Deep Max-Margin Discriminant Projection

Hao Zhang, Bo Chen, *Member, IEEE*, Zhengjue Wang, and Hongwei Liu, *Member, IEEE*

***Abstract*—In this paper, a unified Bayesian max-margin discriminant projection framework is proposed, which is able to jointly learn the discriminant feature space and the max-margin classifier with different relationships between the latent representations and observations. We assume that the latent representation follows a normal distribution whose sufficient statistics are functions of the observations. The function can be flexibly realized through either shallow or deep structures. The shallow structure includes linear, nonlinear kernel-based functions, and even the convolutional projection, which can be further trained layerwisely to build a multilayered convolutional feature learning model. To take the advantage of the deep neural networks, especially their highly expressive ability and efficient parameter learning, we integrate Bayesian modeling and the popular neural networks, for example, mltilayer perceptron and convolutional neural network, to build an end-to-end Bayesian deep discriminant projection under the proposed framework, which degenerated into the existing shallow linear or convolutional projection with the single-layer structure. Moreover, efficient scalable inferences for the realizations with different functions are derived to handle large-scale data via a stochastic gradient Markov chain Monte Carlo. Finally, we demonstrate the effectiveness and efficiency of the proposed models by the experiments on real-world data, including four image benchmarks (MNIST, CIFAR-10, STL-10, and SVHN) and one measured radar high-resolution range profile dataset, with the detailed analysis about the parameters and computational complexity.**

***Index Terms*—Deep model, feature extraction, max-margin, stochastic gradient Markov chain Monte Carlo (SG-MCMC).**

I. Introduction

IT IS well known that feature extraction is a significant and challenging problem in machine learning [1]. Although some optimization-based methods have recently enjoyed a great success in quiet a few applications [2]–[4], all of them are formulated as point estimate problems by optimizing some deterministic objective function, which may lead to some inconvenience [5]. Single point estimate, for instance, is often not sufficient in describing complex data distribution, and challenges usually occur when the objective function is nonconvex. On the contrast, probabilistic graphical models (PGM) are more powerful in describing data and modeling latent variables with uncertainty thanks to performing distribution estimate [6], which is beyond the capability of optimization-based methods. Many literatures [7]–[9] have advocated incorporation of uncertainty estimates during model training to help improve their models' robustness and performance. For example, deep neural networks (DNNs) often suffer from overfitting when the training set is small but the parameter space is large, which may be mitigated in Bayesian inference thanks to a distribution estimation over parameters [6]. In the past few decades, feature extraction based on PGMs with their Bayesian nature has exerted a tremendous fascination on researchers and shown favorable consequence [10].

An simple but quiet essential example is the factor analysis (FA) [11], where the data is decomposed into the product of two matrices, factor loading and latent feature, plus noise. To explore the spatial information, convolutional FA [12] was proposed, which can be seen as a convolution-based dictionary expansion of FA, since one must typically consider all possible shifts of canonical bases or filters. Nevertheless, they are described in an unsupervised manner without utilizing any label information which have relatively underdeveloped discrimination for supervised tasks. Recently, learning predictive latent features with supervised information has attracted a lot of attentions. The well known criterion in the field of classification is the max-margin, which maximizes the margin between different classes [13]. Since Polson and Scott proposed Bayesian SVM [14] via data augmentation technique, Bayesian max-margin supervised models have been developed in various predictive tasks, such as text document categorization [15] and multitask learning [16]. Pu *et al.* [17] compared Bayesian SVM with optimized-based classifiers, such as softmax function and cross-entropy, to illustrate again the advantage of Bayesian inference. Chen *et al.* [18] proposed the max-margin linear discriminant projection (MMLDP), jointly learning the discriminative subspace and classifier under a Bayesian framework. However, for data with complicated multimodal distribution, linear projection is unlikely to be sufficiently flexible to reveal underlying structures, so they also employed the kernel trick to implicitly perform a nonlinear discriminant projection in [18]. By aid of the Dirichlet process mixture [19], Zhang *et al.* [20] proposed the infinite max-margin FA (IMMFA) based on FA, which transforms the global nonlinear problem as multiple local linear problems. Along the other side, aiming at learning powerful nonlinear representations without the need to hand design features, DNNs are treated as nonlinear projections that are

Manuscript received June 5, 2017; revised October 27, 2017; accepted April 19, 2018. Date of publication May 21, 2018; date of current version April 17, 2019. This work was supported in part by the Thousand Young Talent Program of China, in part by NSFC under Grant 61771361, in part by the Fund for Foreign Scholars in University Research and Teaching Programs (the 111 Project) under Grant B18039, and in part by the National Science Fund for Distinguished Young Scholars of China under Grant 61525105. This paper was recommended by Associate Editor J. Jayadeva. *(Corresponding author: Bo Chen.)*

The authors are with the National Laboratory of Radar Signal Processing, Collaborate Innovation Center of Information Sensing and Understanding, Xidian University, Xi'an 710071, China (e-mail: zhanghao_xidian@163.com; bchen@mail.xidian.edu.cn; zhengjuewang@163.com; hwliu@xidian.edu.cn).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author.

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.2018.2831792

embedded into some traditional models to form more powerful ones [21]–[24]. As an example, deep linear discriminant analysis [21] casts LDA as an objective function for DNN through putting an LDA-layer on the top of the DNN and optimizing it end-to-end.

Based upon max-margin criterion, in this paper we propose a unified Bayesian max-margin discriminant projection (MMDP) framework, which is flexible in modeling the relationships between the latent representations and the observations, and is able to jointly learn the discriminant latent features as well as the max-margin classifier. In MMDP, the latent representations are assumed to be drawn from a normal distribution, whose sufficient statistics are functions with respect to the observations. Therefore, our framework is flexible enough to be implemented by different functions, namely different projections. When either a linear projection or a nonlinear one using kernel trick is chosen, the framework works out to be an existing, but shallow model in [18]. By combining MMDP with convolutional operation, we present a new model called max-margin convolutional discriminant projection (MMCDP). Motivated by the successful applications of deep models, we can stack the single-layer MMCDP to form a deep structure with layer-wise training like autoencoder (AE). Additionally, a visualization method similar to [25] and [26] is proposed to better understand what happens in such multilayer model. Inspired by the idea of DNN, two representative DNNs, mltilayer perceptron (MLP) [27] and convolutional neural network (CNN) [28], are embedded into our framework to construct a deep discriminant projection model. Different from the stacked MMCDP, it is trained in an end-to-end fashion such that the information interaction between two layers creates synergy and further boost the performance.

Both MMLDP and IMMFA are inferred via Gibbs sampling which is impractical for large-scale datasets. Shi and Zhu [5] presented online Bayesian passive-aggressive (BayesPA) to perform online Bayesian max-margin learning for streaming data. A popular alternatives utilizing stochastic gradient Markov chain Monte Carlo (SG-MCMC) to generate posterior samples are developed with great flexibility and scalability [29]–[32], and shown better consequence compared with optimization methods like stochastic gradient descend (SGD) or RMSprop [33]. Therefore, we design efficient online learning algorithms for our unified framework by means of SG-MCMC where the gradient for the framework used can be calculated efficiently by backpropagation (BP) [11].

The remainder of this paper is organized as follows. In Section II, we talk about some related works. Section III briefly describes the unified framework of Bayesian MMDP, and gives three special examples including one existing model and two new models. In Section IV, a scalable inference for the framework is presented. Moreover, we analysis the theoretical computational complexity of our method compared with Gibbs sampling at one iteration. In Section V, we conduct large-scale experiments on four image datasets (MNIST, CIFAR-10, STL-10, and SVHN) and one measured radar high-resolution range profile (HRRP) dataset to evaluate our models. Finally, the conclusion is drawn in Section VI.

II. Related Work

As a common and necessary step in many fields of machine learning, supervised feature extraction has been widely studied over the past few decades, because it can figure out which type of representation is relevant with the task at hand. LDA, seeking a projection subspace with class similarity constraint for separating data, is a representative supervised subspace analysis method. Considering some underlying assumptions, which are invariably not satisfied in practice [18], some variants have been presented in [34]–[37]. Although these models achieve better performances than conventional LDA in some problems, the learned feature subspaces enjoy no direct connection to the final classifier and all of them lack probabilistic interpretation.

Probabilistic graphic models aim to infer an entire distribution profile for the latent structure, leading to more powerful in data representation. Following this idea, Yu *et al.* [38] proposed supervised probabilistic PCA and Rai and Daumé [39] developed a supervised Bayesian probabilistic canonical component analysis. However, these models consider the classification problem as the imputation of missing values without taking any classification criterion into account. Along the other side, Gonen [40] introduced the Bayesian supervised dimension reduction model utilizing the conventional generalized linear model (GLMs) as the separation criterion to learn the discriminative subspace and a classifier simultaneously.

As we know, max-margin criterion has been successfully applied in classification [13] and structured output prediction models [41] in the last decade. The recent work in [14] provides an elegant data augmentation formulation for SVM with fully observed input data to give distribution estimations on classifier parameters, which results in analytical conditional distributions that are easy to sample from and flexible to combine with some PGMs. Based on this, a number of Bayesian max-margin supervised models for document analysis [5], [15] and multitask learning [16] were proposed by Zhu *et al.* Chen *et al.* [18] proposed the MMLDP and implemented it with a linear projection and a nonlinear one by kernel trick, both of which are under our unified framework. In order to deal with the global nonlinear problem, IMMFA [20] was proposed, by assembling a set of local regions, where Bayesian nonparametric prior is used to handle the model selection problem, e.g., the underlying number of local regions. In other words, it is flexible in capturing the local linear structures of data hidden in a global nonlinear problem.

Recently, treating DNN as a module to realize nonlinear projection in PGMs becomes popular. Kingma and Welling [23] and Rezende *et al.* [24] have coupled the approach of Bayesian model with DNN giving rise to powerful probabilistic models, called variational AE (VAE). Adding the max-margin criterion to the original VAE and defining the joint learning problem, Li *et al.* [42] proposed the max-margin deep generative model (mmDGM). Different from VAE and mmDGM embedding the DNN into the generative process and the variational distribution of latent representations, our proposed framework embeds it into the prior of latent representations. Besides, mmDGM

has to balance the ability of the model to generate and classify data, with a tradeoff parameter to be tuned, while our framework focus on the discriminant function without any requirement of data reconstruction.

In terms of the learning and inference algorithm, although optimization-based methods, such as SGD, get a good point estimation, the overfitting issue exists in many situations which is typically addressed by early stoping, weight decay, dropout, data augmentation, and so on. Bayesian inference is appealing due to their ability to avoid overfitting by capturing uncertainty during learning [43] where, instead of the point estimate, Bayesian model provides posterior distributions on the parameters. Gibbs sampling is a standard Bayesian inference method but has difficulties in dealing with big data. Though BayesPA [5] was given to perform online Bayesian max-margin learning for streaming data, its requirement of model being conjugate is not appropriate for nonconjugate ones, such as the third special example of our framework. Another direction for scalable Bayesian learning relies on the theory of SG-MCMC. Specifically, Welling and Teh [29] proposed stochastic gradient Langevin dynamics (SGLD), which was extended to some variants to make the learning faster and better [29]–[32]. Among them, stochastic gradient thermostats algorithms [stochastic gradient Nose–Hoover thermostat (SGNHT)] [31] introduced a proper thermostat into the algorithm to prevent the dynamics from drifting away the thermal equilibrium condition which should be satisfied in all dynamics-based sampling methods [44]; combining adaptive preconditioners with SGLD, preconditioned SGLD (pSGLD) was developed to overcome the problem in DNN that it often exhibit pathological curvature and saddle points [6]. Furthermore, it demonstrates that Bayesian model averaging mitigate overfitting through posterior sampling according to SG-MCMC and achieved better performance than SGD and RMSprop counterparts. Given the advantages of SGNHT and pSGLD, in this paper we extend them to our framework.

III. Max-Margin Discriminant Projection

In this section, we formulate the proposed Bayesian MMDP, followed by the discussion of several special examples to demonstrate the flexibility and effectiveness of MMDP.

A. Unified Framework

We consider binary classification with a labeled training dataset $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$, where the response variable y_n takes values from the output space $\mathbf{Y} = \{-1, +1\}$. MMDP consists of two parts: 1) a generative process of the latent feature representation and 2) an suitable classifier for considering the supervising information $\boldsymbol{y} = \{y_n\}_{n=1}^N$, which will be introduced each of them in turn.

1) Generative Process of Latent Feature: To build an flexible relationships between the observation and the feature, we model it as latent variable drawn from normal distribution, whose sufficient statistics are functions with respect to the observations. In other words, the generative process of the latent feature representation can be formally expressed as

$$z_n|\boldsymbol{x}_n \sim \mathcal{N}\Big(\boldsymbol{\mu}_{\boldsymbol{\Theta}}(\boldsymbol{x}_n),\ \operatorname{diag}\Big\{\boldsymbol{\sigma}^2_{\boldsymbol{\Theta}}(\boldsymbol{x}_n)\Big\}\Big) \tag{1}$$

where $\mathcal{N}(\boldsymbol{\mu}, \operatorname{diag}\{\boldsymbol{\sigma}^2\})$ denotes a Gaussian distribution with mean vector $\boldsymbol{\mu} \in \mathbb{R}^{K\times 1}$ and diagonal covariance matrix with $\boldsymbol{\sigma}^2$ being its diagonal elements, and both $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ are any suitable functions with parameters $\boldsymbol{\Theta}$ with respect to $\boldsymbol{x}_n$. In our model, we hope to build a conditional prior so as to have a good posterior representation of latent variable, which will be analyzed detailed through the posterior in the following. Therefore, compared with traditional Bayesian supervised projection models [18], [40], the projection relationship in our framework is more flexible because a suitable one can be embedded according to the type of data and computational complexity. Compared with DNN, we add a distribution to the feature to give a more powerful expression of the observations. Considering the computational complexity of the model, we directly use unit matrix as the covariance matrix, in the following examples. Certainly, it is worth noting that the model can also be inferred via our proposed scalable methods if the original one in (1) is used.

2) Classifier: Given the supervised information, one proper criterion should be chosen to affect and promote the discrimination of the features. Different with [40] using a conventional GLM as the separation criterion and traditional softmax regression in DNN, we prefer using the max-margin criterion to link the label with the feature since it has been proven to be an outstanding choice in learning discriminative models [13], [18], [42] because of a more principled classification interface. With Bayesian max-margin classifier [14], we extend the model (1) to the classification problem by introducing the pseudo-likelihood ϕ expressed as

$$\begin{aligned}\phi(y_n|\boldsymbol{w}) &= \exp\big\{-2\max\big[1 - y_n\big(\boldsymbol{z}_n^T\boldsymbol{w}\big), 0\big]\big\} \\ &= \int_0^\infty \frac{1}{\sqrt{2\pi\lambda_n}} \exp\left(-\frac{\big[1+\lambda_n - y_n\big(\boldsymbol{z}_n^T\boldsymbol{w}\big)\big]^2}{2\lambda_n}\right) d\lambda_n\end{aligned} \tag{2}$$

whose details of the proof is referred to [14]. For simplicity, we use $\boldsymbol{w}$ to represent the concatenation of coefficient $\boldsymbol{w}$ and bias b, each $\boldsymbol{z}_n$ represents $[\boldsymbol{z}_n; 1]$ and λ_n is an augmentated variable. Therefore, the pseudo-posterior of MMDP can be expressed as

$$p(\mathbf{Z}, \boldsymbol{\Theta}, \boldsymbol{w}, \lambda|\boldsymbol{y}, \mathbf{X}) \propto \prod_{n=1}^N p(\boldsymbol{z}_n|\boldsymbol{x}_n, \boldsymbol{\Theta})\phi(y_n, \lambda_n|\boldsymbol{w}, \boldsymbol{z}_n)p(\boldsymbol{w})p(\boldsymbol{\Theta}) \tag{3}$$

where $p(\boldsymbol{w})$ and $p(\boldsymbol{\Theta})$ are the prior of $\boldsymbol{w}$ and $\boldsymbol{\Theta}$. We find in experiments that a simple prior, commonly set $p(\boldsymbol{w}) = \mathcal{N}(0, \sigma_{\boldsymbol{w}}\mathbf{I})$ and $p(\boldsymbol{\Theta}) = \mathcal{N}(0, \alpha_{\boldsymbol{\Theta}}\mathbf{I})$ like [6], is sufficient for MMDP. As a result, the conditional posterior of $\boldsymbol{z}_n$ for training instance can be expressed as $p(z_{nk}|-) = \mathcal{N}(z_{nk}; \mu_{z_{nk}}, \Sigma_{z_{nk}})$ with

$$\begin{aligned}\sigma_{z_{nk}} &= \left([\boldsymbol{\sigma}(\boldsymbol{x}_n)]_k + \frac{w_k^2}{\lambda_n}\right)^{-1} \\ \mu_{z_{nk}} &= \sigma_{z_{nk}}\left[\left(1 + \frac{\xi_n^{-k}}{\lambda_n}\right) y_n w_k + [\boldsymbol{\mu}(\boldsymbol{x}_n)]_k\right]\end{aligned} \tag{4}$$

where $\xi_n^{-k} = 1 - y_n \sum_{h=1,h\neq k}^{K} w_h z_{nh}$. Apparently, beside the true label, both of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ have an influence on the posterior of $\mathbf{z}_n$, which demonstrates the benefits from the flexible and powerful functions in (1) building the relationship between observations and latent variables.

It is clear from the discussion above that different $p(z_n|\boldsymbol{x}_n, \boldsymbol{\Theta})$ results in different specific examples with other items invariable. For convenience of subsequent description it is desirable to give the following definition:

$$G(\mathbf{y}, \lambda, \boldsymbol{w}, \mathbf{Z}, \boldsymbol{\Theta}) \triangleq \prod_{n=1}^{N} \phi(y_n, \lambda_n|\boldsymbol{w}, z_n) p(\boldsymbol{w}) p(\boldsymbol{\Theta}) \tag{5}$$

with which MMDP can be expressed as

$$p(\mathbf{Z}, \boldsymbol{\Theta}, \boldsymbol{w}, \lambda|\mathbf{y}, \mathbf{X}) \propto \prod_{n=1}^{N} p(z_n|\boldsymbol{x}_n, \boldsymbol{\Theta}) G(\mathbf{y}, \lambda, \boldsymbol{w}, \mathbf{Z}, \boldsymbol{\Theta}). \tag{6}$$

In the following, some special examples of our unified framework are discussed to demonstrate its flexibility including the existing and our new proposed ones.

B. Linear Example $\boldsymbol{\mu}(\boldsymbol{x}_n) = \boldsymbol{A}^T \boldsymbol{x}_n$

As a fundamental projection function, linear projection is adopted in an ocean of models. Given the data $\boldsymbol{x}_n \in \mathbb{R}^{P\times 1}$ and the linear projection matrix $\mathbf{A} \in \mathbb{R}^{P\times K}$, we build the generative process of latent feature representation in a linear manner with the function $\boldsymbol{\mu}(\boldsymbol{x}_n) = \mathbf{A}^T \boldsymbol{x}_n$

$$z_n \sim \mathcal{N}\left(\mathbf{A}^T \boldsymbol{x}_n, \mathbf{I}\right). \tag{7}$$

Correspondingly, we get the pseudo-posterior

$$p(\mathbf{Z}, \mathbf{A}, \boldsymbol{w}, \lambda|\mathbf{y}, \mathbf{X}) \propto \prod_{n=1}^{N} p(z_n|\mathbf{A}, \boldsymbol{x}_n) G(\mathbf{y}, \lambda, \boldsymbol{w}, \mathbf{Z}, \mathbf{A}) \tag{8}$$

which is just the MMLDP [18] with Gibbs sampling shown in Appendix A, in the supplementary material. However, for data with complicated multimodal distribution, linear projection is not flexible enough to reveal useful structures. Therefore, we can kernelize the above linear feature extractor for more flexiblity as

$$z_n \sim \mathcal{N}\left(\mathbf{A}^T \Phi(\boldsymbol{x}_n), \mathbf{I}\right) \tag{9}$$

where $\boldsymbol{\mu}(\boldsymbol{x}_n) = \mathbf{A}^T \Phi(\boldsymbol{x}_n)$ and Φ is a nonlinear mapping. In practice, we build the model using kernel trick which is same with kernel MMDP (KMMDP) [18]. Since we utilize some predefined basis vectors to approximate $\mathbf{A}$, it should be emphasized that although the kernel function is employed to do nonlinearly transformation, KMMDP is also a linear projection in Hilbert space. What is more, we have to choose appropriate kernel function and basis vectors handcrafted, which is a arduous problem so far though some approximation methods were proposed in [45]. In addition, both of linear or kernel projections vectorize the original multidimensional data without considering any underlying correlation between instance features, such as spatial and temporal information. In the following Section III-C, we will present a convolutional extension of linear example called MMCDP with a deep architecture via layer-wise training.

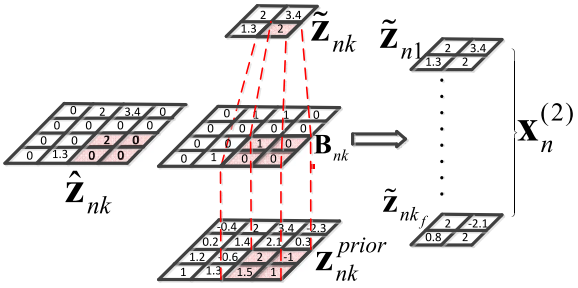


Fig. 1. Explanation of Bayesian approximate max-pooling and the output from layer one, for analysis at layer two (a similar procedure is implemented between any two layers).

C. Convolutional Example $\boldsymbol{\mu}(\boldsymbol{x}_n) = \boldsymbol{A} * \boldsymbol{x}_n$

With an assumption about the nature of image, namely, stationarity of statistics and locality of pixel dependencies [3], MMCDP utilize the spatial information of image since it typically consider all feasible shifts of canonical bases or filters. At first, we introduce its single-layer structure.

1) Single Layer: Suppose the data $\boldsymbol{x}_n \in \mathbb{R}^{n_y \times n_x \times K_c}$ where K_c is the number of channels, we aim to find some filters $\mathbf{A} \in \mathbb{R}^{a_y \times a_x \times K_c \times K_f}$ such that the mean of latent representation z_n is obtained via convolution, that is $\boldsymbol{\mu}(\boldsymbol{x}_n) = \mathbf{A} * \boldsymbol{x}_n$, where $a_x \ll n_x$, $a_y \ll n_y$, and K_f is the number of filters. With this purpose we have

$$z_{nk} \sim \mathcal{N}(\boldsymbol{a}_k * \boldsymbol{x}_n, \mathbf{I}), \quad k = 1, \ldots, K_f \tag{10}$$

where $z_{nk} \in \mathbb{R}^{(n_y - a_y + 1)(n_x - a_x + 1)}$ called feature map and then they are stacked and transformed to a vector $z_n \in \mathbb{R}^{K_f (n_y - a_y + 1)(n_x - a_x + 1) \times 1}$ as the feature of $\boldsymbol{x}_n$. In many cases, the length of z_n is long which may result in dimensional disaster if we regard it as the input of classifier directly, so a Bayesian pooling step is applied to each z_{nk}. Concretely, we impose in each pooling block (the red region in Fig. 1) that

$$\mathbf{B}_{nk} \sim \text{Mult}(1, \rho) \tag{11}$$

where Mult($\bullet$) denotes multinomial distribution. As a result, we have the generative process of latent feature with pooling matrix $\mathbf{B}$

$$\hat{z}_{nk} \sim \mathcal{N}((\boldsymbol{a}_k * \boldsymbol{x}_n) \odot \mathbf{B}_{nk}, \mathbf{I}) \tag{12}$$

where the size of $\hat{z}_{nk}$ and $\mathbf{B}_{nk}$ are the same as that of z_{nk}. According to [46], $\mathbf{B}_{nk}$ should be sampled element-by-element from its conditional posterior, that is given the tth pooling block, we calculate the probability with only ith element being ones

$$p\left(B_{nk,t}^{i} = 1|-\right) \propto \mathcal{N}\left(\hat{z}_{nk}; \quad (\boldsymbol{a}_k * \boldsymbol{x}_n) \odot \mathbf{B}_{nk}, \mathbf{I}\right) \text{Mult}(\mathbf{B}_{nk}; \rho) \tag{13}$$

which is done for all i and t. From (13), it is noted that the conditional posterior of $\mathbf{B}_{nk}$ depends much on $\boldsymbol{a}_k * \boldsymbol{x}_n$. In order to decrease the computational complexity, we introduce an approximate method to get $\mathbf{B}_{nk}$ using prior-maxpooling. As shown in Fig. 1, first, each matrix z_{nk}^{prior}, the prior of z_{nk} obtained by $z_k * \boldsymbol{x}_n$, is divided into a contiguous set of blocks,

with each such block of size $p_y \times p_x$, and so does $\mathbf{B}_{nk}$; second, the elements of one block in $\mathbf{B}_{nk}$ are all zeros except one element being one, and its location is the same as that of the largest-magnitude element in corresponding block in $\boldsymbol{z}_{nk}^{\text{prior}}$. Through this simple but efficient process, we could get the pooling matrix quickly and this strategy can be used for many types of pooling like average pooling and stochastic pooling.

We plug (12) into (3) to get the pseudo-posterior

$$p(\mathbf{Z}, \mathbf{A}, \boldsymbol{w}, \lambda|\mathbf{X}, \boldsymbol{y}) \propto \prod_{n=1}^{N}\prod_{k=1}^{K_f} p\big(\hat{\boldsymbol{z}}_{nk}|\boldsymbol{a}_k, \boldsymbol{x}_n\big) G(\boldsymbol{y}, \lambda, \boldsymbol{w}, \mathbf{Z}, \boldsymbol{\Theta}) \tag{14}$$

where ϕ in G is denoted as $\phi(y_n, \lambda_n|\boldsymbol{w}, \bar{\boldsymbol{z}}_n)$ with $\bar{\boldsymbol{z}}_n$ being the vectorial form of $\{\tilde{\boldsymbol{z}}_{nk}\}_{k=1}^{K_f}$ (see Fig. 1). In other words, only those elements selected by prior-maxpooling are regarded as the input of classifier. As stated above, K is defined as the dimension of latent variable $\bar{\boldsymbol{z}}_n$, thus we have

$$K = \left(\frac{n_y - a_y + 1}{p_y}\right)\left(\frac{n_x - a_x + 1}{p_x}\right) K_f \tag{15}$$

much smaller than $\boldsymbol{z}_n$. Under this setting, the conditional posteriors can be derived analytically. Here, we only show them of $\boldsymbol{a}_k$ and $\boldsymbol{z}_k$ with others similar to that of MMLDP.

1) Sample each filter $\boldsymbol{a}_k$ from $p(\bar{\boldsymbol{a}}_k|-) = \mathcal{N}(\bar{\boldsymbol{a}}_k; \boldsymbol{\mu}_{\bar{\boldsymbol{a}}_k}, \boldsymbol{\sigma}_{\bar{\boldsymbol{a}}_k})$ with

$$\begin{aligned}\boldsymbol{\sigma}_{\bar{\boldsymbol{a}}_k} &= \left(\sum_{n=1}^{N}\sum_{m=1}^{M}\big(\bar{\boldsymbol{x}}_n^m \bar{\boldsymbol{x}}_n^{mT}\big) b_{nk}^m + \alpha_k \mathbf{I}\right)^{-1} \\ \boldsymbol{\mu}_{\bar{\boldsymbol{a}}_k} &= \boldsymbol{\sigma}_{\bar{\boldsymbol{a}}_k} \times \text{Vec}\left(\sum_{n=1}^{N} \hat{\boldsymbol{z}}_{nk} * \boldsymbol{x}_n\right)\end{aligned} \tag{16}$$

where Vec$(\bullet)$ means transforming a matrix to a vector, and $\bar{\boldsymbol{x}}_n^m = \text{Vec}(\boldsymbol{x}_n^m)$, $\bar{\boldsymbol{a}}_k = \text{Vec}(\boldsymbol{a}_k)$. $\{\boldsymbol{x}_n^m\}_{m=1}^{M}$ are all possible patches of $\boldsymbol{x}_n$ used in convolution, and b_{nk}^m is the corresponding element in $\mathbf{B}_{nk}$. It is interesting to note that only the selective patches are used to sample the filter $\boldsymbol{a}_k$, which is an significant reason why the learned filters are well-structured.

2) Sample the selected element in $\hat{\boldsymbol{z}}_{nk}$ from $p(\hat{z}_{nk}^i|-) = \mathcal{N}(\hat{z}_{nk}^i; \mu_{\hat{z}_{nk}^i}, \sigma_{\hat{z}_{nk}^i})$ with

$$\begin{aligned}\sigma_{\hat{z}_{nk}^i} &= \left(1 + \frac{w_k^2}{\lambda_n}\right)^{-1} \\ \mu_{\hat{z}_{nk}^i} &= \sigma_{\hat{z}_{nk}^i}\left[\left(1 + \frac{\xi_n^{-k}}{\lambda_n}\right) y_n w_k + \bar{\boldsymbol{a}}_k^T \bar{\boldsymbol{x}}_n^i\right]\end{aligned} \tag{17}$$

where $\xi_n^{-k} = 1 - y_n \sum_{h=1, h\neq k}^{K} w_h \bar{z}_{nh}$ and those which are not selected are sampled from their prior. In other words, only the sampling process of selected elements in $\hat{\boldsymbol{z}}_{nk}$ is affected by the true label.

In the next part, single-layer MMCDP will be stacked to build a multilayer feature learning model with training layerwise like [12].

2) Multilayer Architecture: After training the first layer and pooling operation, we have $K_f^{(1)}$ feature maps (assuming the number of filters at layer one is $K_f^{(1)}$), which are stacked to constitute a tensor denoted as $\boldsymbol{x}_n^{(2)}$ (see Fig. 1). Associated with the nth image, $\boldsymbol{x}_n^{(2)}$ is treated as the input image at the next layer of the model, which is performed for all the N images. Model fitting at layer two is done analogous to that in (14). Note that because of the pooling step, the size of input decreases as one moves to higher levels. As a result, the basic computational complexity decreases with the increase of layers. This process may be continued layer by layer, but we consider up to three layers in the experiments. Although no activation function is used between layers, max-pooling in our model plays a nonlinear role. Also just for this reason, a simple way to visualize the filters will be proposed next.

3) Model Visualization: In addition to performing classification based on the features after pooling, it is of interest to examine the physical meaning of filters at different layers. In our model, we can show filters directly for $l = 1$, but it is meaningless for $l > 1$, the representation of which should be examined at image plane.

Specifically, given filter $\boldsymbol{a} \in \mathbb{R}^{a_y^{(l)} \times a_x^{(l)} \times K_f^{(l)}}$ at layer l, first we select the top N' data from all training data with highest response and get the value $z_{\max,n}^{(l)} \in \mathbb{R}^1, n = 1, \ldots, N'$. Second corresponding to $z_{\max,n}^{(l)}$ we can find a region $\mathbf{Z}_{\text{block},n}^{(l)} \in \mathbb{R}^{a_y^{(l)} \times a_x^{(l)} \times K_f^{(l)}}$ whose each value is pooled from a block in the output of layer $l-1$, that is the region $\mathbf{Z}_{\text{block},n}^{(l-1)}$. Like this, we go back to the original image and get a region $\mathbf{X}_n^{\text{region}}$ with the visualization of filter $\boldsymbol{a}$ is represented as

$$\sum_{n=1}^{N'} \mathbf{X}_n^{\text{region}} \bullet \frac{z_{\max,n}^{(l)}}{z_{\max,\text{sum}}^{(l)}} \tag{18}$$

where $z_{\max,\text{sum}}^{(l)} = \sum_{n=1}^{N'} z_{\max,n}^{(l)}$ represents the sum of maximum response value for these N' images. For a clearer illustration, there is a figure shown in the supplementary material.

Although this process is a heuristic one, we can analyze how it works according to the process. For a filter at layer l we find the region $\mathbf{X}_n^{\text{region}}$ having a maximum response. If the filter is well-learned, the visualized result is structured because the region $\mathbf{X}_n^{\text{region}}$ is very similar for all N' images. Otherwise, the result is smooth because $\mathbf{X}_n^{\text{region}}$ is different for these images and weighted summation results in smoothness.

D. Deep Network Example $\boldsymbol{\mu}(\boldsymbol{x}_n) = f^{DN}(\boldsymbol{x}_n)$

Although single-layer MMCDP can be stacked to build a deep model in order to improve the representation capability, but its layer-wise training strategy is not expected because it is hard to interact information between two layers. As DNNs are able to extract abstract features by explicitly designed nonlinear and can be trained end-to-end, in this section, we consider learning flexible nonlinear deep representations via DNN, building a deep Bayesian MMDP model directly.

It is generally known that among many types of DNN, MLP, and CNN are widely used, which can be seen as a general

function building a mapping from one space to another. With the help of this view, (1) can be set as

$$z_n \sim \mathcal{N}\big(f_{\mathbf{\Theta}}^{\mathrm{DNN}}(\boldsymbol{x}_n), \mathbf{I}\big) \tag{19}$$

where the mean function f is a deterministic deep network with $\mathbf{\Theta}$ being the network parameters to be learned. Different with traditional MLP and CNN models accomplishing supervised tasks directly, both of them are functional representations to approximate the sufficient statistics of the distribution. In the other words, we do not combine DNN with max-margin criterion directly but build a Bayesian projection model with flexible representative ability.

By substituting (19) into (6), the posterior of the model can be expressed as

$$p(\mathbf{Z}, \boldsymbol{w}, \lambda | y, \mathbf{X}) \propto \prod_{n=1}^{N} p(z_n|\boldsymbol{x}_n, \mathbf{\Theta}) G(\boldsymbol{y}, \lambda, \boldsymbol{w}, \mathbf{Z}, \mathbf{\Theta}). \tag{20}$$

It is worth of noticing that comparing (8), (14), and (20), we can find that MMLDP and MMCDP can be reformulated as the special examples of MMDP-MLP and MMDP-CNN without nonlinear activation function, respectively. In other words, MMDP-MLP and MMDP-CNN have more complex functions and the more powerful expressive capability to approximate the parameters of the distortions of the latent representations, $\mathbf{Z}$. Moreover, unlike multilayer MMCDP, this example is an end-to-end fashion, that is to say that all the parameters in the functions and the classifier can be trained together instead of layer-wise training, such that the information can be interacted easily between each two layers.

For local parameter z_n and λ_n, it can be sampled via Gibbs sampling like MMLDP. However, for parameters $\mathbf{\Theta}$, Gibbs sampling is out of work because its conditional posteriors cannot be derived analytically, which will be solved in the next section.

IV. Scalable Posterior Inference

Although MMLDP and MMCDP can be learned via Gibbs sampling, it requires a full pass through the entire dataset at each iteration, which will be prohibited to be applied directly by emerging large-scale corpora. Moreover, the parameters in f of the deep network example cannot be learned via Gibbs sampling. In this section, we focus on the method for scaling up inference by stochastic algorithm based on dynamical sampling method, where mini-batch instead of the whole dataset are utilized in each iteration of the algorithms.

Recap the posterior of MMDP shown in (3), in the learning phase we are interested in learning the global parameters $\mathbf{\Psi}_g = \{\mathbf{\Theta}, \boldsymbol{w}\}$. For dynamics-based sampling method, it works in an extended space $\Gamma = (\mathbf{\Psi}_g, \boldsymbol{p})$, where $\mathbf{\Psi}_g$ and $\boldsymbol{p}$ simulate the positions and the momenta of particles having unit constant mass in the target system (e.g., a Bayesian model). If we define the potential energy of the system as

$$\tilde{U}(\boldsymbol{w}) = -\frac{N}{\tilde{N}} \sum_{n=1}^{\tilde{N}} \mathbb{E}\big[\log \phi(y_n, \lambda_n | \boldsymbol{w}, \boldsymbol{z}_n)\big] - \log p(\boldsymbol{w}) \tag{21}$$

Algorithm 1 Scalable Inference for MMDP

Input: Data $\{\boldsymbol{x}_n, y_n\}_{n=1}^{N}$, Parameters $\varepsilon_{\mathbf{\Theta}}, \varepsilon_{\boldsymbol{w}}$, $C_{\mathbf{\Theta}}, C_{\boldsymbol{w}}$
Initialize: $\mathbf{\Theta}$, $\boldsymbol{w}$, $\boldsymbol{p}_{\mathbf{\Theta}}^0$, $\boldsymbol{p}_{\boldsymbol{w}}^0$ randomly,
for $t = 1, 2, \ldots$ **do**
  Sample a minibatch $\{\boldsymbol{x}_n, y_n\}_{n=1}^{\tilde{N}}$ from the dataset;
  Collect samples $\{z_n^{(j)}, \lambda_n^{(j)}\}_{n=1,j=1}^{\tilde{N},J}$ using Gibbs sampling;
  Discard the first β burn-in samples ($\beta < J$);
  Use the rest $J - \beta$ samples to calculate the stochastic force of $\mathbf{\Theta}$ and $\boldsymbol{w}$ according to (25) and (24);
  $\boldsymbol{p}_{\mathbf{\Theta}}^{t,0} = \boldsymbol{p}_{\mathbf{\Theta}}^{t-1}$, $\boldsymbol{p}_{\boldsymbol{w}}^{t,0} = \boldsymbol{p}_{\boldsymbol{w}}^{t-1}$,
  for $i = 1$ **to** I **do**
    for $\mathbf{\Psi}_g = \mathbf{\Theta}, \boldsymbol{w}$ **do**
      $\boldsymbol{p}_{\mathbf{\Psi}_g}^{t,i} = \boldsymbol{p}_{\mathbf{\Psi}_g}^{t,i-1} - C_{\mathbf{\Psi}_g} \boldsymbol{p}_{\mathbf{\Psi}_g}^{t,i-1} \varepsilon_{\mathbf{\Psi}_g} + \tilde{F}_{\mathbf{\Psi}_g} \varepsilon_{\mathbf{\Psi}_g} + \sqrt{2C_{\mathbf{\Psi}_g}} \mathcal{N}\big(0, \varepsilon_{\mathbf{\Psi}_g}\big)$,
      $\mathbf{\Psi}_g = \mathbf{\Psi}_g + \boldsymbol{p}_{\mathbf{\Psi}_g}^{t,i} \varepsilon_{\mathbf{\Psi}_g}$,
    end for
  end for
  $\boldsymbol{p}_{\mathbf{\Theta}}^{t} = \boldsymbol{p}_{\mathbf{\Theta}}^{t,I}$, $\boldsymbol{p}_{\boldsymbol{w}}^{t} = \boldsymbol{p}_{\boldsymbol{w}}^{t,I}$
end for

$$\tilde{U}(\mathbf{\Theta}) = -\frac{N}{\tilde{N}} \sum_{n=1}^{\tilde{N}} \mathbb{E}\big[\log p(z_n|\boldsymbol{x}_n, \mathbf{\Theta})\big] - \log p(\mathbf{\Theta}) \tag{22}$$

where $\{\boldsymbol{x}_n\}_{n=1}^{\tilde{N}}$ represents a random subset of $\{\boldsymbol{x}_n\}_{n=1}^{N}$ and $\tilde{N} \ll N$, the Langevin dynamics with diffusion factor C can be described by the following stochastic differential equations:

$$d\mathbf{\Psi}_g = \boldsymbol{p} dt, \quad d\boldsymbol{p} = \tilde{F}\big(\mathbf{\Psi}_g\big) dt - C\boldsymbol{p} dt + \sqrt{2C} d\mathbf{W} \tag{23}$$

where $\mathbf{W}$ is n dependent Wiener process, $d\mathbf{W}$ can be informally written as $\mathcal{N}(0, dt)$ [30], $\tilde{F}(\mathbf{\Psi}_g) = -\nabla \tilde{U}(\mathbf{\Psi}_g)$ is the stochastic force on the system calculated as follows:

$$\tilde{F}(\boldsymbol{w}) = -\frac{N}{\tilde{N}} \sum_{n=1}^{\tilde{N}} \mathbb{E}\left[\frac{z_n z_n^T}{\lambda_n} \boldsymbol{w} - \frac{1 + \lambda_n}{\lambda_n} y_n z_n^T\right] - \sigma_{\boldsymbol{w}} \boldsymbol{w} \tag{24}$$

$$\tilde{F}(\mathbf{\Theta}) = -\frac{N}{\tilde{N}} \sum_{n=1}^{\tilde{N}} \mathbb{E}\left[\frac{\partial \log p(z_n|\boldsymbol{x}_n, \mathbf{\Theta})}{\partial \mathbf{\Theta}}\right] - \sigma_{\mathbf{\Theta}} \mathbf{\Theta} \tag{25}$$

where the expectation is taken over posteriors. Although no closed-form integrations can be obtained for the stochastic force, we thus use Monte Carlo integration to approximate the quantity. Specifically, given $\{\boldsymbol{w}, \mathbf{\Theta}\}$ we collect samples of the local variable $\{\lambda_n, z_n\}_{n=1}^{\tilde{N}}$ by running a few Gibbs sampling to approximate the intractable integrations. Therefore, as long as (23) is replaced by discrete equations with a proper step size ε and I leapfrog steps [47], we get the general scalable inference for MMDP shown in Algorithm 1 which is divided into two parts at each iteration, very similar to EM algorithm. First part (E step), we use the last network parameters to sample the local variables to approximat the expectation in gradient. Seconde part (M step), we update the global variables to achieve better projection function and classifier. In addition, different from conventional DNN updated by standard SGD to achieve a point estimation, our algorithm is equivalent to adding uncertainty to the parameters of DNN.

TABLE I
COMPUTATIONAL COMPLEXITIES OF GMMLDP, OMMLDP, GMMCDP, AND OMMCDP AT EACH ITERATION. SUPPOSE THE SIZE OF ONE KERNEL IS $a_x \times a_y \times K_c$ AND LET $Q = a_x a_y K_c$

	$\mathbf{A}$	$\mathbf{Z}$	$\boldsymbol{w}$	λ
gMMLDP	$O\left(KP^3 + NPK\right)$	$O\left(NPK\right)$	$O\left(K^3 + NK^2\right)$	$O\left(NK\right)$
oMMLDP	$O\left(\tilde{N}P^2 + \tilde{N}PK + 2IPK\right)$	$O\left(\tilde{N}JPK\right)$	$O\left(\tilde{N}K^2 + 2IK\right)$	$O\left(\tilde{N}JK\right)$
gMMCDP	$O\left(K_f Q^3 + NM\left(Q^2 + K_f Q\right)\right)$	$O\left(NQK\right)$	$O\left(K^3 + NK^2\right)$	$O\left(NK\right)$
oMMCDP	$O\left(\tilde{N}MQ^2 + \left(\tilde{N}M + 2I\right)K_f Q\right)$	$O\left(\tilde{N}JQK\right)$	$O\left(\tilde{N}K^2 + 2IK\right)$	$O\left(\tilde{N}JK\right)$

For different specific examples of MMDP framework, however, the general method stated above may not the best choice. In the next section, we will introduce some better choices for the specific examples.

A. Stochastic Gradient Thermostats

Because of the utilization of stochastic force in (23), the dynamics may drift away the equilibrium condition which should be satisfied in all dynamics-based sampling methods [44]. Recently proposed SGNHT introduced a proper thermostat, that is the additional variable ξ, which adaptively controls the mean kinetic energy as follows:

$$\begin{aligned} d\boldsymbol{\Psi}_g &= \boldsymbol{p}dt, \quad d\boldsymbol{p} = \tilde{F}\left(\boldsymbol{\Psi}_g\right)dt - \xi \boldsymbol{p}dt + \sqrt{2C}\mathcal{N}(0, dt) \\ d\xi &= \left(\frac{1}{n}\boldsymbol{p}^T\boldsymbol{p} - 1\right)dt. \end{aligned} \tag{26}$$

Compared with (23), ξ plays a role in controlling the mean kinetic energy adaptively instead of a constant one C. In experiments we find that it brings about stronger robustness for small batch size. With a little change of Algorithm 1 we get the SGNHT method for MMDP framework given in the Appendix B, in the supplementary material.

B. Preconditioned Stochastic Gradient Langevin Dynamics for Deep Network Examples

Although stronger robustness for small batch size is obtained by SGNHT, it plays a little worse in DNN example of the framework, because gradients in DNN often suffer from the vanishing/exploding problem [48]. What is more, a fixed stepsize ε is used which makes it difficult to choose a proper one for different layers. Li *et al.* [6] proposed pSGLD to solve this problem using the second-order information $G(\boldsymbol{\Psi}_g)$

$$d\boldsymbol{\Psi}_g = \frac{\varepsilon}{2}\left[G\left(\boldsymbol{\Psi}_g\right)\tilde{F}\left(\boldsymbol{\Psi}_g\right)\right] + G^{\frac{1}{2}}\left(\boldsymbol{\Psi}_g\right)\mathcal{N}(0, \varepsilon \mathbf{I}). \tag{27}$$

$G(\boldsymbol{\Psi}_g)$ is updated sequentially via only the current gradient and only estimated a diagonal one as follows:

$$V\left(\boldsymbol{\Psi}_g^{t+1}\right) = \delta V\left(\boldsymbol{\Psi}_g^{t}\right) + (1-\delta)\bar{g}\left(\boldsymbol{\Psi}_g^{t}; D^t\right) \odot \bar{g}\left(\boldsymbol{\Psi}_g^{t}; D^t\right) \tag{28}$$

$$G\left(\boldsymbol{\Psi}_g^{t+1}\right) = \operatorname{diag}\left(\mathbf{1} \oslash \left(\gamma \mathbf{1} + \sqrt{V\left(\boldsymbol{\Psi}_g^{t+1}\right)}\right)\right) \tag{29}$$

where $\delta \in [0, 1]$ balance the weight of historical and current gradient, and γ controls the extremes of the curvature in preconditioner. Operators $\odot$ and $\oslash$ represent element-wise matrix product and division, respectively. The mean gradient of $\boldsymbol{\Theta}$ in DNN example is

$$\bar{g}(\boldsymbol{\Theta}; D) = \frac{1}{\tilde{N}}\sum_{n=1}^{\tilde{N}} -\frac{1}{2}\frac{\partial \|\boldsymbol{z}_n - f(\boldsymbol{x}_n)\|_2^2}{\partial \boldsymbol{\Theta}} \tag{30}$$

which can be quickly and efficiently got using BP. This is the most crucial point of the low computational complexity of this model. With stated above, the scalable inference for DNN example is shown in the Appendix C, in the supplementary material.

C. Prediction

Different with traditional optimization-based model having a point estimation, our model performs distribution estimate based on SG-MCMC samples. Therefore, after the burn-in stage, we can average over all of the collected samples to predict the label of a new data x^* as follows:

$$y^* = \operatorname{sign}\left(\frac{1}{T - T_{\text{burn-in}}}\sum_{t=T_{\text{burn-in}}+1}^{T} \boldsymbol{w}_t^T \boldsymbol{z}_t^*\right) \tag{31}$$

where $\{\boldsymbol{z}_t^* = \boldsymbol{\mu}_{\boldsymbol{\Theta}_t}(\boldsymbol{x}^*)\}_{t=T_{\text{burn-in}}+1}^{T}$ and $\boldsymbol{\Theta}_t$ is drawn from its corresponding posterior base on $T - T_{\text{burn-in}}$ collected samples. Thus, instead of integrating the latent variables, we average the final outputs from the max-margin classifier, which is able to boost the performance in the experiments. Although the problem we discussed is for binary classification, there have been several strategies to realize multiclass classification via binary one. In this paper, we choose one-vs-all strategy, since its effectiveness has been analyzed and proved theoretically and experimentally.

D. Computational Complexity

In Table I, we list the per-iteration complexity of each parameter in MMLDP and MMCDP using Gibbs sampling (denoted as gMMLDP and gMMCDP) and scalable inference (denoted as oMMLDP and oMMCDP), respectively. For gMMLDP to sample the projection matrix $\mathbf{A}$, although we can calculate $\sum_{n=1}^{N}\boldsymbol{x}_n\boldsymbol{x}_n^T$ before iteration, we need inverse a $P \times P$ matrix, making it not practical for high-dimensional data. It is not dominant to infer other parameters in gMMLDP since $K \ll P$. For oMMLDP, although we need calculate $\sum_{n=1}^{\tilde{N}}\boldsymbol{x}_n\boldsymbol{x}_n^T$ to sample $\mathbf{A}$, a small batchsize $\tilde{N}$ is needed. In addition, we do not need to inverse a $P \times P$ matrix, which is suitable for scalable inference. What is more, it takes a little time to perform I leapfrog steps, because the computational complexity

TABLE II
ILLUSTRATION OF THE DATASETS

Dataset	Size	Train Num.	Test Num.	Preprocessing
MNIST	28×28	60000	10000	-
CIFAR-10	$32 \times 32 \times 3$	50000	10000	GCN, ZCA
STL-10	$96 \times 96 \times 3$	5000	8000	GCN, ZCA
SVHN	$32 \times 32 \times 3$	604388	26032	LCN

of every leapfrog step is low and I is small in a multitude of cases. Since we use J samples to calculate the expectation, the number of N in gMMLDP is replaced by $\tilde{N}J$ in oMMLDP.

For gMMCDP when sample $\mathbf{A}$, we cannot calculate $\sum_{n=1}^{N}\sum_{m=1}^{M} \boldsymbol{x}_n^m (\boldsymbol{x}_n^m)^T b_{nk}^m$ in advance like gMMLDP, because pooling matrix $\mathbf{B}_{nk}$ changes at each iteration. Furthermore, the dimension of $\boldsymbol{z}_n$ sometimes is large according to (15), so we have to calculate the inverse matrix when sample $\boldsymbol{w}$. For oMMCDP it takes shorter time because a small batchsize $\tilde{N}$ is needed and the gradients are calculated to update A and W instead of inversing a big matrix. The computational complexity of DNN example is different according to the type of DNN, but all gradients for $\boldsymbol{\Theta}$ can be quickly and efficiently got by BP algorithm, with other parameters similar to that of oMMLDP.

V. EXPERIMENTS

In this section, we illustrate the effectiveness of our proposed framework and examples by conducting experiments on four image bechmarks (MNIST [28], CIFAR-10 [49], STL-10 [50], and SVHN [51]). In the following, we first introduce four image datasets used to assess the performance of each algorithm including the detailed parameter configurations of each example of framework. Afterward, recognition accuracy of different models are showed with detailed analysis and the learned filters by MMCDP are displayed. Then in the next part, main hyper-parameters used in our examples are evaluated with MNIST dataset. What is more, we investigate how our models work in a specific target recognition problem, radar HRRP automatic target recognition (ATR), followed [52]–[54] with detailed data description and result analysis.

A. Datasets and Basic Configurations of MMDP

Detailed illustrations of the four datasets are listed in Table II, where global contrast normalization (GCN) and ZCA whitening follows [55], while LCN represents local contrast normalization following [42]. Some hyper-parameter used are as follows which are consistent for all datasets. The prior in our model is set as $p(\boldsymbol{\Psi}_g) = N(0, \sigma^2\mathbf{I})$ like [6], with $\sigma^2 = 1$ if not specifically mentioned. In addition, we employ a block decay strategy for stepsize ε for scalable inference, which is decreases by half every T epochs and $T = 20$. We apply max-pooling and rectified linear unit in CNN-like example. The concrete configurations for different dataset using different models are shown in Appendix D, in the supplementary material. All the experiments are implemented on MATLAB and run on an Intel Core i7-4790 3.60 GHz CPU with 16.00 GB RAM.

TABLE III
TEST ERROR OF THE CLASSIFICATION RESULTS ON ALL FOUR DATASETS

Models	MNIST	CIFAR-10	STL-10	SVHN
RBM[49]	-	34.30%	48.50%	-
DBN[56]	1.20%	-	-	-
FNN[6]	1.59%	-	-	-
MMVA[42]	0.90%	-	-	-
gMMLDP	9.45%	-	-	-
oMMLDP	9.38%	-	-	-
MMDP-MLP	1.23%	-	-	-
CDBN[57]	0.82%	21.12%	-	-
CNN[58]	0.55%	25.94%	40.35%	4.90%
SPCNN[59]	0.47%	15.13%	-	2.80%
CMMVA[42]	0.45%	-	-	3.09%
CKN[60]	0.53%	21.70%	39.73%	-
MMCDP-Layer1	3.21%	30.23%	54.72%	17.23%
MMCDP-Layer2	1.41%	23.87%	47.61%	12.46%
MMCDP-Layer3	0.89%	20.48%	41.66%	6.67%
MMDP-CNN	0.62%	18.67%	39.84%	4.82%
MMDP-CNN-with-pretrain	0.59%	17.89%	37.67%	3.96%

B. Classification Results on Various Datasets

The different examples of MMDP is compared with other models, whose classification results on each dataset are demonstrated in Table III with no data augmentation. It is worth noting that since single-layer MMCDP can be seen a "base layer" of MMDP-CNN, we can use the filters learned by MMCDP to initialize the ones in MMDP-CNN.

According to the results we can see that: 1) the linear example, MMLDP, achieves the worst results compared with other examples and hard to use directly for some datasets due to its high computational complexity for high dimensional data; 2) high-layer MMCDP evidently promotes the recognition rate based on low-layer model, which proves that deep architecture of MMCDP can learn more discriminative latent feature space; and 3) our proposed example of framework, convolutional example (MMCDP) and deep network example (MMDP-MLP and MMDP-CNN), are competitive with related models, indicating the effectiveness of special examples and universality of the framework.

C. Computational Time

Although we give theoretical computational complexity of MMLDP and MMCDP in Table I to illustrate the efficiency of our scalable inference compared with Gibbs sampling, in this part we list the average time for one epoch of some representative algorithms (for CIFAR-10 STL-10 and SVHN, we pay more attention on convolution-like models because a comparable classification performance is got by these algorithms) at training stage in Table IV to give a more intuitive impression, especially for convolutional-type algorithms. Although the time spent by each epoch in MMCDP and MMDP-CNN is longer than CNN because a few sampling steps are needed to update the posterior of $\boldsymbol{z}$, the number of epochs in our algorithms is smaller than CNN. In detail, to achieve the performance listed in Table III, CNN takes 500 epochs but our models take 300 epochs thanks to the distribution estimation over parameters in DNN and the classifier. In terms

TABLE IV
ONE EPOCH TIME OF SOME ALGORITHMS AT TRAINING STAGE IN SECONDS (S) ON ALL FOUR DATASETS

Models	MNIST	CIFAR-10	STL-10	SVHN
DBN[56]	23.2	-	-	-
oMMLDP	25.5	-	-	-
MMDP-MLP	38.6	-	-	-
CNN[58]	36.8	43.2	42.3	196.6
MMCDP-Layer1	56.2	68.9	78.5	274.2
MMCDP-Layer2	49.7	61.3	69.2	253.9
MMCDP-Layer3	40.1	53.2	60.1	237.6
MMDP-CNN	42.8	46.7	45.8	212.3

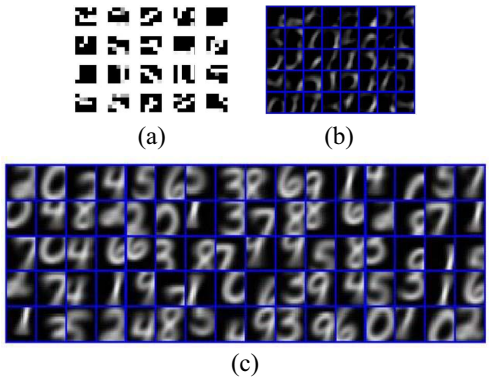


Fig. 2. Filters learned from MNIST dataset by MMCDP at different layers. (a) Layer 1. (b) Layer 2. (c) Layer 3.

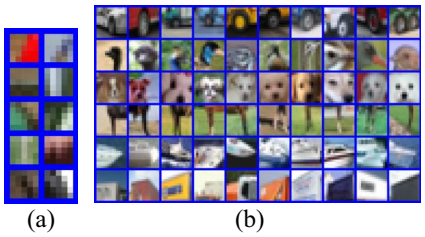


Fig. 3. Filters learned from CIFAR-10 dataset by MMCDP at different layers. (a) Layer 1. (b) Layer 2. Note that each row corresponds one filter in (b).

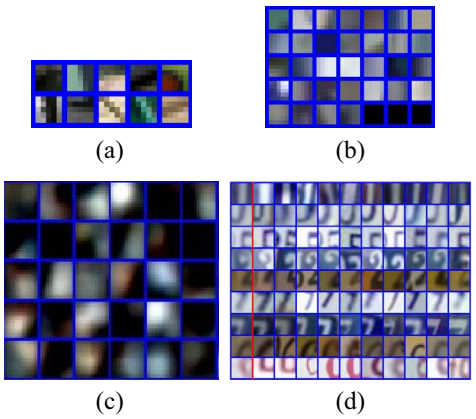


Fig. 4. Filters learned from STL-10 and SVHN dataset by MMCDP. Learned from STL-10 at (a) layer 1 and (b) layer 2. Learned from SVHN at (c) layer 1 and (d) layer 2.

of testing time, since MMDP-CNN needs to collect some random samples $\boldsymbol{w}$, $\boldsymbol{\Theta}$ according to (31), it takes a bit longer than CNN at testing stage. For instance, it takes about 0.13 s on one sample of SVHN and 0.38 s on one sample of STL10 for CNN, while 0.28 s on one sample of SVHN and 0.51 s on one sample of STL10 for MMDP-CNN.

D. Filter Visualization

For the specific example MMCDP, we introduce an efficient method to show the visualized filters in order to understand how the model works. In this part, we list the results of five datasets, respectively, and give a detailed analysis.

1) MNIST: Fig. 2 shows the learned filters at each layer. It can be observed qualitatively that the filters at layer one are only some strong points like a texture extractor; filters at layer two are much sharper and take on forms characteristic of digits and parts of digits; filters at layer three have much more meaningful shape, exceedingly similar with the digits. In other words, different layers of the model concentrate on different characteristics of the digits.

2) CIFAR-10: We give the visualization results on CIFAR-10 in Fig. 3. For layer one, the filters are shown directly. Since the pictures are small, the selective regions contain most of colorful images, which makes no sense if we show weighted mean results directly. Therefore, we select the top ten regions $\mathbf{X}_n^{\text{region}}$ in (18) with the highest activations, and choose the more obvious meaning filters to show like [26]. It can be seen that the top ten parts for one filter are very similar in physical, like animals' leg and head, the wheel and the prow.

3) STL-10: The filters learned on STL-10 dataset is shown in Fig. 4. The layer-one filters are shown directly and some meaningful colorful simple textures can be observed like that on CIFAR-10. For layer 2, although the weighted mean causes the filters having less obvious physical meaning than MNIST dataset, the results are persuasive because the structural ones indicate the selective regions $\{\mathbf{X}_n^{\text{region}}\}_{n=1}^{N'}$ are similar. The "black smooth" ones, we think, mean the selective regions are not similar and their corresponding filters are meaningless. Moreover, we observe that the coefficients of classifier corresponding to the features obtained by these meaningless filters are relative small.

4) SVHN: The visualization is shown in Fig. 4. Among them, all the filters at layer one are shown in Fig. 4(a) and some meaningful filters at layer two are shown in Fig. 4(b) with the first column denoting the weighted mean results and other columns denoting top ten samples. We can come to a similar conclusions like those stated above.

E. Sensitivity Analysis of MMDP

In this paper, we propose the scalable inference for our MMDP framework and apply it to different examples. In this part, we compare the convergence speed and analyze the models' sensitivity to some key parameters on MNIST dataset, whose results give a reference setting for other datasets.

1) Batch Size $|B|$: Figs. 5–7 present the test error with different batch sizes for three examples, respectively. We can see that the convergence speed and accuracy of different batch

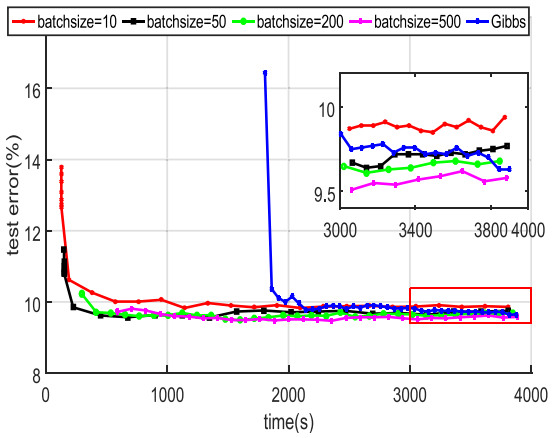


Fig. 5. Test errors of gMMLDP and oMMLDP with different batch sizes on the MNIST dataset.

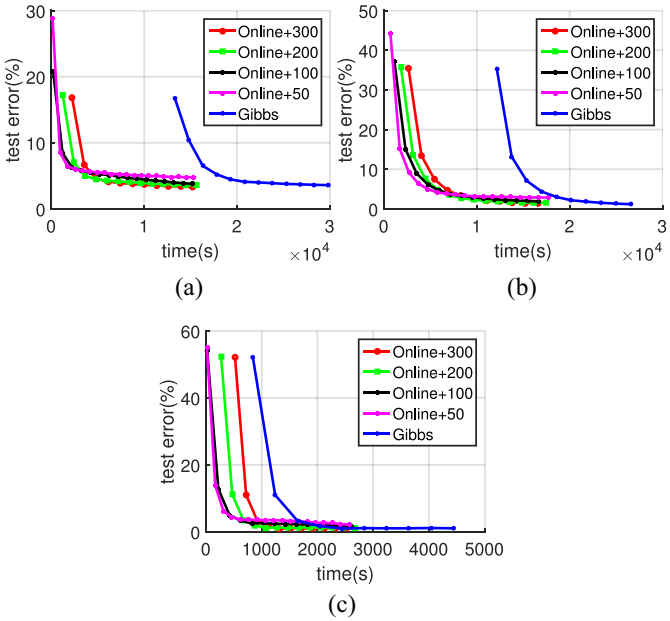


Fig. 6. Test errors of gMMCDP and oMMCDP with different batch sizes on the MNIST dataset. (a) Layer 1. (b) Layer 2. (c) Layer 3.

sizes vary. First, the convergence speed of our scalable inference method is must faster than Gibbs sampling and both of them have similar results when convergence. Second, if we choose a batch size too small, such as $|B| = 10$, the performance is low because each iteration would not provide sufficient evidence for updating. Thirdly, although the larger batch size the lower error, we usually choose medium batch sizes in practice with the thought of the balance between test error and convergence speed.

Moreover, since the dimension of z_n decreases with the increase of the number of layers for MMCDP, the complexity for calculating the inverse of matrix in sampling classifier gets smaller, which leads to a smaller difference between oMMCDP and gMMCDP. Comparing Figs. 5 and 6 with Fig. 7, we find that oMMCDP and oMMLDP possess stronger robustness with smaller batchsize than MMDP-MLP and MMDP-CNN. It illustrates that a proper thermostat ξ introduced by SGNHT

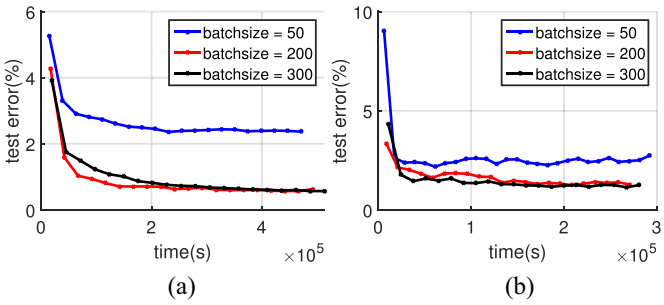


Fig. 7. Test errors of (a) MMDP-MLP and (b) MMDP-CNN with different batch sizes on the MNIST dataset.

TABLE V
EFFECT OF THE NUMBER OF SAMPLES AND BURN-IN STEPS USED MMDP-CNN ON MNIST DATASET

J \ β	0	2	4	6	8
3	0.71	0.78			
5	0.64	0.67	0.76		
9	0.63	0.64	0.66	0.67	0.73
11	0.63	0.65	0.67	0.68	0.74

adaptively controls the stochastic noise caused by small batchsize. In our experiment, we attempt to use SGNHT for DNN example but find that although it have stronger robustness for small batchsize, the convergence speed is much slower than pSGLD. So in the future research, we consider add thermostat ξ into pSGLD to increase the robustness of our model.

2) Number of Leapfrog and Samples: Since the computational complexity of Algorithm 1 is linear in both I and J, while Algorithm 2 is linear in J, we desire to know how these parameters influence the quality of the models.

First, in order to understand how large β samples which are discarded after burn-in step, is sufficient, we consider the setting of the pairs (J, β) and check the consequence of three examples with $|\mathrm{B}| = 200$. We conduct the experiment on the model of MMDP-CNN with the results shown in Table V. It can be seen that the number of collected samples to calculate the expectation in (24) and (25) exhibits a more significant role in the performance than the burn-in steps. In other words, we often choose small β and big J to achieve better accuracy.

Second, we analyze which pair of (I, J) in Algorithm 1 makes superior classification, whose result is presented in Fig. 8. As we can see, for $J = 1$, the accuracy is lower for all I because of noisy approximation of the expectation. But for large enough J, simply $I = 1$ is promising. In addition, we observe that algorithms with medium number of iterations ($I = 3, 5$) result in better performance, which may be attributed to the fact that too small I may lead to updating incompletely and too big I may lead to falling into local solution. It means that it is better for the minibatch and worse for the whole data.

F. Model Uncertainty

In this section, we do another experiment to give more evidence to support the motivation of using Bayesian methods.

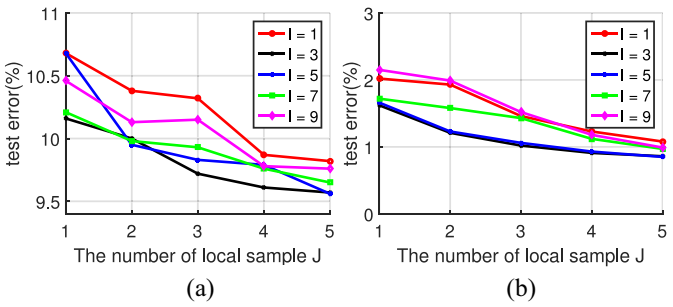


Fig. 8. Test errors of oMMLDP and MMCDP with different combinations of (I, J) on the MNIST dataset. (a) oMMLDP. (b) MMCDP at layer 3.

TABLE VI
MNIST CLASSIFICATION TASK WITH DIFFERENT TRAINING SIZES

Model	Number of Training data			
	1000	5000	30000	60000
CNN	84.62±1.05	96.58±0.33	98.85±0.18	99.39±0.12
MMDP-CNN	89.16±0.85	96.95±0.27	98.86±0.15	99.38±0.11

We train standard CNN and MMDP-CNN on MNIST with different training data sizes with results shown in Table VI. The results were achieved by 100 independent experiments. From Table VI, we can see that when the number of training data is large enough, the performance of the point estimation-based CNN is as good as Bayesian method, but Bayesian method performs better when the training set is relative small, which illustrates the advantage of the Bayesian method: do a better response for out-of-sample [6]. In addition, the variance of Bayesian method is smaller than that of the point estimation especially when the training data is small which is attributed to the distribution estimation of Bayesian SVM and model uncertainty in network's parameters.

G. Radar High-Resolution Range Profile Target Recognition

We consider measured radar HRRP signal from three real airplanes including Yak-42 (a large and medium-sized jet aircraft), Cessna Citation S/II (a small-sized jet aircraft), and An-26 (a medium-sized propeller aircraft), which are widely used in this paper field [52]–[54]. The detailed parameters about radar and airplanes have been listed in Appendix E, in the supplementary material. including some data examples.

In order to demonstrate the robustness about elevation angles, according to [53], we choose the data from the second and the fifth segments of Yak-4, the sixth and the seventh segments of Cessna Citation S/II, the fifth and the sexth segments of An-26 as the training samples, and other data segments are taken as test samples with concrete illustration shown in the supplementary material. Therefore, there are 14 000 HRRP samples for training and 5200 samples for test with each sample being a 256-D vector. In this experiment, the L_2-norm normalized power spectrum feature of HRRP was used to perform classification on account of its time-shift invariance. Following the parameter setting-up given Appendix D, in the supplementary material. We report the classification performance of related methods on this application in Table VII, where includes the shallow models (LDA, KSVD, and PCA), deep models (DBN, DAE, and CNN) and our models (MMCDP, MMDP-MLP, and MMDP-CNN). LSVM here only serves as a simple baseline with no feature extraction. From the Table VII, we can find several phenomena as follows. First, compared with shallow models, the nonlinear hierarchical architecture of DNN is more advantageous to learn the effective feature for HRRP target recognition which results in a higher recognition rate. Second, our Bayesian deep projection models perform better than DBN and DAE for HRRP recognition. We attribute it to the distribution estimation in Bayesian model and the utilization of supervised information. Third, we apply CNN using 1-D convolution on HRRP dataset as a baseline of our MMDP-CNN algorithm. we find that MMDP-CNN achieves better performance even without pretraining. In the future, the average profile can be taken into account for our framework to extract more robust features like [53]. In addition, Fig. 9 shows the filters learned

TABLE VII
CLASSIFICATION PERFORMANCE FOR THE HRRP DATASET

Methods	Test error
LSVM	13.30%
LDA	18.70%
K-SVD	25.30%
PCA	16.19%
DBNs-1layer	9.96%
DBNs-2layer	9.71%
DBNs-3layer	10.71%
DAEs-1layer	11.69%
DAEs-2layer	9.71%
DAEs-3layer	9.58%
CNN-1layer	11.25%
CNN-2layer	9.36%
CNN-3layer	8.30%
MMCDP-1layer	10.38%
MMCDP-2layer	8.41%
MMDP-MLP	8.39%
MMDP-CNN	8.26%
MMDP-CNN-with-pretrain	8.07%

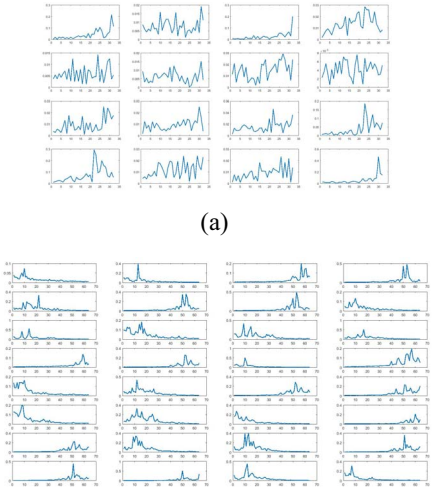


Fig. 9. Filters learned from HRRP dataset by MMCDP. (a) Layer 1. (b) Layer 2.

by MMCDP on HRRP dataset, which is rarely shown in other models. Different with previous image datasets, HRRP is a 1-D signal resulting in less meaningful filters. However, we can also observe some structural information like wavelet.

VI. Conclusion

We propose a unified framework for Bayesian MMDP called MMDP with three special examples discussed, including MMLDP, MMCDP, and MMDP-DNN. Among them, MMLDP is an existing linear projection which can be also kernelized for nonlinear projection. MMCDP and MMDP-DNN are our proposed new models based on the framework. Both of them have deep structure and utilize convolution operation to improve the capability of feature extraction. Furthermore, we also provide different scalable inference for different examples. In experiments, we achieve comparable classification accuracy on MNIST, CIFAR-10, STL-10, SVHN, and radar HRRP target recognition. Moreover, we can see that MMCDP can learn meaningful filters via the visualization method. In addition, we conduct our models in radar HRRP ATR field and achieve excellent performance.

In all experiments, we choose MATLAB as our computing platform, which may limit the computation ability. In the future, we are interested in building better structure under our framework with help of GPU on some sophisticated package and program language.

References

[1] S. Ding and Z. Shi, "Supervised feature extraction algorithm based on improved polynomial entropy," *J. Inf. Sci.*, vol. 32, no. 4, pp. 309–315, 2006.

[2] R. B. Girshick, "Fast R-CNN," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 1440–1448.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[4] S. Hong, H. Oh, and B. Han, "Learning transferrable knowledge for semantic segmentation with deep convolutional neural network," in *Proc. Comput. Vis. Pattern Recognit.*, 2015, pp. 3204–3212.

[5] T. Shi and J. Zhu, "Online Bayesian passive-aggressive learning," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 1084–1122, 2014.

[6] C. Li, C. Chen, D. Carlson, and L. Carin, "Preconditioned stochastic gradient Langevin dynamics for deep neural networks," in *Proc. Amer. Assoc. Artif. Intell.*, 2016, pp. 1788–1794.

[7] A. Korattikara, V. Rathod, K. Murphy, and M. Welling, "Bayesian dark knowledge," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28. Montreal, QC, Canada, 2015, pp. 3438–3446.

[8] J. M. Hernandezlobato and R. P. Adams, "Probabilistic backpropagation for scalable learning of Bayesian neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1861–1869.

[9] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1613–1622.

[10] H. Wang, X. Shi, and D. Yeung, "Natural-parameter networks: A class of probabilistic neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 118–126.

[11] C. M. Bishop, *Pattern Recognition and Machine Learning* (Information Science and Statistics). New York, NY, USA: Springer-Verlag, 2006.

[12] B. Chen *et al.*, "Deep learning with hierarchical convolutional factor analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1887–1901, Aug. 2013.

[13] V. Vapnik, *The Nature of Statistical Learning Theory* (Information Science and Statistics). New York, NY, USA: Springer-Verlag, 2000.

[14] N. G. Polson and S. L. Scott, "Data augmentation for support vector machines," *Bayesian Anal.*, vol. 6, no. 1, pp. 1–24, 2011.

[15] J. Zhu, N. Chen, H. Perkins, and B. Zhang, "Gibbs max-margin topic models with data augmentation," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1073–1110, 2014.

[16] C. Li, J. Zhu, and J. Chen, "Bayesian max-margin multi-task learning with data augmentation," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 415–423.

[17] Y. Pu *et al.*, "Variational autoencoder for deep learning of images, labels and captions," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2352–2360.

[18] B. Chen *et al.*, "Max-margin discriminant projection via data augmentation," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 7, pp. 1964–1976, Jul. 2015.

[19] C. E. Antoniak, "Mixture of Dirichlet processes with applications to Bayesian nonparametric problems," *Ann. Stat.*, vol. 2, no. 6, pp. 1152–1174, 1974.

[20] X. Zhang, B. Chen, H. Liu, L. Zuo, and B. Feng, "Infinite max-margin factor analysis via data augmentation," *Pattern Recognit.*, vol. 52, pp. 17–32, Apr. 2016.

[21] M. Dorfer, R. Kelz, and G. Widmer, "Deep linear discriminant analysis," in *Proc. Int. Conf. Learn. Represent.*, 2015.

[22] G. Andrew, R. Arora, J. A. Bilmes, and K. Livescu, "Deep canonical correlation analysis," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1247–1255.

[23] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Represent.*, 2014.

[24] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1278–1286.

[25] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 818–833.

[26] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene CNNs," in *Proc. Int. Conf. Learn. Represent.*, 2014.

[27] C. R. Gent and C. P. Sheppard, "Predicting time series by a fully connected neural network trained by back propagation," *Comput. Control Eng. J.*, vol. 3, no. 3, pp. 109–112, 1992.

[28] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[29] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient Langevin dynamics," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 681–688.

[30] T. Chen, E. B. Fox, and C. Guestrin, "Stochastic gradient hamiltonian Monte Carlo," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1683–1691.

[31] N. Ding *et al.*, "Bayesian sampling using stochastic gradient thermostats," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3203–3211.

[32] X. Shang, Z. Zhu, B. Leimkuhler, and A. J. Storkey, "Covariance-controlled adaptive Langevin thermostat for large-scale Bayesian sampling," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 37–45.

[33] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Netw. Mach. Learn.*, vol. 4, no. 2, pp. 26–31, 2012.

[34] H.-T. Chen, H.-W. Chang, and T.-L. Liu, "Local discriminant embedding and its variants," in *Proc. Comput. Vis. Pattern Recognit.*, 2005, pp. 846–853.

[35] X. Wang and X. Tang, "Dual-space linear discriminant analysis for face recognition," in *Proc. Comput. Vis. Pattern Recognit.*, Washington, DC, USA, 2004, pp. 564–569.

[36] J. Ye and T. Wang, "Regularized discriminant analysis for high dimensional, low sample size data," in *Proc. Int. Conf. Knowl. Disc. Data Mining*, 2006, pp. 454–463.

[37] L. Clemmensen, T. Hastie, D. Witten, and B. Ersbøll, "Sparse discriminant analysis," *Technometrics*, vol. 53, no. 4, pp. 406–413, 2012.

[38] S. Yu, K. Yu, V. Tresp, H. P. Kriegel, and M. Wu, "Supervised probabilistic principal component analysis," in *Proc. Int. Conf. Knowl. Disc. Data Mining*, 2006, pp. 464–473.

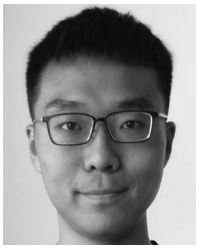
[39] P. Rai and H. Daumé, III, "Multi-label prediction via sparse infinite CCA," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1518–1526.

[40] M. Gonen, "Bayesian supervised dimensionality reduction," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 2179–2189, Dec. 2013.

[41] B. Taskar, C. Guestrin, and D. Koller, "Max-margin Markov networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 25–32.

[42] C. Li, J. Zhu, T. Shi, and B. Zhang, "Max-margin deep generative models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1837–1845.

[43] D. J. C. Mackay, "A practical Bayesian framework for backpropagation networks," *Neural Comput.*, vol. 4, no. 3, pp. 448–472, 1992.
[44] M. E. Tuckerman, *Statistical Mechanics: Theory and Molecular Simulation*. Oxford, U.K.: Oxford Univ. Press, 2010.
[45] J. Lu, S. C. H. Hoi, J. Wang, P. Zhao, and Z.-Y. Liu, "Large scale online kernel learning," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 1613–1655, 2016.
[46] Y. Pu, X. Yuan, A. Stevens, C. Li, and L. Carin, "A deep generative deconvolutional image model," in *Proc. 19th Int. Conf. Artif. Intell. Stat.*, 2016, pp. 741–750.
[47] M. Girolami and B. Calderhead, "Riemann manifold Langevin and hamiltonian Monte Carlo methods," *J. Royal Stat. Soc. B (Stat. Methodol.)*, vol. 73, no. 2, pp. 123–214, 2011.
[48] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.
[49] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Univ. Toronto, Toronto, ON, Canada, May 2009.
[50] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. Int. Conf. Artif. Intell. Stat.*, vol. 15, 2011, pp. 215–223.
[51] Y. Netzer *et al.*, "Reading digits in natural images with unsupervised feature learning," in *Proc. Adv. Neural Inf. Process. Syst. Workshop Deep Learn. Unsupervised Feature Learn.*, 2012.
[52] B. Chen, H. Liu, J. Chai, and Z. Bao, "Large margin feature weighting method via linear programming," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 10, pp. 1475–1488, Oct. 2009.
[53] B. Feng, B. Chen, and H. Liu, "Radar HRRP target recognition with deep networks," *Pattern Recognit.*, vol. 61, pp. 379–393, Jan. 2017.
[54] L. Du *et al.*, "Noise robust radar HRRP target recognition based on multitask factor analysis with small training data size," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3546–3559, Jul. 2012.
[55] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Courville, and Y. Bengio, "Maxout networks," in *Proc. Int. Conf. Mach. Learn.*, vol. 28, 2013, pp. 1319–1327.
[56] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
[57] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. Int. Conf. Mach. Learn.*, 2009, pp. 609–616.
[58] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 675–678.
[59] M. D. Zeiler and R. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2013.
[60] J. Mairal, P. Koniusz, Z. Harchaoui, and C. Schmid, "Convolutional kernel networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2627–2635.



Hao Zhang received the B.S. degree in electronic engineering from Xidian University, Xi'an, China, in 2012, where he is currently pursuing the Ph.D. degree in signal and information processing.

His current research interests include statistical machine learning and radar automatic target recognition.

Bo Chen (M'13) received the B.S., M.S., and Ph.D. degrees from Xidian University, Xi'an, China, in 2003, 2006, and 2008, respectively, all in electronic engineering.

He became a Postdoctoral Fellow, a Research Scientist, and a Senior Research Scientist with the Department of Electrical and Computer Engineering, Duke University, Durham, NC, USA, from 2008 to 2012. Since 2013, he has been a Professor with the National Laboratory for Radar Signal Processing, Xidian University. His current research interests include statistical machine learning, statistical signal processing, and radar automatic target detection and recognition.

Dr. Chen was a recipient of the Honorable Mention for 2010 National Excellent Doctoral Dissertation Award and is selected into One Thousand Young Talent Program in 2014.

Zhengjue Wang received the B.S. and M.S. degrees in electronic engineering from Xidian University, Xi'an, China, in 2013 and 2016, respectively, where she is currently pursuing the Ph.D. degree in signal and information processing.

Her current research interests include machine learning and radar automatic target recognition.

Hongwei Liu (M'04) received the M.S. and Ph.D. degrees in electronic engineering from Xidian University, Xi'an, China, in 1995 and 1999, respectively.

From 2001 to 2002, he was a Visiting Scholar with the Department of Electrical and Computer Engineering, Duke University, Durham, NC, USA. He is currently a Professor with the National Laboratory of Radar Signal Processing, Xidian University. His current research interests include radar automatic target recognition, radar signal processing, and adaptive signal processing.