. SDR methods differ in the way that they model the dependence between variables. The dependence measure can be, for example, mutual information [13], a squared-loss variant of mutual information [14], or a smooth estimator for mutual information [15].

For supervised learning tasks, dimensionality reduction and prediction steps are generally performed separately in a serial manner. Supervised dimensionality reduction algorithms use class labels to find a better subspace for the prediction step, but they generally have their own target functions different from the one that the learner in the projected subspace uses, leading to low prediction performance. Hence, coupled training of these two steps may improve the overall system performance.

Following this idea, Biem *et al.* [16] propose a multilayer perceptron variant that performs coupled feature extraction and classification. Coupled training of the projection matrix and the classifier is also studied in the framework of support vector machines by introducing the projection matrix into the optimization problem solved [17]–[19]. There are also metric

Manuscript received April 6, 2012; revised July 29, 2012, November 8, 2012, and January 29, 2013; accepted January 31, 2013. Date of publication March 6, 2013; date of current version November 18, 2013. This work was supported by the Academy of Finland (Finnish Centre of Excellence in Computational Inference Research (COIN) under Grant 251170). This paper was recommended by Editor S.-F. Su.

The author is with the Helsinki Institute for Information Technology, Department of Information and Computer Science, Aalto University School of Science, 00076 Espoo, Finland (e-mail: mehmet.gonen@aalto.fi).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TCYB.2013.2245321

learning methods that try to transfer the neighborhood in the input space to the projected subspace in nearest neighbor settings [20]–[23] or to minimize the empirical risk [24].

Yu *et al.* [25] propose a supervised probabilistic PCA and an efficient solution method, but the algorithm is developed only for real outputs. Rish *et al.* [26] formulate a supervised dimensionality reduction algorithm coupled with generalized linear models for binary classification and regression and maximize a target function composed of input and output likelihood terms using an iterative algorithm.

Our main motivation for this work is to formulate a joint probabilistic model of dimensionality reduction and supervised learning with a fully Bayesian treatment. We have two main goals: 1) improving prediction performance due to coupled learning of the model parameters and 2) benefiting from the advantages of a fully Bayesian formulation, namely, allowing us to integrate prior knowledge in the form of prior distributions and to do model selection.

In this paper, we propose a simple and novel *Bayesian supervised dimensionality reduction* (BSDR) method where the linear projection matrix and the supervised learning parameters of a linear model are learned together to improve prediction performance in the projected subspace. We make the following contributions. In Section II, we give the probabilistic model of our approach for multiclass classification. We then outline the Gibbs sampling approach for inference on this model. In order to reduce the computational complexity of inference and prediction steps, we introduce a deterministic variational approximation. We also extend our formulation to find the intrinsic dimensionality using *automatic relevance determination* (ARD) and discuss the key properties of our algorithm. We compare our algorithm with seven baseline linear dimensionality reduction algorithms on three benchmark data sets and one image recognition data set in Section III.

II. BAYESIAN SUPERVISED DIMENSIONALITY REDUCTION

Performing dimensionality reduction and supervised learning separately (generally with two different objective functions) may not result in a predictive subspace and may have low generalization performance. We propose to combine linear dimensionality reduction and linear supervised learning in a joint probabilistic model in order to obtain a more predictive subspace. The main idea is to map the training instances to a subspace using a linear projection matrix and to estimate the target outputs in this projected subspace. In such a case, we should consider the predictive performance of the projected subspace while learning the projection matrix. We give detailed derivations for multiclass classification, but our derivations can be easily extended to binary classification and regression estimation tasks.

Fig. 1 illustrates the proposed probabilistic model for multiclass classification with a graphical model and its distributional assumptions. The reason for choosing these specific distributions in our probabilistic model becomes clear when we explain our inference procedures. The basic steps of our algorithm can be summarized as follows: 1) The data matrix \mathbf{X} is used to project data points into a low-dimensional space using the

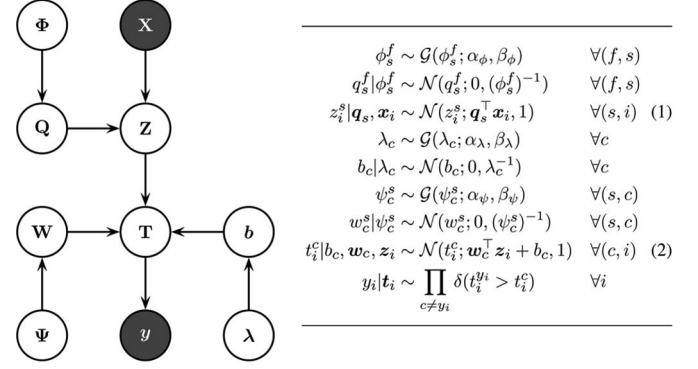


Fig. 1. BSDR for multiclass classification.

TABLE I
LIST OF NOTATION

N	Number of training instances
K	Number of classes
D	Dimensionality of input space
R	Dimensionality of projected subspace
\mathbf{X}	$D \times N$ matrix of input data
\mathbf{Q}	$D \times R$ matrix of projection variables
Φ	$D \times R$ matrix of precision priors over projection variables
\mathbf{Z}	$R \times N$ matrix of projected variables
\mathbf{W}	$R \times K$ matrix of weight parameters
Ψ	$R \times K$ matrix of precision priors over weight parameters
\mathbf{b}	$K \times 1$ vector of bias parameters
λ	$K \times 1$ vector of precision priors over bias parameters
\mathbf{T}	$N \times K$ matrix of score variables
\mathbf{y}	$N \times 1$ vector of associated target values from $\{1, \dots, K\}$

projection matrix \mathbf{Q} ; 2) the low-dimensional representations of data points \mathbf{Z} and the classification parameters $\{\mathbf{b}, \mathbf{W}\}$ are used to calculate the classification scores; and 3) finally, the given class label vector \mathbf{y} is generated from the score matrix \mathbf{T} . Different from conventional generative methods, we follow a discriminative approach in our formulation. The input data points from the original feature space are used to generate the low-dimensional representations, which are fed into a probabilistic linear classifier as input data. We basically combine the advantages of probabilistic and discriminative formulations with a fully Bayesian treatment.

The notation that we use throughout this paper is given in Table I. The superscripts index the rows of matrices, whereas the subscripts index the columns of matrices and the entries of vectors. As shorthand notations, all prior variables in the model are denoted by $\Xi = \{\lambda, \Phi, \Psi\}$, where the remaining variables are denoted by $\Theta = \{\mathbf{b}, \mathbf{Q}, \mathbf{T}, \mathbf{W}, \mathbf{Z}\}$ and the hyperparameters are denoted by $\zeta = \{\alpha_\lambda, \beta_\lambda, \alpha_\phi, \beta_\phi, \alpha_\psi, \beta_\psi\}$. Dependence on ζ is omitted for clarity throughout this paper. $\mathcal{N}(\cdot; \mu, \Sigma)$ denotes the normal distribution with the mean vector μ and the covariance matrix Σ . $\mathcal{G}(\cdot; \alpha, \beta)$ denotes the gamma distribution with the shape parameter α and the scale parameter β . $\delta(\cdot)$ denotes the Kronecker delta function that returns 1 if its argument is true and 0 if otherwise.

We use a standard projection operation to map the data points into a low-dimensional subspace and a standard linear classification model when calculating the score variables. These score variables between the class labels and the projected instances are introduced to make the inference procedures more efficient

[27]. This extra set of variables allows us to calculate full conditional distributions easily because they break the dependence between the label vector and other random variables in our graphical model. Exact inference for our probabilistic model is intractable, and we formulate two different inference strategies: Gibbs sampling and variational approximation.

A. Inference Using Gibbs Sampling

We first present our Gibbs sampling solution for inference. The basic idea behind the Gibbs sampling is to learn the posterior distributions using the full conditional distributions [28]. The joint likelihood of the graphical model in Fig. 1 can be written as

$$p(\mathbf{y}, \Theta, \Xi | \mathbf{X}) = p(\Phi)p(\mathbf{Q}|\Phi)p(\mathbf{Z}|\mathbf{Q}, \mathbf{X})p(\lambda)p(\mathbf{b}|\lambda) \\ \times p(\Psi)p(\mathbf{W}|\Psi)p(\mathbf{T}|\mathbf{b}, \mathbf{W}, \mathbf{Z})p(\mathbf{y}|\mathbf{T}).$$

We can easily find the full conditionals of the model parameters and the latent variables. The full conditional distributions are found to be from standard distributions due to the full conjugacy of our graphical model. The details of our Gibbs sampling strategy are given in Appendix A.

B. Inference Using Variational Approximation

Inference using a Gibbs sampling approach is computationally expensive. We instead formulate a deterministic variational approximation, which is comparable to Gibbs sampling in terms of generalization performance as we demonstrate with experimental results and more efficient in terms of computation time. The variational methods use a lower bound on the marginal likelihood using an ensemble of factored posteriors to find the joint parameter distribution [29]. Although the factorable ensemble implies the independence of the approximate posteriors, there is not a strong coupling between the parameters of our model. The factorable ensemble approximation of the required posterior for our model can be written as

$$p(\Theta, \Xi | \mathbf{X}, \mathbf{y}) \approx q(\Theta, \Xi) \\ = q(\Phi)q(\mathbf{Q})q(\mathbf{Z})q(\lambda)q(\Psi)q(\mathbf{b}, \mathbf{W})q(\mathbf{T})$$

and each factor in the ensemble is defined just like its full conditional distribution

$$q(\Phi) = \prod_{f=1}^D \prod_{s=1}^R \mathcal{G}(\phi_s^f; \alpha(\phi_s^f), \beta(\phi_s^f)) \\ q(\mathbf{Q}) = \prod_{s=1}^R \mathcal{N}(\mathbf{q}_s; \mu(\mathbf{q}_s), \Sigma(\mathbf{q}_s)) \\ q(\mathbf{Z}) = \prod_{i=1}^N \mathcal{N}(\mathbf{z}_i; \mu(\mathbf{z}_i), \Sigma(\mathbf{z}_i)) \\ q(\lambda) = \prod_{c=1}^K \mathcal{G}(\lambda_c; \alpha(\lambda_c), \beta(\lambda_c))$$

$$q(\Psi) = \prod_{s=1}^R \prod_{c=1}^K \mathcal{G}(\psi_c^s; \alpha(\psi_c^s), \beta(\psi_c^s)) \\ q(\mathbf{b}, \mathbf{W}) = \prod_{c=1}^K \mathcal{N}\left(\begin{bmatrix} b_c \\ \mathbf{w}_c \end{bmatrix}; \mu(b_c, \mathbf{w}_c), \Sigma(b_c, \mathbf{w}_c)\right) \\ q(\mathbf{T}) = \prod_{i=1}^N \mathcal{TN}(\mathbf{t}_i; \mu(\mathbf{t}_i), \Sigma(\mathbf{t}_i), \rho(\mathbf{t}_i))$$

where $\alpha(\cdot)$, $\beta(\cdot)$, $\mu(\cdot)$, and $\Sigma(\cdot)$ denote the shape parameter, the scale parameter, the mean vector, and the covariance matrix for their arguments, respectively. $\mathcal{TN}(\cdot; \mu, \Sigma, \rho(\cdot))$ denotes the truncated normal distribution with the mean vector μ , the covariance matrix Σ , and the truncation rule $\rho(\cdot)$ such that $\mathcal{TN}(\cdot; \mu, \Sigma, \rho(\cdot)) \propto \mathcal{N}(\cdot; \mu, \Sigma)$ if $\rho(\cdot)$ is true and $\mathcal{TN}(\cdot; \mu, \Sigma, \rho(\cdot)) = 0$ if otherwise.

We can bound the marginal likelihood using Jensen's inequality

$$\log p(\mathbf{y} | \mathbf{X}) \geq E_{q(\Theta, \Xi)} [\log p(\mathbf{y}, \Theta, \Xi | \mathbf{X})] \\ - E_{q(\Theta, \Xi)} [\log q(\Theta, \Xi)] \quad (3)$$

and optimize this bound by maximizing with respect to each factor separately until convergence. The approximate posterior distribution of a specific factor τ can be found as

$$q(\tau) \propto \exp(E_{q(\{\Theta, \Xi\} \setminus \tau)} [\log p(\mathbf{y}, \Theta, \Xi | \mathbf{X})]).$$

For our model, owing to the conjugacy, the resulting approximate posterior distribution of each factor follows the same distribution as the corresponding factor.

1) *Inference Details:* The dimensionality reduction part has two sets of parameters: the projection matrix that has normally distributed entries and the prior matrix that determines the precisions for this projection matrix. The approximate posterior distribution of the priors can be formulated as a product of gamma distributions

$$q(\Phi) = \prod_{f=1}^D \prod_{s=1}^R \mathcal{G}\left(\phi_s^f; \alpha_\phi + \frac{1}{2}, \left(\frac{1}{\beta_\phi} + \frac{(\widetilde{q_s^f})^2}{2}\right)^{-1}\right) \quad (4)$$

where the tilde notation denotes the posterior expectations as usual, i.e., $\widetilde{f(\tau)} = E_{q(\tau)}[f(\tau)]$, and we can calculate the scale parameters using the posterior sufficient statistics of the projection matrix entries. The approximate posterior distribution of the projection matrix is a product of multivariate normal distributions

$$q(\mathbf{Q}) = \prod_{s=1}^R \mathcal{N}\left(\mathbf{q}_s; \Sigma(\mathbf{q}_s) \mathbf{X} \widetilde{\mathbf{Z}}^s, \left(\text{diag}(\widetilde{\phi_s}) + \mathbf{X} \mathbf{X}^\top\right)^{-1}\right) \quad (5)$$

where the mean vectors and the covariance matrices depend on the posterior expectations of the projected instances and the priors.

The approximate posterior distribution of the projected instances can be found as a product of multivariate normal

distributions

$$q(\mathbf{Z}) = \prod_{i=1}^N \mathcal{N}\left(\mathbf{z}_i; \Sigma(\mathbf{z}_i) \left(\widetilde{\mathbf{Q}}^\top \mathbf{x}_i + \widetilde{\mathbf{W}} \widetilde{\mathbf{t}}_i - \widetilde{\mathbf{W}} \mathbf{b} \right), \left(\mathbf{I} + \widetilde{\mathbf{W}} \widetilde{\mathbf{W}}^\top \right)^{-1} \right) \quad (6)$$

where we need the posterior sufficient statistics of the supervised learning parameters to calculate the mean vectors and the covariance matrices in addition to the posterior expectations of the projection matrix and the score variables.

The supervised learning part has two sets of parameters: the bias vector and the weight matrix that have normally distributed entries and the corresponding priors that are from the gamma distribution. The approximate posterior distributions of the priors on the bias vector and the weight matrix can be formulated as products of gamma distributions

$$q(\boldsymbol{\lambda}) = \prod_{c=1}^K \mathcal{G}\left(\lambda_c; \alpha_\lambda + \frac{1}{2}, \left(\frac{1}{\beta_\lambda} + \frac{\widetilde{b}_c^2}{2}\right)^{-1}\right) \quad (7)$$

$$q(\boldsymbol{\Psi}) = \prod_{s=1}^R \prod_{c=1}^K \mathcal{G}\left(\psi_c^s; \alpha_\psi + \frac{1}{2}, \left(\frac{1}{\beta_\psi} + \frac{(\widetilde{w}_c^s)^2}{2}\right)^{-1}\right) \quad (8)$$

where the posterior sufficient statistics of the supervised learning parameters are needed to calculate the scale parameters. The approximate posterior distribution of the supervised learning parameters is a product of multivariate normal distributions

$$q(\mathbf{b}, \mathbf{W}) = \prod_{c=1}^K \mathcal{N}\left(\begin{bmatrix} b_c \\ \mathbf{w}_c \end{bmatrix}; \Sigma(b_c, \mathbf{w}_c) \begin{bmatrix} \mathbf{1}^\top \widetilde{\mathbf{t}}^c \\ \widetilde{\mathbf{Z}} \widetilde{\mathbf{t}}^c \end{bmatrix}, \begin{bmatrix} \widetilde{\lambda}_c + N & \mathbf{1}^\top \widetilde{\mathbf{Z}}^\top \\ \widetilde{\mathbf{Z}} \mathbf{1} & \text{diag}(\widetilde{\psi}_c) + \widetilde{\mathbf{Z}} \widetilde{\mathbf{Z}}^\top \end{bmatrix}^{-1} \right) \quad (9)$$

where we need the posterior sufficient statistics of the projected instances, and the posterior expectations of the priors and the score variables.

The approximate posterior distribution of the score variables is a product of truncated multivariate normal distributions

$$q(\mathbf{T}) = \prod_{i=1}^N \mathcal{TN}\left(\mathbf{t}_i; \widetilde{\mathbf{W}}^\top \widetilde{\mathbf{z}}_i + \widetilde{\mathbf{b}}, \mathbf{I}, \prod_{c \neq y_i} \delta(t_i^c > t_i^c)\right) \quad (10)$$

where the untruncated mean values depend on the posterior expectations of the projected instances and the supervised learning parameters. However, we need to find the posterior expectations of the score variables in order to update the approximate posterior distributions of the projected instances and the supervised learning parameters. We can approximate these expectations using a naive sampling approach [30].

2) *Complete Algorithm:* The complete inference algorithm is listed in Algorithm 1. The inference mechanism sequentially updates the approximate posterior distributions of the model parameters and the latent variables until convergence, which can be checked by monitoring the lower bound in (3). The first term of the lower bound corresponds to the sum of exponential forms of the distributions in the joint likelihood. The second term is the sum of negative entropies of the approximate posteriors in

the ensemble. The only nonstandard distribution in the second term is the truncated multivariate normal distributions of the score variables. We can calculate their negative entropies using the normalization constants that can be calculated while finding their expectations [30].

Algorithm 1 Bayesian Supervised Dimensionality Reduction

Require: $\mathbf{X}, \mathbf{y}, R, \alpha_\lambda, \beta_\lambda, \alpha_\phi, \beta_\phi, \alpha_\psi, \beta_\psi$

- 1) Initialize $q(\mathbf{Q}), q(\mathbf{Z}), q(\mathbf{b}, \mathbf{W})$, and $q(\mathbf{T})$ randomly
 - 2) **repeat**
 - 3) Update $q(\boldsymbol{\Phi})$ and $q(\mathbf{Q})$ using (4) and (5)
 - 4) Update $q(\mathbf{Z})$ using (6)
 - 5) Update $q(\boldsymbol{\lambda}), q(\boldsymbol{\Psi})$, and $q(\mathbf{b}, \mathbf{W})$ using (7), (8), and (9)
 - 6) Update $q(\mathbf{T})$ using (10)
 - 7) **until** convergence
 - 8) **return** $q(\mathbf{Q})$ and $q(\mathbf{b}, \mathbf{W})$
-

3) *Prediction:* In the prediction step, we can replace $p(\mathbf{Q}|\mathbf{X}, \mathbf{y})$ with its approximate posterior distribution $q(\mathbf{Q})$ and obtain the predictive distribution of the projected instance \mathbf{z}_\star for a new data point \mathbf{x}_\star as

$$p(\mathbf{z}_\star | \mathbf{x}_\star, \mathbf{X}, \mathbf{y}) = \prod_{s=1}^R \mathcal{N}(\mathbf{z}_\star^s; \mu(\mathbf{q}_s)^\top \mathbf{x}_\star, 1 + \mathbf{x}_\star^\top \Sigma(\mathbf{q}_s) \mathbf{x}_\star).$$

The predictive distribution of the score variables \mathbf{t}_\star can also be found by replacing $p(\mathbf{b}, \mathbf{W}|\mathbf{X}, \mathbf{y})$ with its approximate posterior distribution $q(\mathbf{b}, \mathbf{W})$

$$p(\mathbf{t}_\star | \mathbf{X}, \mathbf{y}, \mathbf{z}_\star) = \prod_{c=1}^K \mathcal{N}\left(t_\star^c; \mu(b_c, \mathbf{w}_c)^\top \begin{bmatrix} 1 \\ \mathbf{z}_\star \end{bmatrix}, 1 + \begin{bmatrix} 1 & \mathbf{z}_\star^\top \Sigma(b_c, \mathbf{w}_c) \end{bmatrix} \begin{bmatrix} 1 \\ \mathbf{z}_\star \end{bmatrix} \right)$$

and the predictive distribution of the class label y_\star can be formulated using these score variables

$$p(y_\star = c | \mathbf{x}_\star, \mathbf{X}, \mathbf{y}) = E_{p(u)} \left[\prod_{j \neq c} \Phi \left(\frac{u \Sigma(t_\star^c) + \mu(t_\star^c) - \mu(t_\star^j)}{\Sigma(t_\star^j)} \right) \right]$$

where the random variable u is standardized normal and $\Phi(\cdot)$ is the standardized normal cumulative distribution function. The required expectation can also be found using a naive sampling approach [30].

C. Finding Intrinsic Dimensionality Using ARD

The dimensionality of the projected subspace is generally selected using a cross-validation strategy or is fixed before learning. Instead of these two naive approaches, the intrinsic dimensionality of the data can be identified while learning the model parameters. The typical choice is to use ARD [31] that defines independent multivariate Gaussian priors on the columns of the projection matrix. The distributional assumptions for the columnwise prior case can be given as

$$\begin{aligned} \phi_s &\sim \mathcal{G}(\phi_s; \alpha_\phi, \beta_\phi) & \forall s \\ q_s^f | \phi_s &\sim \mathcal{N}(q_s^f; 0, \phi_s^{-1}) & \forall (f, s) \end{aligned}$$

and the approximate posterior distributions of the priors and the projection matrix become

$$q(\phi) = \prod_{s=1}^R \mathcal{G} \left(\phi_s; \alpha_\phi + \frac{D}{2}, \left(\frac{1}{\beta_\phi} + \frac{\widetilde{\mathbf{q}}_s^\top \mathbf{q}_s}{2} \right)^{-1} \right)$$

$$q(\mathbf{Q}) = \prod_{s=1}^R \mathcal{N} \left(\mathbf{q}_s; \Sigma(\mathbf{q}_s) \mathbf{X} \widetilde{\mathbf{z}}^s, \left(\widetilde{\phi}_s \mathbf{I} + \mathbf{X} \mathbf{X}^\top \right)^{-1} \right).$$

D. Discussion

Updating the projection matrix \mathbf{Q} using (5) is the most time-consuming step, which requires inverting $D \times D$ matrices for the covariance calculations and dominates the overall running time. When D is very large, the dimensionality of the input space should be reduced using an unsupervised dimensionality reduction method (e.g., PCA) before running the algorithm. Matrix inversion operations would increase the running time significantly when D is larger than some thousands. If the dimensionality of the input space is not very large, there is no need for this preprocessing step because it loses some of the information in the original data due to dimensionality reduction.

By multiplying the projection matrix \mathbf{Q} and the supervised learning parameters \mathbf{W} , we can find the model parameters of a linear classifier for the original data representation. However, if the number of classes is larger than the projected subspace dimensionality (i.e., $K > R$), the parameter matrix $\mathbf{Q}\mathbf{W}$ is guaranteed to be of low rank due to this decomposition leading to a more regularized solution. For multivariate regression estimation, our model can be interpreted as a full Bayesian treatment of reduced-rank regression [32].

We modify the precision priors for the projection matrix to determine the dimensionality of the projected subspace automatically. These priors can also be modified to decide which features should be used by changing entrywise priors on the projection matrix with rowwise sparse priors. This formulation allows us to perform feature selection and classification at the same time. In summary, we have three different prior formulations: 1) entrywise; 2) columnwise; and 3) rowwise. These formulations can be given as

$$\Phi_{\mathcal{E}} = \begin{bmatrix} \phi_1^1 & \phi_2^1 & \dots & \phi_R^1 \\ \phi_1^2 & \phi_2^2 & \dots & \phi_R^2 \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1^D & \phi_2^D & \dots & \phi_R^D \end{bmatrix}$$

$$\Phi_{\mathcal{C}} = \begin{bmatrix} \phi_1^D & \phi_2^D & \dots & \phi_R^D \\ \phi_1^D & \phi_2^D & \dots & \phi_R^D \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1^D & \phi_2^D & \dots & \phi_R^D \end{bmatrix}$$

$$\Phi_{\mathcal{R}} = \begin{bmatrix} \phi_1^D & \phi_1^D & \dots & \phi_1^D \\ \phi_2^D & \phi_2^D & \dots & \phi_2^D \\ \vdots & \vdots & \ddots & \vdots \\ \phi_D^D & \phi_D^D & \dots & \phi_D^D \end{bmatrix}$$

where the entrywise formulation, denoted as $\Phi_{\mathcal{E}}$, places separate priors for each entry of the projection matrix, the column-

TABLE II
COMPARISON OF GIBBS SAMPLING AND VARIATIONAL APPROXIMATION ON SMALL DATA SETS

	Method	Iris	Wine
$R = 1$	Gibbs	96.27±1.51	90.89±4.38
	Variational	95.60±1.67	89.89±4.46
$R = 2$	Gibbs	96.13±1.60	95.22±2.72
	Variational	95.60±1.67	97.67±2.19

wise formulation, denoted as $\Phi_{\mathcal{C}}$, places separate priors for each column of the projection matrix, and the rowwise formulation, denoted as $\Phi_{\mathcal{R}}$, places separate priors for each row of the projection matrix. If a particular column of $\Phi_{\mathcal{C}}$ is given a very high value during inference, the corresponding entries in the projection matrix are forced to have very small variance parameters and to become zero, eliminating the corresponding subspace dimension. Similarly, if a particular row of $\Phi_{\mathcal{R}}$ is assigned a very high value, the corresponding feature is eliminated from the model. The sparsity of the projection parameters can be tuned by changing the hyperparameters $(\alpha_\phi, \beta_\phi)$. Using sparsity-inducing priors, e.g., $(\alpha_\phi, \beta_\phi) = (0.001, 1000)$, produces results analogous to that using the ℓ_1 -norm on the projection parameters, whereas using uninformative priors, e.g., $(\alpha_\phi, \beta_\phi) = (1, 1)$, resembles using the ℓ_2 -norm.

III. EXPERIMENTS

We test our algorithm BSDR on three benchmark data sets and one image recognition data set by comparing it with seven (three unsupervised and four supervised) linear baseline algorithms, namely, PCA, *locality preserving projections* (LPPs) [3], *neighborhood preserving embedding* (NPE) [4], FDA, *neighborhood components analysis* (NCA) [20], *maximally collapsing metric learning* (MCML) [21], and *max-margin distance analysis* (MMDA) [9]. BSDR combines dimensionality reduction and supervised learning in a joint framework. In order to have comparable algorithms, we perform supervised learning using *multinomial probit* (MNP) model, after reducing dimensionality using baseline algorithms. The suffix +MNP corresponds to learning a classifier in the projected subspace using MNP. We also report the classification results obtained by MNP without dimensionality reduction to see what we lose by projecting instances to a low-dimensional subspace.

We implement both Gibbs sampling and variational approximation methods for MNP and BSDR in Matlab. Our Matlab implementation is available at <http://users.ics.aalto.fi/gonen/bsdr/>. The default hyperparameter values for MNP and BSDR are selected as $(\alpha_\lambda, \beta_\lambda, \alpha_\psi, \beta_\psi) = (1, 1, 1, 1)$ and $(\alpha_\lambda, \beta_\lambda, \alpha_\phi, \beta_\phi, \alpha_\psi, \beta_\psi) = (1, 1, 1, 1, 1, 1)$, respectively. We take 10 000 draws for the Gibbs sampling, discard the first 5000 as the burn-in, and retain every fifth realization of parameters after the burn-in for inference and prediction, giving us a total of 1000 samples, whereas we run the variational approximation for 500 iterations.

We use our own Matlab implementations for PCA, FDA, and MMDA. We solve semidefinite programming problems of MMDA using the CVX optimization toolbox [33], [34].

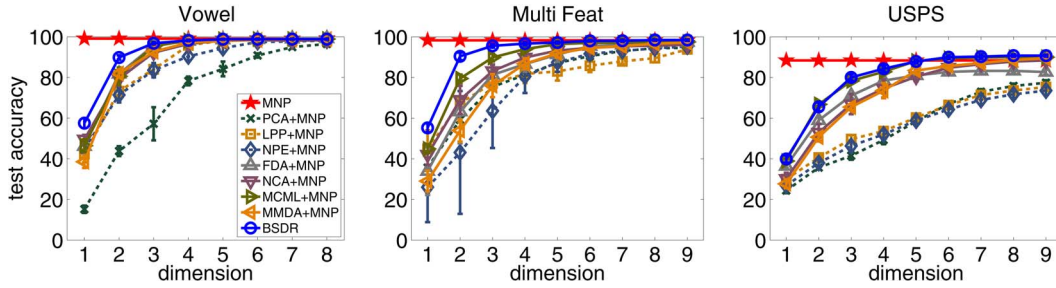


Fig. 2. Comparison of algorithms with MNP.

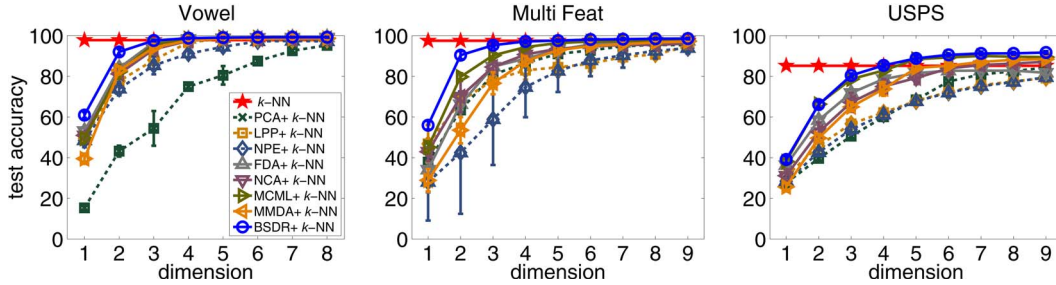


Fig. 3. Comparison of algorithms with k -NN.

LPP, NPE, NCA, and MCML implementations are integrated from the Matlab Toolbox for Dimensionality Reduction, which is publicly available at http://homepage.tudelft.nl/19j49/Matlab_Toolbox_for_Dimensionality_Reduction.html. We use the provided default parameter values for LPP, NPE, NCA, MCML, and MMDA.

For each experiment on benchmark data sets, we create ten random splits from the whole data set by dividing into two parts (one for training and one for test) and report the means and the standard deviations of the classification accuracies.

A. Gibbs Sampling Versus Variational Approximation

In order to compare Gibbs sampling and variational approximation, we perform experiments on two small benchmark data sets, namely, *Iris* and *Wine*, from the University of California, Irvine (UCI) repository, which is publicly available at <http://archive.ics.uci.edu/ml/>. Table II gives the average test accuracies on these two data sets obtained by Gibbs sampling and variational approximation. We see that variational approximation obtains very similar average test accuracies compared to Gibbs sampling. For small data sets, the training times of these two methods are very close, but the difference becomes significant for larger data sets. We use only variational approximation in the following experiments on larger data sets.

B. Classification Experiments on Benchmark Data Sets

We use three benchmark data sets to compare our algorithm with the baseline algorithms. *Vowel* is the 26-dimensional data set of a speaker performing nine different vowels 300 times per vowel collected for a vocal joystick system [35]. We use 100 data points per vowel for training. *MultiFea*t is the 649-dimensional data set of 2000 handwritten digits from the UCI repository. We use 100 data points per digit for training. *USPS* is the database of 16×16 grayscale handwritten digits

[36]. We use 100 samples per digit for training and test, giving us a data set of 2000.

Fig. 2 gives the average test accuracies on the *Vowel* data set. We see that BSRDR clearly outperforms all of the baseline algorithms for less than four dimensions, whereas all supervised algorithms obtain very similar performances after three dimensions. For example, BSRDR achieves at least 7% higher average test accuracy compared to the other algorithms when the projected subspace is 2-D. However, supervised baseline algorithms, namely, FDA+MNP, NCA+MNP, MCML+MNP, and MMDA+MNP, obtain very good average training accuracies, and this shows that they do not generalize well to out-of-sample points. Fig. 2 also gives the average test accuracies on the *MultiFea*t data set. We see that BSRDR clearly outperforms all of the baseline algorithms for less than five dimensions. Similar to the *Vowel* data set, BSRDR significantly outperforms other algorithms by more than 11% when the projected subspace is 2-D. The average test accuracies on the *USPS* data set are also given in Fig. 2. For all dimensions tried, BSRDR and MCML+MNP have higher average test accuracies compared to other algorithms. Note that we could not report the results of NCA+MNP, MCML+MNP, and MMDA+MNP on the original representations of the *MultiFea*t and *USPS* data sets due to excessive computation time. Instead, we reduce the dimensionality to 30 using PCA before training NCA, MCML, and MMDA.

We use MNP to classify projected instances when we compare BSRDR with baseline dimensionality reduction algorithms in terms of classification performance. This may give BSRDR an unfair advantage because it contains MNP in its formulation. We also replicate the experiments using *k*-nearest neighbor (k -NN) as the supervised learning algorithm after reducing dimensionality. The suffix $+k$ -NN corresponds to learning a classifier in the projected subspace using k -NN. As we can see from Fig. 3, the classification performances on the three data sets are very similar to the ones obtained using MNP. This

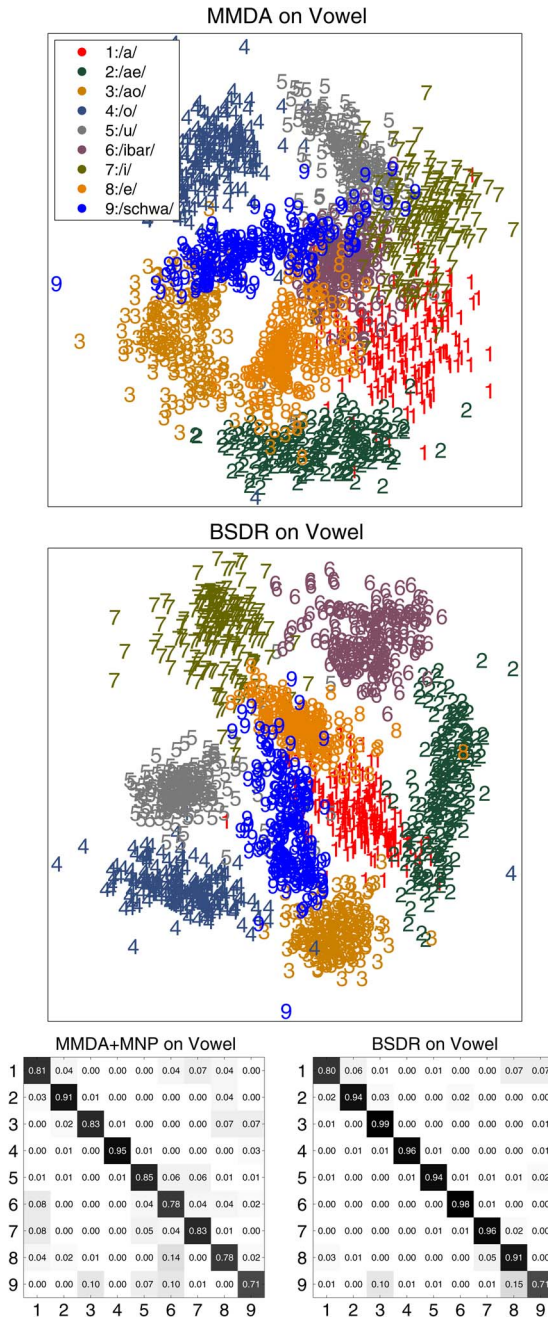


Fig. 4. 2-D embeddings and confusion matrices on the Vowel data set. Rows of confusion matrices correspond to true class labels, whereas columns correspond to predicted class labels.

shows that the superiority of BS DR on very low dimensions cannot be explained by the use of MNP only.

These three benchmark data sets are known to be “easy” classification tasks (i.e., they do not contain too much noise) in their original feature representations, and the classification performance after dimensionality reduction is not expected to become better. We see that MNP without dimensionality reduction obtains the best classification results, and this is in complete agreement with the earlier studies.

The high performance of BS DR for low-dimensional subspaces can also be observed in Fig. 4. For the 2-D subspace case on the Vowel data set, MMDA is the best working one from the baseline methods, but it cannot separate /ao/, /u/, /ibar/, /i/,

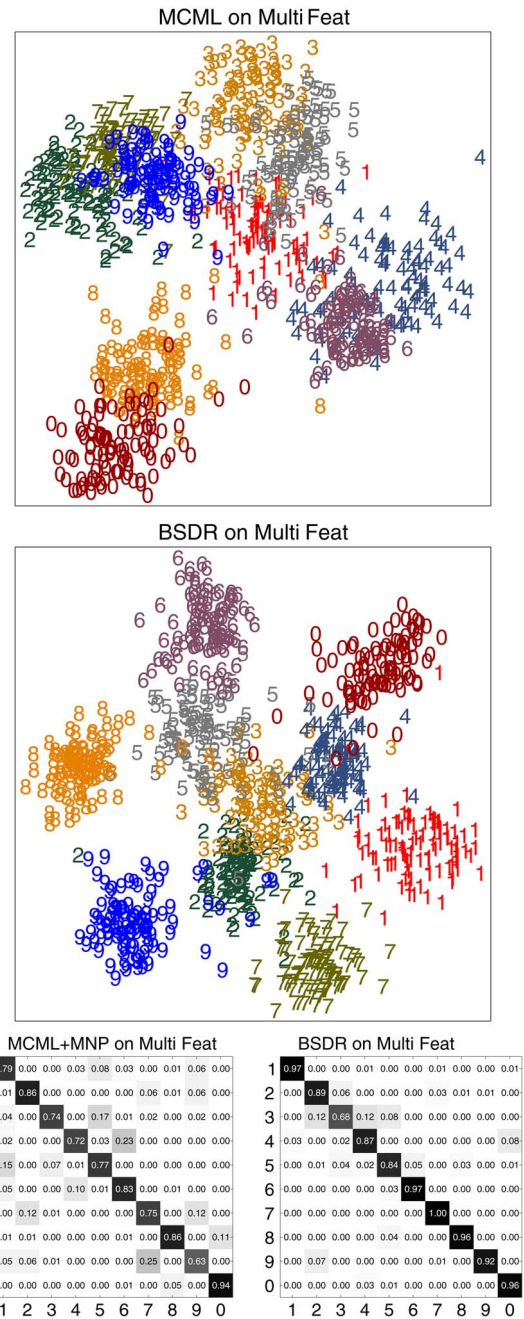


Fig. 5. 2-D embeddings and confusion matrices on the MultiFeat data set. Rows of confusion matrices correspond to true class labels, whereas columns correspond to predicted class labels.

and /e/ well as we can see from the confusion matrix. However, BS DR is able to construct well-separated clusters, which is evident from the fact that BS DR achieves 91.00% accuracy, whereas MMDA+MNP is able to get 82.83% accuracy. For the 2-D subspace case on the MultiFeat data set, which is illustrated in Fig. 5, MCML is the best working one from the baseline methods, but it cannot separate the digits well as we can see from the confusion matrix. MCML+MNP and BS DR achieve 78.90% and 90.60% accuracies, respectively.

In order to illustrate the performance of our model with ARD priors, we compare the entrywise prior with the columnwise prior on the Vowel data set. The hyperparameters of BS DR with ARD are selected as $(\alpha_\lambda, \beta_\lambda, \alpha_\phi, \beta_\phi, \alpha_\psi, \beta_\psi) = (1, 1,$

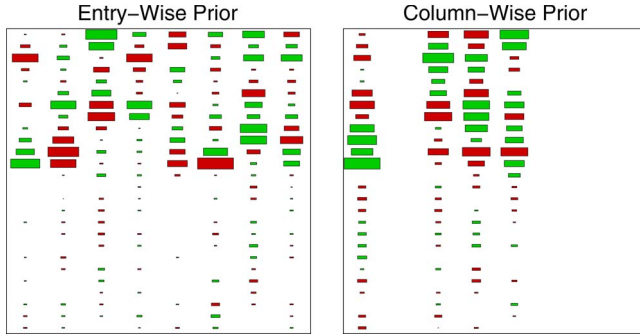


Fig. 6. Projection matrices on the Vowel data set.

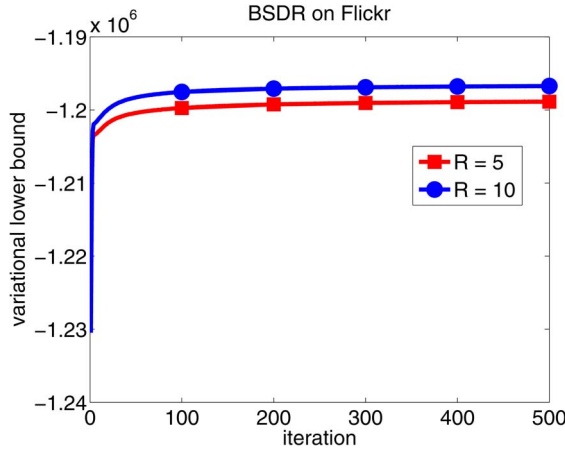


Fig. 7. Convergence behavior of BSDR on the Flickr data set.

TABLE III
EFFECT OF SAMPLE SIZE ON THE Vowel DATA SET

Method	10% of Training Data	Full Training Data
MNP	95.87 \pm 1.29	99.09 \pm 0.15
BSDR	95.01 \pm 1.14	98.81 \pm 0.25

0.001, 1000, 1, 1) to promote sparsity. Fig. 6 shows the projection matrices of these two alternatives for the case $R = 8$. We can see that four of the columns (i.e., empty columns) are assigned zero weights with the help of priors, eliminating the corresponding subspace dimensions. BSDR with ARD obtains 98.49% \pm 0.21% average test accuracy using only four out of eight subspace dimensions, whereas BSDR obtains 98.81% \pm 0.25% average test accuracy using all of the subspace dimensions.

We also perform an additional set of experiments on the Vowel data set to show the effect of sample size. We train MNP and BSDR with $R = 8$ using only 10% of the training data (i.e., ten samples per class instead of 100) and compare their classification performances with the earlier results obtained with the full training data. Table III gives the average test accuracies of these two scenarios on the Vowel data set. We see that both MNP and BSDR with the reduced training set obtain worse accuracies than with the full training set. This result is not surprising due to the discriminative nature of these algorithms. Similar to MNP, BSDR requires a moderate number of training samples to estimate the model parameters properly without overfitting.

TABLE IV
COMPARISON OF METHODS ON THE Flickr DATA SET

Method	Test Accuracy		Average Precision	
MNP	50.41		0.1974	
	$R = 5$	$R = 10$	$R = 5$	$R = 10$
PCA+MNP	32.87	39.79	0.1693	0.1755
LPP+MNP	32.06	36.26	0.1718	0.1870
NPE+MNP	31.10	36.92	0.1656	0.1840
FDA+MNP	41.56	48.27	0.3403	0.3806
NCA+MNP	42.08	49.82	0.2369	0.2678
MCML+MNP	42.96	49.01	0.2471	0.2428
MMDA+MNP	38.32	50.63	0.1769	0.2463
BSDR	47.61	54.09	0.3421	0.3838

C. Image Classification and Retrieval Experiments

We also test our algorithm BSDR in a more realistic (i.e., noisy) setup by performing classification and retrieval experiments on an image recognition data set. The Flickr image retrieval data set from [37] contains 3411 images from 13 animal categories, namely, squirrel, cow, cat, zebra, tiger, lion, elephant, whales, rabbit, snake, antlers, wolf, and hawk. Each animal image is represented using 634-dimensional low-level image features (e.g., color histogram, edge direction histogram, etc.). We use 2054 images for training and the rest for testing as provided. In retrieval experiments, each test image is considered as a separate query, and training images are ranked based on their cosine similarities with the given test image. The cosine similarity is calculated using the subspace projections obtained. A training image is taken as relevant if it belongs to the category of the test image. We evaluate the retrieval results using the mean average precision score.

Table IV shows the classification and retrieval results on the Flickr data set under two different settings, namely, $R = 5$ and 10. We see that BSDR clearly outperforms all of the baseline algorithms in terms of classification performance. BSDR is the only algorithm that can achieve a significantly better classification accuracy than MNP with the original feature representation by improving the accuracy by more than 3% using only ten dimensions. Note that we could not report the results of NCA+MNP, MCML+MNP, and MMDA+MNP on the original representation of the Flickr data set due to excessive computation time. Instead, we again reduce the dimensionality to 30 using PCA before training NCA, MCML, and MMDA.

In order to illustrate the convergence behavior of our algorithm, we report the variational lower bound of BSDR on the Flickr data set throughout 500 iterations in Fig. 7. We see that BSDR is able to quickly converge under both settings, namely, $R = 5$ and 10, before 100 iterations. We observe the same behavior for two small data sets and three benchmark data sets used in the previous experiments.

BSDR is also the best algorithm in terms of retrieval performance. For example, BSDR with $R = 10$ achieves almost twice the mean average precision of the original feature representation. Fig. 8 displays one test image from each category and the first seven training images in the ranked result list obtained by BSDR with $R = 10$ for that test image. We see









































































































Query	#1	#2	#3	Results #4	#5	#6	#7
 squirrel	 squirrel	 squirrel	 snake	 squirrel	 squirrel	 squirrel	 hawk
 cow	 cow	 elephant	 cow	 cow	 cow	 cow	 cow
 cat	 cat	 wolf	 cat	 cat	 cat	 cat	 cat
 zebra	 zebra	 zebra	 zebra	 zebra	 zebra	 zebra	 zebra
 tiger	 tiger	 tiger	 tiger	 tiger	 tiger	 tiger	 tiger
 lion	 lion	 lion	 lion	 lion	 lion	 lion	 lion
 elephant	 snake	 elephant	 elephant	 elephant	 whales	 zebra	 elephant
 whales	 whales	 whales	 whales	 whales	 whales	 whales	 whales
 rabbit	 rabbit	 rabbit	 rabbit	 rabbit	 cat	 rabbit	 wolf
 snake	 snake	 hawk	 snake	 snake	 snake	 snake	 snake
 antlers	 antlers	 antlers	 antlers	 antlers	 antlers	 antlers	 antlers
 wolf	 rabbit	 wolf	 wolf	 lion	 wolf	 rabbit	 wolf
 hawk	 hawk	 hawk	 hawk	 hawk	 hawk	 hawk	 hawk

Fig. 8. Sample queries and result images obtained by BSDR with $R = 10$ on the Flickr data set.

that the initial images in the result list are very meaningful for most of the categories even though there are some mistakes for confusing category groups such as {cat, wolf} and {rabbit, cat}. However, these mistakes are not surprising due to shape and color similarities between these groups, which make them hard to separate from each other with simple feature representations such as color histogram and edge direction histogram. Our method also decreases the computational complexity of retrieval tasks due to low-dimensional representation used for images as in *indexing* and *hashing* schemes. When we need to retrieve images similar to a query image, we can calculate the similarities between the query image and other images very fast.

IV. CONCLUSION

We present a BSDR method that couples linear dimensionality reduction and linear supervised learning in a principled way. We give detailed derivations for multiclass classification with two different inference strategies: Gibbs sampling and variational approximation. We then extend our model to find the intrinsic dimensionality automatically. Our method can easily be applied to binary classification and regression estimation tasks by changing the supervised learning part of our graphical model. Experimental results on three benchmark data sets and one image recognition data set show that our model outperforms seven baseline linear dimensionality reduction algorithms in terms of classification and retrieval performances.

The proposed model can be extended in different directions: First, we can perform a nonlinear projection using kernel values as input data, or we can use Gaussian processes instead of MNP in our probabilistic model to increase the prediction performance. Second, we can learn a unified subspace from different feature representations using multiple kernel learning similar to the classification formulation in [38]. These extensions are more meaningful for the $\mathbf{x}_i \rightarrow \mathbf{z}_i \rightarrow y_i$ dependence that we use as opposed to the $\mathbf{z}_i \rightarrow \mathbf{x}_i$ and $\mathbf{z}_i \rightarrow y_i$ dependences used in [25] and [26]. Finally, we can learn a unified subspace for multiple target outputs (i.e., multilabel learning) or multiple input representations (i.e., multitask learning).

APPENDIX

DETAILS OF GIBBS SAMPLING

A. Sampling Details

The full conditional distribution of the priors is a product of gamma distributions on the matrix entries

$$p(\Phi|\mathbf{Q}) = \prod_{f=1}^D \prod_{s=1}^R \mathcal{G}\left(\phi_s^f; \alpha_\phi + \frac{1}{2}, \left(\frac{1}{\beta_\phi} + \frac{(q_s^f)^2}{2}\right)^{-1}\right)$$

and the full conditional distribution of the projection matrix is a product of multivariate normal distributions on its columns

$$p(\mathbf{Q}|\mathbf{X}, \mathbf{Z}, \Phi) = \prod_{s=1}^R \mathcal{N}\left(\mathbf{q}_s; (\text{diag}(\phi_s) + \mathbf{X}\mathbf{X}^\top)^{-1} \times \mathbf{X}\mathbf{z}^s, (\text{diag}(\phi_s) + \mathbf{X}\mathbf{X}^\top)^{-1}\right)$$

where we can sample from these two distributions using standard methods.

The full conditional distribution of the projected instances is a product of multivariate normal distributions on the matrix columns

$$p(\mathbf{Z}|\mathbf{b}, \mathbf{Q}, \mathbf{W}, \mathbf{X}) = \prod_{i=1}^N \mathcal{N}\left(\mathbf{z}_i; (\mathbf{I} + \mathbf{W}\mathbf{W}^\top)^{-1} \times (\mathbf{Q}^\top \mathbf{x}_i + \mathbf{W}\mathbf{t}_i - \mathbf{W}\mathbf{b}), (\mathbf{I} + \mathbf{W}\mathbf{W}^\top)^{-1}\right)$$

where we can sample each data point separately.

The full conditional distributions of the priors are products of gamma distributions

$$p(\lambda|\mathbf{b}) = \prod_{c=1}^K \mathcal{G}\left(\lambda_c; \alpha_\lambda + \frac{1}{2}, \left(\frac{1}{\beta_\lambda} + \frac{b_c^2}{2}\right)^{-1}\right)$$

$$p(\Psi|\mathbf{W}) = \prod_{s=1}^R \prod_{c=1}^K \mathcal{G}\left(\psi_c^s; \alpha_\psi + \frac{1}{2}, \left(\frac{1}{\beta_\psi} + \frac{(w_c^s)^2}{2}\right)^{-1}\right)$$

and the joint full conditional distribution of the bias vector and the weight matrix is a product of multivariate normal distributions

$$p(\mathbf{b}, \mathbf{W}|\mathbf{T}, \mathbf{Z}, \lambda, \Psi) = \prod_{c=1}^K \mathcal{N}\left(\begin{bmatrix} b_c \\ \mathbf{w}_c \end{bmatrix}; \begin{bmatrix} \lambda_c + N & \mathbf{1}^\top \mathbf{Z}^\top \\ \mathbf{Z}\mathbf{1} & \text{diag}(\psi_c) + \mathbf{Z}\mathbf{Z}^\top \end{bmatrix}^{-1} \times \begin{bmatrix} \mathbf{1}^\top \mathbf{t}^c \\ \mathbf{Z}\mathbf{t}^c \end{bmatrix}, \begin{bmatrix} \lambda_c + N & \mathbf{1}^\top \mathbf{Z}^\top \\ \mathbf{Z}\mathbf{1} & \text{diag}(\psi_c) + \mathbf{Z}\mathbf{Z}^\top \end{bmatrix}^{-1}\right)$$

where each of them is defined for one class. Denison *et al.* [39] give a similar formulation for MNP models without the bias parameters on the original data points instead of the projected instances.

The full conditional distribution of the score variables is a product of truncated multivariate normal distributions

$$p(\mathbf{T}|\mathbf{b}, \mathbf{W}, \mathbf{y}, \mathbf{Z}) = \prod_{i=1}^N \mathcal{TN}\left(\mathbf{t}_i; \mathbf{W}^\top \mathbf{z}_i + \mathbf{b}, \mathbf{I}, \prod_{c \neq y_i} \delta(t_i^{y_i} > t_i^c)\right)$$

where the truncation depends on the class label. In order to sample from this truncated distribution, we can use the naive approach of drawing samples until the sample satisfies the truncation condition (i.e., rejection sampling) [27]. If the number of classes is large, this strategy may become very inefficient due to high rejection rate. In this case, we can also use a Gibbs subsampler to draw samples from this truncated multivariate normal distribution [40].

B. Prediction

In the prediction step, we need to estimate the projected instance \mathbf{z}_\star and the class label y_\star for a new data point \mathbf{x}_\star . The predictive distribution of \mathbf{x}_\star can be written as

$$p(y_\star = c|\mathbf{x}_\star, \mathbf{X}, \mathbf{y}) = \int p(y_\star = c|\mathbf{t}_\star) p(\mathbf{t}_\star|\mathbf{x}_\star, \mathbf{X}, \mathbf{y}) d\mathbf{t}_\star$$

and can be approximated by drawing samples from the full posterior. For each sample from $\{\mathbf{b}_{(l)}, \mathbf{Q}_{(l)}, \mathbf{W}_{(l)}\}_{l=B+1}^S$, we

can draw $\mathbf{z}_{\star(l)}$ using (1) and $\mathbf{t}_{\star(l)}$ using (2), where B is the number of burn-in samples and S is the number of total samples. Then, \mathbf{z}_{\star} can be approximated as

$$\mathbf{z}_{\star} = \frac{1}{S-B} \sum_{l=B+1}^S \mathbf{z}_{\star(l)}$$

and the approximate predictive distribution can be obtained by the following Monte Carlo estimate:

$$\begin{aligned} p(y_{\star} = c | \mathbf{x}_{\star}, \mathbf{X}, \mathbf{y}) \\ = \frac{1}{S-B} \sum_{l=B+1}^S E_{p(u)} \left[\prod_{j \neq c} \Phi \left(u + t_{\star(l)}^c - t_{\star(l)}^j \right) \right] \end{aligned}$$

where the expectation can again be approximated using a straightforward sampling approach.

REFERENCES

- [1] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philos. Mag.*, vol. 2, no. 6, pp. 559–572, 1901.
- [2] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, no. 2, pp. 179–188, Sep. 1936.
- [3] X. He and P. Niyogi, "Locality preserving projections," in *Proc. Adv. Neural Inf. Process. Syst.*, 2004, vol. 16, pp. 585–591.
- [4] X. He, D. Cai, S. Yan, and H.-J. Zhang, "Neighborhood preserving embedding," in *Proc. 10th IEEE Int. Conf. Comput. Vis.*, 2005, pp. 1208–1213.
- [5] M. Sugiyama, "Dimensionality reduction of multimodal labeled data by local Fisher discriminant analysis," *J. Mach. Learn. Res.*, vol. 8, pp. 1027–1061, May 2007.
- [6] T. Zhang, D. Tao, X. Li, and J. Yang, "Patch alignment for dimensionality reduction," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1299–1313, Sep. 2009.
- [7] Y. Zhang and D.-Y. Yeung, "Worst-case linear discriminant analysis," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, vol. 23, pp. 2568–2576.
- [8] D. Tao, X. Li, X. Wu, and S. J. Maybank, "Geometric mean for subspace selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 260–274, Feb. 2009.
- [9] W. Bian and D. Tao, "Max-min distance analysis by using sequential SDP relaxation for dimension reduction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 1037–1050, May 2011.
- [10] N. Parrish and M. R. Gupta, "Dimensionality reduction by local discriminative Gaussians," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012, pp. 559–566.
- [11] K.-C. Li, "Sliced inverse regression for dimension reduction," *J. Amer. Stat. Assoc.*, vol. 86, no. 414, pp. 316–327, Jun. 1991.
- [12] R. D. Cook, *Regression Graphics: Ideas for Studying Regression Through Graphics*. New York, NY, USA: Wiley, 1996.
- [13] A. Globerson and N. Tishby, "Sufficient dimensionality reduction," *J. Mach. Learn. Res.*, vol. 3, pp. 1307–1331, Mar. 2003.
- [14] T. Suzuki and M. Sugiyama, "Sufficient dimension reduction via squared-loss mutual information estimation," in *Proc. 13th Int. Conf. Artif. Intell. Stat.*, 2010, pp. 804–811.
- [15] L. Faivishevsky and J. Goldberger, "Dimensionality reduction based on non-parametric mutual information," *Neurocomputing*, vol. 80, pp. 31–37, Mar. 2012.
- [16] A. Biem, S. Katagiri, and B.-H. Juang, "Pattern recognition using discriminative feature extraction," *IEEE Trans. Signal Process.*, vol. 45, no. 2, pp. 500–504, Feb. 1997.
- [17] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Mach. Learn.*, vol. 46, no. 1–3, pp. 131–159, 2002.
- [18] A. Kocsor, K. Kovács, and C. Szepesvári, "Margin maximizing discriminant analysis," in *Proc. 15th Eur. Conf. Mach. Learn.*, 2004, pp. 227–238.
- [19] F. Pereira and G. Gordon, "The support vector decomposition machine," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 689–696.
- [20] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov, "Neighbourhood components analysis," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, vol. 17, pp. 513–520.
- [21] A. Globerson and S. Roweis, "Metric learning by collapsing classes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, vol. 18, pp. 451–458.
- [22] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *J. Mach. Learn. Res.*, vol. 10, pp. 207–244, Dec. 2009.
- [23] M. Villegas and R. Paredes, "Dimensionality reduction by minimizing nearest-neighbor classification error," *Pattern Recognit. Lett.*, vol. 32, no. 4, pp. 633–639, Mar. 2011.
- [24] W. Bian and D. Tao, "Constrained empirical risk minimization framework for distance metric learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 8, pp. 1194–1205, Aug. 2012.
- [25] S. Yu, K. Yu, V. Tresp, H.-P. Kriegel, and M. Wu, "Supervised probabilistic principal component analysis," in *Proc. 12th ACM SIGKDD*, 2006, pp. 464–473.
- [26] I. Rish, G. Grabarnik, G. Cecchi, F. Pereira, and G. J. Gordon, "Closed-form supervised dimensionality reduction with generalized linear models," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 832–839.
- [27] J. H. Albert and S. Chib, "Bayesian analysis of binary and polychotomous response data," *J. Amer. Stat. Assoc.*, vol. 88, no. 422, pp. 669–679, Jun. 1993.
- [28] A. E. Gelfand and A. F. M. Smith, "Sampling-based approaches to calculating marginal densities," *J. Amer. Stat. Assoc.*, vol. 85, no. 410, pp. 398–409, Jun. 1990.
- [29] M. J. Beal, "Variational algorithms for approximate Bayesian inference," Ph.D. dissertation, The Gatsby Computational Neuroscience Unit, University College London, London, U.K., 2003.
- [30] M. Girolami and S. Rogers, "Variational Bayesian multinomial probit regression with Gaussian process priors," *Neural Comput.*, vol. 18, no. 8, pp. 1790–1817, Aug. 2006.
- [31] R. M. Neal, *Bayesian Learning for Neural Networks*. New York, NY, USA: Springer-Verlag, 1996, ser. Lecture Notes in Statistics 118.
- [32] M. K.-S. Tso, "Reduced-rank regression and canonical analysis," *J. Roy. Stat. Soc. Ser. B (Methodol.)*, vol. 43, no. 2, pp. 183–189, 1981.
- [33] M. Grant and S. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, V. Blondel, S. Boyd, and H. Kimura, Eds. New York, NY, USA: Springer-Verlag, 2008, ser. Lecture Notes in Control and Information Sciences, pp. 95–110.
- [34] M. Grant and S. Boyd, May 2012, CVX: Matlab software for disciplined convex programming, version 1.22. [Online]. Available: <http://cvxr.com/cvx>
- [35] J. A. Bilmes, J. Malkin, X. Li, S. Harada, K. Kilanski, K. Kirchhoff, R. Wright, A. Subramanya, J. A. Landay, P. Dowden, and H. Chizeck, "The vocal joystick," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2006, pp. 625–628.
- [36] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 5, pp. 550–554, May 1994.
- [37] N. Chen, J. Zhu, and E. P. Xing, "Predictive subspace learning for multi-view data: A large margin approach," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, vol. 23, pp. 361–369.
- [38] T. Damoulas and M. Girolami, "Combining feature spaces for classification," *Pattern Recognit.*, vol. 42, no. 11, pp. 2671–2683, Nov. 2009.
- [39] D. G. T. Denison, C. C. Holmes, B. K. Mallick, and A. F. M. Smith, *Bayesian Methods for Nonlinear Classification and Regression*. Hoboken, NJ, USA: Wiley, 2002.
- [40] R. McCulloch and P. E. Rossi, "An exact likelihood analysis of the multinomial probit model," *J. Econometrics*, vol. 64, no. 1/2, pp. 207–240, Sep/Oct. 1994.



Mehmet Gönen received the B.Sc. degree in industrial engineering and the M.Sc. and Ph.D. degrees in computer engineering from Boğaziçi University, İstanbul, Turkey, in 2003, 2005, and 2010, respectively.

He was a Teaching Assistant with the Department of Computer Engineering, Boğaziçi University. He is currently doing his postdoctoral work at the Helsinki Institute for Information Technology, Department of Information and Computer Science, Aalto University School of Science, Espoo, Finland. His research interests include support vector machines, kernel methods, Bayesian methods, optimization for machine learning, dimensionality reduction, information retrieval, and computational biology applications.