




# Online Bayesian max-margin subspace learning for multi-view classification and regression

Jia He<sup>1,2,4</sup> · Changying Du<sup>3,5</sup> · Fuzhen Zhuang<sup>1,4</sup>  · Xin Yin<sup>1</sup> · Qing He<sup>1,4</sup> · Guoping Long<sup>5</sup>

Received: 23 April 2018 / Revised: 16 September 2019 / Accepted: 4 October 2019 /

Published online: 25 October 2019

© The Author(s), under exclusive licence to Springer Science+Business Media LLC, part of Springer Nature 2019

## Abstract

Multi-view data have become increasingly popular in many real-world applications where data are generated from different information channels or different views such as image + text, audio + video, and webpage + link data. Last decades have witnessed a number of studies devoted to multi-view learning algorithms, especially the predictive latent subspace learning approaches which aim at obtaining a subspace shared by multiple views and then learning models in the shared subspace. However, few efforts have been made to handle online multi-view learning scenarios. In this paper, we propose an online Bayesian multi-view learning algorithm which learns predictive subspace with the max-margin principle. Specifically, we first define the latent margin loss for classification or regression in the subspace, and then cast the learning problem into a variational Bayesian framework by exploiting the pseudo-likelihood and data augmentation idea. With the variational approximate posterior inferred from the past samples, we can naturally combine historical knowledge with new arrival data, in a Bayesian passive-aggressive style. Finally, we extensively evaluate our model on several real-world data sets and the experimental results show that our models can achieve superior performance, compared with a number of state-of-the-art competitors.

**Keywords** Multi-view learning · Online learning · Bayesian subspace learning · Max-margin · Classification · Regression

## 1 Introduction

Nowadays, multi-view data are often generated from multiple information channels continuously, e.g., hundreds of YouTube videos consisting of visual, audio and text features are uploaded every minute. Different views usually contain complementary information, and multi-view learning can exploit this information to learn representation that is more expressive than that of single-view learning method. Therefore, multi-view representation learning has become a very promising topic with wide applicability. Multi-view learning arouses amounts of interests in the past decades (Zhao et al. 2017; Quang et al. 2013; Sun

---

Editor: Ulf Brefeld.

---

Extended author information available on the last page of the article

and Chao 2013; Li et al. 2016; Ye et al. 2015; Liu et al. 2016; Chen and Zhou 2018). Nowadays, there are many multi-view learning approaches, e.g., multiple kernel learning (Gönen and Alpaydm 2011), disagreement-based multi-view learning (Blum and Mitchell 1998), late fusion methods which combine outputs of the models constructed from different view features (Ye et al. 2012) and subspace learning methods for multi-view data (Chen et al. 2012). Among them, the multi-view subspace learning approaches aim at obtaining a subspace shared by multiple views and then learning models in the shared subspace (Sharma et al. 2012; Hardoon et al. 2004; Guo and Xiao 2012; Zhang et al. 2017; Brbić and Kopriva 2018). They are very useful for cross-view classification and retrieval tasks. However, these approaches may overfit small training data. A max-margin harmonium model (MMH) (Chen et al. 2012) was proposed to avoid overfitting by introducing the max-margin principle to the latent subspace Markov network for multi-view data. But MMH is under the maximum entropy discrimination framework and cannot infer the penalty parameter of max-margin models in Bayesian style automatically. In Du et al. (2015), a posterior-regularized Bayesian approach is proposed to combine principal component analysis (PCA) with the max-margin learning, which can infer the penalty parameter of max-margin models but cannot address multi-view data.

On the other hand, multi-view data often cannot be collected in a single time due to temporal and spatial constrictions in applications, while the traditional multi-view algorithm needs store the entire training samples. Online learning is an efficient method to address this problem. Online learning starts from the Perceptron algorithm (Rosenblatt 1958) and many efforts have been made on the studies of online learning (Cesa-Bianchi and Lugosi 2006; Hazan et al. 2007; Chechik et al. 2010). Unfortunately, there are few studies about online multi-view learning methods. OPMV (Zhu et al. 2015) is one of the few online multi-view learning algorithms. OPMV jointly optimizes the composite objective functions with consistency linear constraints for different views. It doesn't take the relevance of different views into account. OPMV is formulated as a point estimate by optimizing some deterministic objective function, without consideration of the model uncertainty. Online passive-aggressive (PA) learning provides a method for online large-margin learning (Crammer et al. 2006). Although it enjoys strong discriminative ability suitable for predictive tasks, it is also formulated as a point estimate by optimizing some deterministic objective function. The point estimate can be affected seriously by inappropriate regularization, outliers and noises, especially when the training data arrive sequentially. Based on the online PA learning, Shi and Zhu (2013) proposed a Bayesian PA learning method which infers a posterior under the Bayesian framework instead of a point estimate. Nevertheless, these online learning methods cannot process multi-view data. To the best of our knowledge, there have been few efforts focused on online multi-view learning under the Bayesian framework.

We address the aforementioned problems by developing an online Bayesian multi-view subspace learning method with the max-margin principle. Specifically, we first propose a predictive subspace learning method based on factor analysis and define a latent margin loss for classification in the subspace. Then we cast the learning problem into a variational Bayesian framework by exploiting the pseudo-likelihood and data augmentation idea (Zhu et al. 2014) which allows us to automatically infer the penalty parameter. With the variational approximate posterior inferred from the past samples, we can naturally combine historical knowledge with new arriving data, in a Bayesian passive-aggressive style. We update our model with the training data coming batch by batch, instead of storing all training data.

In the previous work, we propose a Bayesian multi-view learning algorithm (He et al. 2016) to learn predictive subspace with max-margin principle and then extend this model to the online scenario with streaming data. However, our preliminary work mainly focuses on

classification tasks and is not available to predict the continuous response values like rating scores for hotel reviews and movie reviews.

In this paper, we further extend our Bayesian multi-view subspace learning model to solve multi-view regression problem which is based on the  $\epsilon$ -insensitive Support Vector Regression (SVR) (Zhu et al. 2009). Regression tasks have been successfully applied in many fields like finance (Jiang and He 2012; Kazem et al. 2013), transportation (Lin et al. 2013), meteorology (Kalteh 2013; Chen and Jie 2014), and other fields. We have applied our proposed regression model on many real multi-view regression data sets to predict the global rating score of the hotel, the rating of the movie and the relative location of the CT image on the axial axis, while our classification model can not predict the continuous values of regression tasks. Furthermore, we also extend our batch regression model to the online scenario which can deal with streaming data. Experiments on synthetic and various real classification and regression tasks show both our batch and online models have superior performance, compared with a number of competitors.

The paper is structured as follows. Section 2 introduces the related work. Section 3 presents the Bayesian subspace multi-view classification (regression) models and their online versions. Section 4 presents the variational inference for our models. Section 5 presents empirical results and Sect. 6 concludes.

## 2 Related work

The earliest works of multi-view learning are introduced by Blum and Mitchell (1998) and Yarowsky (1995). Nowadays, there are many multi-view learning approaches, e.g., multiple kernel learning (Gönen and Alpaydm 2011), disagreement-based multi-view learning (Blum and Mitchell 1998) and late fusion methods which combine outputs of the models constructed from different view features (Ye et al. 2012). Especially, the multi-view subspace learning algorithms learn the latent salient representation of multi-view data (Sharma et al. 2012; Hardoon et al. 2004; Zhang et al. 2017; Brbić and Kopriva 2018). This approach aims at obtaining a subspace shared by multiple views and then learning models in the shared subspace. However, most of these approaches are formulated as a point estimate by optimizing some deterministic objective function. The point estimate can be affected seriously by inappropriate regularization, outliers and noises.

The above methods mainly focus on multi-view classification tasks, while multi-view regression tasks of which the response values are continuous are also applied in many fields. Most of the existing multi-view regression models are based on a maximum likelihood approach or optimizing some deterministic objective function (Zheng et al. 2015; Lan et al. 2016; Wang et al. 2013; Merugu et al. 2006), whose objective is to learn an optimal value for each parameter involved in the model, thus can be affected seriously by outliers and noises. What's worse, most existing multi-view regression solvers require user pre-specified the penalty parameter as input, but in many real machine learning applications, the optimal penalty parameter may be hard to be determined in advance. Though these solvers provide an automatic parameter selection mechanism by conducting cross-validation on the training samples, they may overfit small training sets.

Online learning starts from the Perceptron algorithm (Rosenblatt 1958) and has attracted much attention during the past years (Cesa-Bianchi and Lugosi 2006; Hazan et al. 2007; Grangier and Bengio 2008; Chechik et al. 2010; Chen et al. 2017). Crammer proposes the Online passive-aggressive (PA) learning which provides a general framework for online large-

margin learning (Crammer et al. 2006), with many applications (Chiang et al. 2008). Online Bayesian passive-aggressive learning presents a generic framework of performing online learning for Bayesian max-margin models (Shi and Zhu 2013). Chen et al. (2017) proposes an online partial least square optimization method to study a non-convex formulation for multi-view representation learning, which can be efficiently solved by a simple stochastic gradient method. Xie et al. (2017) proposes dynamic multi-view hashing (DMVH), which can adaptively augment hash codes according to dynamic changes of image. And an online unsupervised multi-view model is proposed for feature selection (Shao et al. 2016).

Unfortunately, there are few online multi-view learning methods for multi-view data. OPMV (Zhu et al. 2015) is one of the few online multi-view classification models. This approach jointly optimizes the composite objective functions with consistency linear constraints for different views. It doesn't take the cross view latent relationship into consideration. OPMV is formulated as a point estimate by optimizing some deterministic objective function, without consideration of the model uncertainty. For regression, there are some online regression models (Parrella 2007; Ting 2011; Nguyen-Tuong et al. 2009; Deng et al. 2016). These methods can deal with the multi-view learning by concatenating all views to form a new single view. But these methods don't take the relevance of the different views into account and most of the existing models are based on point estimation, without consideration of the model uncertainty. As a result, their prediction performance can be effected seriously by online outliers and noises.

### 3 The proposed model

In this section, we firstly propose the max-margin subspace learning based on factor analysis. Then we develop a multi-view classification and a multi-view regression with max-margin subspace learning under the Bayesian framework. Finally, we extend the batch model to the online scenario which trains the model with the samples coming batch by batch.

#### 3.1 Max-margin subspace learning

Suppose we have a set of  $N$  observations  $\mathbf{x}^{(n)}$ ,  $n = 1, \dots, N$  in  $d$ -dimension feature space. Factor analysis projects an observation into a low dimensional space that captures the latent feature of data. In factor analysis, the generative process for each observation  $\mathbf{x}$  is as follows:

$$\begin{aligned} \mathbf{z} &\sim \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}_m) \\ \varepsilon &\sim \mathcal{N}(\varepsilon|\mathbf{0}, \Phi) \\ \mathbf{x} &= \boldsymbol{\mu} + \mathbf{W}\mathbf{z} + \varepsilon, \end{aligned} \tag{1}$$

where  $\varepsilon \in \mathbb{R}^{d \times 1}$  denotes the Gaussian noise,  $\Phi \in \mathbb{R}^{d \times d}$  is a dimension variance matrix of  $\varepsilon$ ,  $\boldsymbol{\mu} \in \mathbb{R}^{d \times 1}$  is the mean value of  $\mathbf{x}$ ,  $\mathbf{W} \in \mathbb{R}^{d \times m}$  is the factor loading matrix,  $\mathbf{z}$  is a  $m$ -dimensional latent variable.

So we can get the maximum likelihood  $\ell_s$  of  $\{\mathbf{x}^{(n)}\}$ ,  $n = 1, \dots, N$

$$\max_{\boldsymbol{\mu}, \mathbf{W}, \Phi} \ell_s(\boldsymbol{\mu}, \mathbf{W}, \Phi) = \max_{\boldsymbol{\mu}, \mathbf{W}, \Phi} \log \prod_{n=1}^N \frac{\exp(-\frac{1}{2}(\mathbf{x}^{(n)} - \boldsymbol{\mu})^T (\mathbf{W}\mathbf{W}^T + \Phi)^{-1}(\mathbf{x}^{(n)} - \boldsymbol{\mu}))}{(2\pi)^{d/2} |\mathbf{W}\mathbf{W}^T + \Phi|^{1/2}}.$$

However, factor analysis is an unsupervised model, which learns the latent variables of the observations without using any supervised side information. The max-margin principle

can be introduced to incorporate supervised side information into the factor analysis model. Now, we need to devise a loss function that integrates the max-margin principle for prediction with latent subspace discovery. Suppose we have a  $1 \times N$  response vector  $\mathbf{y}$  with its element  $y^{(n)}, n = 1, \dots, N$ . If there is a classification task, we suppose  $y^{(n)} \in \{+1, -1\}$ . Especially, if there is a regression task,  $y^{(n)}$  will be a continuous value. We define  $\tilde{\mathbf{z}} = [\mathbf{z}^T, 1]^T$  as the augmented latent representation of observation  $\mathbf{x}$ , and let  $f(\mathbf{x}; \tilde{\mathbf{z}}, \boldsymbol{\eta}) = \boldsymbol{\eta}^T \tilde{\mathbf{z}}$  be a discriminant function parameterized by  $\boldsymbol{\eta}$ . For classification, we can compute the following margin loss on training data  $(\mathbf{X}, \mathbf{y})$  with fixed values of  $\mathbf{Z}$  and  $\boldsymbol{\eta}$ :

$$\ell_m(\mathbf{Z}, \boldsymbol{\eta}) = \sum_{n=1}^N \max(0, 1 - y^{(n)} f(\mathbf{x}^{(n)})). \tag{2}$$

For regression, we choose to minimize the  $\epsilon$ -insensitive loss, which is used in standard support vector regression (SVR) (Smola and Lkopf 2004):

$$\ell_m(\mathbf{Z}, \boldsymbol{\eta}) = \sum_{n=1}^N \max(0, |y^{(n)} - f(\mathbf{x}^{(n)})| - \epsilon), \tag{3}$$

where  $\epsilon \in \mathbb{R}_+$  is the precision parameter, which is usually small.

The max-margin subspace learning (M<sup>2</sup>SL) model can be formulated as follows:

$$\max_{\boldsymbol{\mu}, \mathbf{W}, \Phi, \mathbf{Z}, \boldsymbol{\eta}} \ell_s(\boldsymbol{\mu}, \mathbf{W}, \Phi) - C \ell_m(\mathbf{Z}, \boldsymbol{\eta}), \tag{4}$$

where  $C$  is the regularization parameter.

### 3.2 Multi-view classification with Bayesian M<sup>2</sup>SL

Then we propose a Bayesian max-margin subspace multi-view learning (BM<sup>2</sup>SMVL) model. In this section, we present the classification model. We consider the general multi-class classification. The binary case can be similarly derived.

We assume that  $N_v$  is the number of views,  $N_c$  is the number of classes,  $d_i$  is the dimension of the  $i$ -th view, the data matrix of the  $i$ -th view is  $\mathbf{X}_i \in \mathbb{R}^{d_i \times N}$  consisting of  $N$  observations  $\mathbf{x}_i^{(n)}$  in  $d_i$ -dimension feature space. We define  $\mathbf{y}, y^{(n)} \in \{1, \dots, N_c\}, n = 1, \dots, N$  as a  $1 \times N$  label vector and  $\mathbf{x}^{(n)} = \{\mathbf{x}_i^{(n)}\}, i = 1, \dots, N_v$  as the  $n$ -th observation.

In our BM<sup>2</sup>SMVL model, we learn the latent variable  $\mathbf{z}_n$  from which the observation  $\mathbf{x}^{(n)}$  is generated. For the  $n$ -th observation  $\mathbf{x}^{(n)}$ , each view  $\mathbf{x}_i^{(n)}$  of  $\mathbf{x}^{(n)}$  is generated from the latent variable  $\mathbf{z}^{(n)}$ . Next, we impose a prior distribution over the parameters in Eq. (1). So we can get the generative process for the  $n$ -th observations as follows:

$$\begin{aligned} \boldsymbol{\mu}_i &\sim \mathcal{N}(\boldsymbol{\mu}_i | \mathbf{0}, \beta_i^{-1} \mathbf{I}_{d_i}) \\ \alpha_i &\sim \prod_{j=1}^m \Gamma(\alpha_{ij} | a_{\alpha_i}, b_{\alpha_i}) \\ \mathbf{W}_i | \alpha_i &\sim \prod_{j=1}^{d_i} \mathcal{N}(w_{ij} | \mathbf{0}, \text{diag}(i)) \\ \phi_i &\sim \Gamma(\phi_i | a_{\phi_i}, b_{\phi_i}) \\ \mathbf{x}_i^{(n)} | \mathbf{z}^{(n)} &\sim \mathcal{N}(\mathbf{x}_i^{(n)} | \mathbf{W}_i \mathbf{z}^{(n)} + \boldsymbol{\mu}_i, \phi_i^{-1} \mathbf{I}_{d_i}), \end{aligned}$$

where  $\Gamma(\cdot)$  is the Gamma distribution,<sup>1</sup> and  $\beta, a_{\alpha_i}, b_{\alpha_i}, a_{\phi_i}, b_{\phi_i}$  are the hyper-parameters,  $\mathbf{W}_i \in \mathbb{R}^{d_i \times m}$ . The prior on  $\mathbf{W}_i$  and  $\alpha_i$  is introduced according to the automatic relevance determination (ARD) (Reents and Urbanczik 1998). In order to improve the efficiency of our algorithm, we define the variance matrix  $\Phi_i$  of the  $\mathbf{x}_i^{(n)}$  as a diagonal matrix  $\phi_i^{-1} \mathbf{I}_{d_i}$ . Let  $\Omega = (\boldsymbol{\mu}, \boldsymbol{\alpha}, \mathbf{W}, \boldsymbol{\phi}, \mathbf{z})$  denote all the parameters and latent variables.  $p_0(\Omega) = p_0(\boldsymbol{\mu})p_0(\mathbf{W}, \boldsymbol{\alpha})p_0(\boldsymbol{\phi})p_0(\mathbf{z})$  is the prior of  $\Omega$ . According to the regularized Bayesian inference (Zhu et al. 2014), we re-express Eq. (1) as a Bayesian posterior distribution. It can be verified that the Bayesian posterior distribution  $p(\Omega|\mathbf{X}) = p_0(\Omega)p(\mathbf{X}|\Omega)/p(\mathbf{X})$  is equal to the solution of the following optimization problem:

$$\min_{q(\Omega) \in \mathcal{P}} \text{KL}(q(\Omega)||p_0(\Omega)) - \mathbb{E}_{q(\Omega)}[\log p(\mathbf{X}|\Omega)], \tag{5}$$

where  $\text{KL}(q||p)$  is the Kullback–Leibler (KL) divergence, and  $\mathcal{P}$  is the space of probability distributions. When the observations are given,  $p(\mathbf{X})$  is a constant.

Then we introduce a multi-class classification by using the large-margin approach based on the one-VS-rest idea for SVM. We redefine the label  $y^{(n)}$  of the  $n$ -th observation as a  $1 \times Nc$  label vector. If the  $n$ -th observation’s label  $y_n^{(c)}$  belongs to the  $c$ -th class, we define  $y_c^{(n)} = +1$  otherwise  $y_c^{(n)} = -1$ . We have  $Nc$  classifiers, and take the  $c$ -th classification for an example:  $f_c(\mathbf{x}^{(n)}; \tilde{\mathbf{z}}^{(n)}, \boldsymbol{\eta}_c) = \boldsymbol{\eta}_c^T \tilde{\mathbf{z}}^{(n)}$  denotes a discriminant function. Under the Bayesian framework, we impose a prior on  $\boldsymbol{\eta}_c$  as follows:

$$\begin{aligned} v_c &\sim p_0(v_c) = \Gamma(v_c|a_{v_c}, b_{v_c}) \\ p(\boldsymbol{\eta}_c|v_c) &= \mathcal{N}(\boldsymbol{\eta}_c|\mathbf{0}, v_c^{-1} \mathbf{I}_{(m+1)}), \end{aligned}$$

where  $a_{v_c}$  and  $b_{v_c}$  are hyper-parameters.  $v_c$  controls the inverse variance of  $\boldsymbol{\eta}_c$ , playing a similar role as the penalty parameter in conventional SVM. If  $v_c$  has a posterior distribution concentrated at large values,  $\boldsymbol{\eta}_c$  will tend to be small, then the model will be simple and may not perform very well on training data. In conventional SVM, the penalty parameter  $v_c$  is learned by time-consuming cross-validation. Instead, it can be determined automatically as part of Bayesian inference in our model. So we can automatically infer the penalty parameter of the max-margin model. For simplify, let  $\Theta = \{(\boldsymbol{\eta}_c, v_c)\}_{c=1}^{Nc}$ .

Then we can replace the margin loss with the expected margin loss for the classification. So we introduce

$$\varphi(\mathbf{y}|\mathbf{Z}, \boldsymbol{\eta}) = \prod_{n=1}^N \prod_{c=1}^{Nc} \exp\{-2C \cdot \max(0, 1 - y_c^{(n)} \boldsymbol{\eta}_c^T \tilde{\mathbf{z}}^{(n)})\} \tag{6}$$

as the pseudo-likelihood of the  $n$ -th data’s label variable. Next we get our final model as follows:

$$\min_{q(\Omega, \Theta) \in \mathcal{P}} \text{KL}(q(\Omega, \Theta)||p_0(\Omega, \Theta)) - \mathbb{E}_{q(\Omega)}[\log p(\mathbf{X}|\Omega)] - \mathbb{E}_{q(\Omega, \Theta)}[\log(\varphi(\mathbf{y}|\mathbf{Z}, \boldsymbol{\eta}))], \tag{7}$$

where  $p_0(\Omega, \Theta)$  is the prior,  $p_0(\Omega, \Theta) = p_0(\Omega)p_0(\Theta)$ ,  $p_0(\Theta_c) = p(\boldsymbol{\eta}_c|v_c)p_0(v_c)$  and  $C$  is the regularization parameter. Solving problem (7), we can get the posterior distribution

$$q(\Omega, \Theta) = \frac{p_0(\Omega, \Theta)p(\mathbf{X}|\Omega)\varphi(\mathbf{y}|\mathbf{Z}, \boldsymbol{\eta})}{\phi(\mathbf{X}, \mathbf{y})}, \tag{8}$$

where  $\phi(\mathbf{X}, \mathbf{y})$  is the normalization constant. In order to approximate  $q(\Omega, \Theta)$  we use variational approximate inference which is introduced in the Sect. 4.

<sup>1</sup> We use the shape-rate parameterization, i.e.,  $\alpha$  and  $\beta$  are the shape and rate parameter respectively

### 3.3 Multi-view regression with Bayesian M<sup>2</sup>SL

In this section, we present the max-margin latent subspace multi-view learning (BM<sup>2</sup>SMVL) for regression.

The Bayesian multi-view subspace learning model is the same with the classification model, the difference is the loss function. We use the  $\epsilon$ -insensitive loss for regression (Zhu et al. 2009), so we introduce

$$\varphi_R(\mathbf{y}|\mathbf{Z}, \boldsymbol{\eta}, \epsilon) = \prod_{n=1}^N \exp\{-2C_R \cdot \max(|y^{(n)} - \boldsymbol{\eta}^T \tilde{\mathbf{z}}^{(n)}| - \epsilon, 0)\}$$

as the pseudo-likelihood of the  $n$ -th data. Under the Bayesian framework, we impose a prior on  $\boldsymbol{\eta}$  as follows:

$$\begin{aligned} p_0(v) &= \Gamma(v|a_v, b_v) \\ p(\boldsymbol{\eta}|v) &= \mathcal{N}(\boldsymbol{\eta}|\mathbf{0}, v^{-1}\mathbf{I}_{(m+1)}). \end{aligned}$$

For simplify, let  $\Theta = \{\boldsymbol{\eta}, v, \epsilon\}$  for regression. Similar to the classification model, we can get the posterior distribution for regression as:

$$q(\Omega, \Theta) = \frac{p_0(\Omega, \Theta)p(\mathbf{X}|\Omega)\varphi_R(\mathbf{y}|\mathbf{Z}, \boldsymbol{\eta}, \epsilon)}{\phi(\mathbf{X}, \mathbf{y})}. \tag{9}$$

### 3.4 Online version of BM<sup>2</sup>SMVL

The goal of online learning is to minimize the cumulative loss for a certain prediction task from the sequentially arriving training samples. In this section, we present an online BM<sup>2</sup>SMVL (OBM<sup>2</sup>SMVL) based on the online Bayesian passive-aggressive (BayesPA) learning framework for Bayesian max-margin models (Shi and Zhu 2013).

Assuming we have already got the posterior  $q_t(\Omega, \Theta)$  at time  $t$ , when the new data  $(\mathbf{x}^{(t+1)}, y^{(t+1)})$  is coming, we need update the new posterior distribution  $q_{t+1}(\Omega, \Theta)$ . For simplify, we denote  $\mathbf{x}^{(t+1)} = \{\mathbf{x}_i^{(t+1)}\}_{i=1}^{N_v}$ . Generally, we define  $\omega$  as the parameterized model and  $\ell(\omega; \mathbf{x}^{(t+1)}, y^{(t+1)})$  as the loss for the new data  $(\mathbf{x}^{(t+1)}, y^{(t+1)})$ .

Insted of updating a point estimate of  $\omega$ , online BayesPA sequentially infers a new posterior distribution  $q_{t+1}(\omega)$ . It sequentially infers a new posterior distribution  $q_{t+1}(\omega)$  on the arrival of new data  $(\mathbf{x}^{(t+1)}, y^{(t+1)})$  by solving the following optimization problem:

$$\min_{q(\omega) \in \mathcal{P}} \text{KL}(q(\omega)\|q_t(\omega)) - \mathbb{E}_{q(\omega)}[\log p(\mathbf{x}^{(t+1)}|\omega)] + \ell(\omega; \mathbf{x}^{(t+1)}, y^{(t+1)}).$$

The online model includes three main updating rules. Firstly, we hope  $\text{KL}(q(\omega)\|q_t(\omega))$  is as small as possible. It means that  $q_{t+1}(\omega)$  is close to  $q_t(\omega)$ . Secondly, the likelihood of the new data  $\mathbb{E}_{q(\omega)}[\log p(\mathbf{x}^{(t+1)}|\omega)]$  is high enough. Thirdly, the loss of the new data  $\ell(\omega; \mathbf{x}^{(t+1)}, y^{(t+1)})$  is as small as possible. It means that the new model  $q_{t+1}(\omega)$  suffers little loss from the new data. Note that the latent variable  $\mathbf{z}$  for each data is also included in  $\omega$ , but the prior distribution of latent variable  $\mathbf{z}^{t+1}$  for the new arrival data  $(\mathbf{x}^{(t+1)}, y^{(t+1)})$  is not the posterior of  $\mathbf{z}^t$ , but  $p_0(\mathbf{z})$ . So the updating rule for latent variables  $\mathbf{z}$  is different from other parameters.

Considering that in reality, sometimes we can obtain several training samples during a short moment, so we can use them as a mini-batch to learn the model, which is effective in reducing the noise in data and cutting the time for online learning. Suppose that we have a mini-batch of

$M$  samples at time  $t$ , for simplicity, we use  $\mathbf{X}^{(t+1)} = \{\mathbf{x}^{(l)}\}$ ,  $\mathbf{y}^{(t+1)} = \{y^{(l)}\}$  ( $l = 1, 2, \dots, L$ ) to denote the mini-batch observed in time  $t + 1$ .

To introduce the online idea to the above multi-view classification  $\text{BM}^2\text{SMVL}$ , we let  $(\Omega, \Theta)$  denote  $\omega$  and  $\mathbf{y}^{(t+1)} = \{y_c^{(t+1)}\}_{c=1}^{N_c}$ . A new posterior distribution  $q_{t+1}(\Omega, \Theta)$  on the arrival of new data  $(\mathbf{X}^{(t+1)}, \mathbf{y}^{(t+1)})$  can be gotten by solving the following optimization problem:

$$\min_{q(\Omega, \Theta) \in \mathcal{P}} \text{KL}(q(\Omega, \Theta) \| q_t(\Omega, \Theta)) - \mathbb{E}_{q(\Omega, \Theta)}[\log p(\mathbf{X}^{(t+1)} | \Omega, \Theta)] + \ell(\Omega, \Theta; \mathbf{X}^{(t+1)}, \mathbf{y}^{(t+1)}).$$

As above, we introduce  $\varphi(\cdot)$  function to replace the hinge loss as the pseudo-likelihood. So the formula is replaced by:

$$\min_{q(\Omega, \Theta) \in \mathcal{P}} \text{KL}(q(\Omega, \Theta) \| q_t(\Omega, \Theta)) - \mathbb{E}_{q(\Omega)}[\log p(\mathbf{X}^{(t+1)} | \Omega)] - \mathbb{E}_{q(\Omega, \Theta)}[\log(\varphi(\mathbf{y}^{(t+1)} | \tilde{\mathbf{z}}, \boldsymbol{\eta}))].$$

Similar to Eq. (8), we can get the posterior distribution:

$$q_{t+1}(\Omega, \Theta) = \frac{q_t(\Omega, \Theta) p(\mathbf{X}^{(t+1)} | \Omega) \varphi(\mathbf{y}^{(t+1)} | \tilde{\mathbf{z}}, \boldsymbol{\eta})}{\phi(\mathbf{X}^{(t+1)}, \mathbf{y}^{(t+1)})},$$

where  $\phi(\mathbf{X}^{(t+1)}, \mathbf{y}^{(t+1)})$  is the normalization constant. Note that, the latent variable  $\mathbf{z}^t$  is unrelated to the new posterior, because the variable  $\mathbf{z}^{t+1}$ 's prior is  $p_0(\mathbf{z})$ . Let  $(\Omega, \Theta \setminus \mathbf{z}^t)$  denote all variables in  $\Omega$  and  $\Theta$  except  $\mathbf{z}^t$ , then we can further get

$$q_{t+1}(\Omega, \Theta) = \frac{q_t(\Omega, \Theta \setminus \mathbf{z}^t) p_0(\mathbf{z}) p(\mathbf{X}^{(t+1)} | \Omega) \varphi(\mathbf{y}^{(t+1)} | \tilde{\mathbf{z}}, \boldsymbol{\eta})}{\phi(\mathbf{X}^{(t+1)}, \mathbf{y}^{(t+1)})}. \tag{10}$$

The online  $\text{BM}^2\text{SMVL}$  for regression is similar to that for classification. So we can easily get the new posterior with the new arrival data for regression:

$$q_{t+1}(\Omega, \Theta) = \frac{q_t(\Omega, \Theta \setminus \mathbf{z}^t) p_0(\mathbf{z}) p(\mathbf{X}^{(t+1)} | \Omega) \varphi_R(\mathbf{y}^{(t+1)} | \tilde{\mathbf{z}}, \boldsymbol{\eta}, \epsilon)}{\phi(\mathbf{X}^{(t+1)}, \mathbf{y}^{(t+1)})}. \tag{11}$$

In order to approximate  $q_{t+1}(\Omega, \Theta)$  we use variational approximate inference which is introduced in Sect. 4.

### 4 Variational inference

Because the posterior is intractable to compute, we apply the variational inference method (Beal 2003) to approximate the posteriors in Eqs. (8) and (9) for  $\text{BM}^2\text{SMVL}$  and in Eqs. (10) and (11) for  $\text{OBM}^2\text{SMVL}$ . This method is much more efficient than sampling based methods (Gilks 2005).

#### 4.1 Data augmentation

Since the pseudo-likelihood function  $\varphi(\cdot)$  involves a max operator which is difficult and inefficient for posterior inference. We re-express the pseudo-likelihood function into the integration of a function with augmented variable based on the data augmentation idea (Polson



and Scott 2011; Zhu et al. 2014). For classification in BM<sup>2</sup>SMVL, we replace the pseudo-likelihood  $\varphi(\cdot)$  with  $\varphi(y_c^{(n)}|\bar{\mathbf{z}}^{(n)}, \boldsymbol{\eta}_c)$ :

$$\varphi(\cdot) = \int_0^\infty \frac{\exp\left\{\frac{-1}{2\lambda_c^{(n)}}[\lambda_c^{(n)} + C(1 - y_c^{(n)}\boldsymbol{\eta}_c^T\bar{\mathbf{z}}^{(n)})]^2\right\} d\lambda_c^{(n)}}{\sqrt{2\pi\lambda_c^{(n)}}},$$

where  $\lambda_c^{(n)}$  ( $n = 1, \dots, N$ ) are the auxiliary variables introduced to deal with the max function. Let  $\boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_N]^T$ , then the augmented pseudo likelihood of  $\mathbf{y}$ ,  $\boldsymbol{\lambda}$  can be expressed as:

$$\varphi(\mathbf{y}, \boldsymbol{\lambda}|\mathbf{Z}, \boldsymbol{\eta}) = \prod_{n=1}^N \prod_{c=1}^{N_c} \frac{\exp\left\{\frac{-1}{2\lambda_c^{(n)}}[\lambda_c^{(n)} + C(1 - y_c^{(n)}\boldsymbol{\eta}_c^T\bar{\mathbf{z}}^{(n)})]^2\right\}}{\sqrt{2\pi\lambda_c^{(n)}}}.$$

Similarly, we introduce the augmented variable to the pseudo-likelihood function  $\varphi(\cdot)$  for classification in OBM<sup>2</sup>SMVL:

$$\varphi(\mathbf{y}^{(t+1)}, \boldsymbol{\lambda}^{(t+1)}|\mathbf{Z}, \boldsymbol{\eta}) = \prod_{l=1}^L \prod_{c=1}^{N_c} \frac{\exp\left\{\frac{-1}{2\lambda_c^{(l)}}[\lambda_c^{(l)} + C(1 - y_c^{(l)}\boldsymbol{\eta}_c^T\bar{\mathbf{z}}^{(l)})]^2\right\}}{\sqrt{2\pi\lambda_c^{(l)}}}.$$

For regression, the  $\epsilon$ -insensitive loss  $\varphi_R(y^{(n)}|\mathbf{z}^{(n)}, \boldsymbol{\eta})$  can be represented as a dual scale mixture of Gaussian distributions based on data augmentation:

$$\begin{aligned} \varphi_R(\cdot) &= \int_0^\infty \exp\left\{\frac{-[\lambda^{(n)} + C_R(y^{(n)} - \boldsymbol{\eta}^T\bar{\mathbf{z}}^{(n)} - \epsilon)]^2}{2\lambda^{(n)}}\right\} \cdot \frac{d\lambda^{(n)}}{\sqrt{2\pi\lambda^{(n)}}} \\ &\times \int_0^\infty \exp\left\{\frac{-[\theta^{(n)} + C_R(\boldsymbol{\eta}^T\bar{\mathbf{z}}^{(n)} - y^{(n)} - \epsilon)]^2}{2\theta^{(n)}}\right\} \cdot \frac{d\theta^{(n)}}{\sqrt{2\pi\theta^{(n)}}}, \end{aligned}$$

where  $\lambda^{(n)}$  and  $\theta^{(n)}$  ( $n = 1, \dots, N$ ) are the auxiliary variables introduced to deal with the max function. Let  $\boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_N]^T$  and  $\boldsymbol{\theta} = [\theta_1, \dots, \theta_N]^T$ , then the augmented pseudo likelihood  $\varphi_R(\mathbf{y}, \boldsymbol{\lambda}, \boldsymbol{\theta}|\mathbf{Z}, \boldsymbol{\eta}, \epsilon)$  of  $\mathbf{y}$ ,  $\boldsymbol{\lambda}$ ,  $\boldsymbol{\theta}$  can be expressed as:

$$\begin{aligned} \varphi_R(\cdot) &= \prod_{n=1}^N \frac{\exp\left\{\frac{1}{-2\lambda^{(n)}}[\lambda^{(n)} + C_R(y^{(n)} - \boldsymbol{\eta}^T\bar{\mathbf{z}}^{(n)} - \epsilon)]^2\right\}}{\sqrt{2\pi\lambda^{(n)}}} \\ &\times \frac{\exp\left\{\frac{1}{-2\theta^{(n)}}[\theta^{(n)} + C_R(\boldsymbol{\eta}^T\bar{\mathbf{z}}^{(n)} - y^{(n)} - \epsilon)]^2\right\}}{\sqrt{2\pi\theta^{(n)}}}. \end{aligned}$$

Similarly, we introduce the augmented variable to the pseudo-likelihood function  $\varphi_R(\mathbf{y}^{(t+1)}, \boldsymbol{\lambda}^{(t+1)}, \boldsymbol{\theta}^{(t+1)}|\mathbf{Z}, \boldsymbol{\eta}, \epsilon)$  for regression in OBM<sup>2</sup>SMVL:

$$\begin{aligned} \varphi_R^{(t+1)}(\cdot) &= \prod_{l=1}^L \frac{\exp\left\{\frac{1}{-2\lambda^{(l)}}[\lambda^{(l)} + C_R(y^{(n)} - \boldsymbol{\eta}^T\bar{\mathbf{z}}^{(n)} - \epsilon)]^2\right\}}{\sqrt{2\pi\lambda^{(l)}}} \\ &\times \frac{\exp\left\{\frac{1}{-2\theta^{(l)}}[\theta^{(n)} + C_R(\boldsymbol{\eta}^T\bar{\mathbf{z}}^{(n)} - y^{(n)} - \epsilon)]^2\right\}}{\sqrt{2\pi\theta^{(l)}}}. \end{aligned}$$

## 4.2 Variational approximate inference

Next, we apply the mean-field variational method to approximating the posterior distributions.

### 4.2.1 Variational inference in BM<sup>2</sup>SMVL

In this section, we take the classification model as an example. Firstly, we define a family of factorized but free-form variational distributions:

$$V(\Omega, \Theta, \lambda) = V(\mu) V(\mathbf{W}) V(\alpha) V(\phi) V(\mathbf{Z}) V(\eta) V(\lambda) V(v).$$

The main idea of variational Bayesian inference is that we need to minimize the KL divergence  $KL(V(\Omega, \Theta, \lambda) \| q(\Omega, \Theta, \lambda))$  between the approximating distribution and the target posterior. Next, we initialize the distributions of  $V(\Omega, \Theta, \lambda)$ . Then we iteratively update each parameter of our model by fixing other parameters as the current estimates. Now, we give the joint distribution of data and parameters:

$$p(\Omega, \Theta, \lambda, \mathbf{X}, \mathbf{y}) = p_0(\mu) p(\mathbf{W} | \mu) p_0(\alpha) p_0(\phi) p_0(\mathbf{Z}) p(\eta | v) \cdot p_0(v) p(\mathbf{X} | \mu, \mathbf{W}, \phi, \mathbf{Z}) \varphi(\mathbf{y}, \lambda | \mathbf{Z}, \eta).$$

It can be shown that when keeping all other factors fixed, the optimal distribution  $V^*(\lambda)$  satisfies

$$V^*(\lambda) \propto \exp\{\mathbb{E}_{-\lambda}[\log p(\Omega, \Theta, \lambda, \mathbf{X}, \mathbf{y})]\},$$

where  $\mathbb{E}_{-\lambda}$  denotes the expectation with respect to  $V(\Omega, \Theta, \lambda)$  over all variables except for  $\lambda$ . Then we can get the updating formula for  $\mathbb{E}_{-\lambda}$ :

$$\begin{aligned} V^*(\lambda) &= \prod_{c=1}^{N_c} \prod_{n=1}^N \mathcal{GIG}(\lambda_c^{(n)} | \frac{1}{2}, 1, \chi_c^{(n)}) \\ \chi_c^{(n)} &= C^2 \langle (1 - y_c^{(n)} \eta_c^T \tilde{\mathbf{z}}^{(n)})^2 \rangle \\ \mathbb{E}_{\lambda^{(n)}}[(\lambda^{(n)})^{-1}] &= 1/\sqrt{\chi_{\lambda^{(n)}}}, \end{aligned} \tag{12}$$

where  $\langle \cdot \rangle$  represents the expectation,  $\mathcal{GIG}(\cdot)$  is the generalized inverse Gaussian distribution. Similarly, we can get the updating formulas for all other factors. And the main steps for mean-field methods can be found in.<sup>2</sup> Since they are tedious and easy to derive, here we only provide the equations for  $\mathbf{Z}$ , other updating formulas can be found in the ‘‘Appendix’’.

$$\begin{aligned} V^*(\mathbf{Z}) &= \prod_{n=1}^N \mathcal{N}(\mathbf{z}^{(n)} | \mu_{\mathbf{z}}^{(n)}, \Sigma_{\mathbf{z}}^{(n)}) \\ \Sigma_{\mathbf{z}}^{(n)} &= \left\{ C^2 \sum_{c=1}^{N_c} \langle \tilde{\eta}_c \tilde{\eta}_c^T \rangle (\lambda_c^{(n)})^{-1} + \mathbf{I}_m \right. \\ &\quad \left. + \sum_{i=1}^{N_v} \langle \phi_i \rangle \langle \mathbf{W}_i^T \mathbf{W}_i \rangle \right\}^{-1} \\ \mu_{\mathbf{z}}^{(n)} &= \Sigma_{\mathbf{z}}^{(n)} \left\{ \sum_{i=1}^{N_v} \langle \phi_i \rangle \langle \mathbf{W}_i^T \rangle (\mathbf{x}_i^{(n)} - \langle \mu_i \rangle) \right\} \end{aligned}$$

<sup>2</sup> [https://en.wikipedia.org/wiki/Variational\\_Bayesian\\_methods](https://en.wikipedia.org/wiki/Variational_Bayesian_methods).

$$\begin{aligned} & + \{C(1 + C((\lambda^{(n)})^{-1}))y^{(n)}\langle \tilde{\boldsymbol{\eta}} \rangle \\ & - C^2\langle (\lambda^{(n)})^{-1} \rangle \langle \eta_{(m+1)} \tilde{\boldsymbol{\eta}} \rangle \} \\ \mathbb{E}_{\mathbf{Z}^{(n)}}[\mathbf{Z}^{(n)}] & = \boldsymbol{\mu}_{\mathbf{Z}}^{(n)}, \end{aligned} \tag{13}$$

where  $\tilde{\boldsymbol{\eta}}$  denotes the first  $m$  dimensions of  $\boldsymbol{\eta}$ , i.e.,  $\boldsymbol{\eta} = [\tilde{\boldsymbol{\eta}}, \eta_{(m+1)}]$ .

For regression, only the variables  $\mathbf{Z}$ ,  $\boldsymbol{\eta}$ ,  $\boldsymbol{\lambda}$  and  $\boldsymbol{\theta}$  are different from the classification model. The updating formulas for  $\mathbf{Z}$ ,  $\boldsymbol{\eta}$ ,  $\boldsymbol{\lambda}$  and  $\boldsymbol{\theta}$  can also be found in the ‘‘Appendix’’. We summarize the proposed BM<sup>2</sup>SMVL model in Algorithm 1.

---

**Algorithm 1** BM<sup>2</sup>SMVL

---

**Input:**

the multi-view data  $\{\mathbf{X}_i\}_{i=1}^{N_v}$ , the response vector  $\mathbf{y}$ , the subspace dimension  $m$  and the maximal number of iterations  $maxIter$ .

**Output:**

$\mathbb{E}_{\boldsymbol{\eta}}[\boldsymbol{\eta}]$ ,  $\mathbb{E}_{\boldsymbol{\mu}_i}[\boldsymbol{\mu}_i]$ ,  $\mathbb{E}_{\mathbf{W}_i}[\mathbf{W}_i]$ ,  $\mathbb{E}_{\phi_i}[\phi_i]$

**Method:**

- 1: Initialize  $\mathbb{E}_v[v]$ ,  $\mathbb{E}_{\boldsymbol{\lambda}}[\boldsymbol{\lambda}]$ ,  $\mathbb{E}_{\boldsymbol{\eta}}[\boldsymbol{\eta}]$ ,  $\mathbb{E}_{\boldsymbol{\mu}_i}[\boldsymbol{\mu}_i]$ ,  $\mathbb{E}_{\mathbf{W}_i}[\mathbf{W}_i]$ ,  $\mathbb{E}_{\phi_i}[\phi_i]$ ,  $\mathbb{E}_{\boldsymbol{\alpha}_i}[\boldsymbol{\alpha}_i]$  ( $i = 1, \dots, N_v$ ),  $\mathbb{E}_{\mathbf{Z}}[\mathbf{Z}]$ ,  $\mathbb{E}_{\boldsymbol{\theta}}[\boldsymbol{\theta}]$  (only for regression);
  - 2: **for**  $iter = 1$  to  $maxIter$  **do**
  - 3:   Update  $\mathbb{E}_{\boldsymbol{\mu}_i}[\boldsymbol{\mu}_i]$ ,  $\mathbb{E}_{\mathbf{W}_i}[\mathbf{W}_i]$ ,  $\mathbb{E}_{\phi_i}[\phi_i]$ ,  $\mathbb{E}_{\boldsymbol{\alpha}_i}[\boldsymbol{\alpha}_i]$  and  $\mathbb{E}_v[v]$  according to Eq.(18), Eq.(17), Eq.(16), Eq.(19) and Eq.(20) respectively;
  - 4:   **if** Task is classification **then**
  - 5:     Update  $\mathbb{E}_{\mathbf{Z}}[\mathbf{Z}]$ ,  $\mathbb{E}_{\boldsymbol{\lambda}}[\boldsymbol{\lambda}]$  and  $\mathbb{E}_{\boldsymbol{\eta}}[\boldsymbol{\eta}]$  according to Eq.(13), Eq.(12), Eq.(21) respectively for classification
  - 6:   **else** {Task is regression}
  - 7:     Update  $\mathbb{E}_{\mathbf{Z}}[\mathbf{Z}]$ ,  $\mathbb{E}_{\boldsymbol{\lambda}}[\boldsymbol{\lambda}]$ ,  $\mathbb{E}_{\boldsymbol{\theta}}[\boldsymbol{\theta}]$  and  $\mathbb{E}_{\boldsymbol{\eta}}[\boldsymbol{\eta}]$  according to Eq.(22), Eq.(23), Eq.(24), Eq.(25) respectively for regression
  - 8:   **end if**
  - 9: **end for**
  - 10: **return**  $\mathbb{E}_{\boldsymbol{\eta}}[\boldsymbol{\eta}]$ ,  $\mathbb{E}_{\boldsymbol{\mu}_i}[\boldsymbol{\mu}_i]$ ,  $\mathbb{E}_{\mathbf{W}_i}[\mathbf{W}_i]$ ,  $\mathbb{E}_{\phi_i}[\phi_i]$
- 

**4.2.2 Variational inference in OBM<sup>2</sup>SMVL**

Now, we use variational inference to approximate  $q_{t+1}(\Omega, \Theta)$  in OBM<sup>2</sup>SMVL model. It is very similar to the posterior  $q(\Omega, \Theta)$ . Firstly, we give the joint distribution of data and parameters:

$$\begin{aligned} p(\Omega, \Theta, \boldsymbol{\lambda}^{(t+1)}, \mathbf{x}^{(t+1)}, \mathbf{y}^{(t+1)}) & = p_0(\boldsymbol{\mu})p(\mathbf{W}|\boldsymbol{\mu})p_0(\boldsymbol{\alpha})p_0(\phi)p_0(\mathbf{Z})p(\boldsymbol{\eta}|v)p_0(v) \\ & \cdot p(\mathbf{x}^{(t+1)}|\boldsymbol{\mu}, \mathbf{W}, \phi, \mathbf{Z})\varphi(\mathbf{y}^{(t+1)}, \boldsymbol{\lambda}^{(t+1)}|\mathbf{z}, \boldsymbol{\eta}). \end{aligned}$$

It can be shown that when keeping all other factors fixed, the optimal distribution  $V^*(\boldsymbol{\lambda}^{(t+1)})$  satisfies

$$V^*(\boldsymbol{\lambda}^{(t+1)}) \propto \exp\{\mathbb{E}_{-\boldsymbol{\lambda}^{(t+1)}}[\log p(\Omega, \Theta, \boldsymbol{\lambda}, \mathbf{x}^{(t+1)}, \mathbf{y}^{(t+1)})]\},$$

where  $\mathbb{E}_{-\boldsymbol{\lambda}^{(t+1)}}$  denotes the expectation with respect to  $V(\Omega, \Theta, \boldsymbol{\lambda}^{(t+1)})$  over all variables except for  $\boldsymbol{\lambda}^{(t+1)}$ . Then we can get the updating formula for  $\mathbb{E}_{-\boldsymbol{\lambda}^{(t+1)}}$ :

$$V^*(\boldsymbol{\lambda}^{(t+1)}) = \prod_{l=1}^L \prod_{c=1}^{N_c} \mathcal{GIG} \left( \lambda_c^{(l)} \middle| \frac{1}{2}, 1, \chi_c^{(l)} \right)$$

$$\begin{aligned} \chi_{\lambda_c^{(l)}} &= C^2 \langle (1 - y_c^{(l)} (\boldsymbol{\eta}_c^{(t+1)})^T \tilde{\mathbf{z}}^{(l)})^2 \rangle. \\ \mathbb{E}_{\lambda_c^{(l)}} [(\lambda_c^{(l)})^{-1}] &= 1 / \sqrt{\chi_{\lambda_c^{(l)}}}. \end{aligned} \tag{14}$$

Similarly, we can get the updating formulas for all other factors. Since they are tedious and easy to derive, here we only provide the equations for  $\mathbf{Z}^{(t+1)}$ , other updating formulas can be found in the ‘‘Appendix’’.

$$\begin{aligned} V^*(\mathbf{Z}^{(t+1)}) &= \prod_{l=1}^L \mathcal{N}(\mathbf{z}^{(l)} | \mu_{\mathbf{z}}^{(l)}, \Sigma_{\mathbf{z}}^{(l)}) \\ \Sigma_{\mathbf{z}}^{(l)} &= \left\{ C^2 \sum_{c=1}^{N_c} \langle \tilde{\boldsymbol{\eta}}_c^{(t+1)} (\tilde{\boldsymbol{\eta}}_c^{(t+1)})^T \rangle \langle \lambda_c^{(l)-1} \rangle + \mathbf{I}_m \right. \\ &\quad \left. + \sum_{i=1}^{N_v} \langle \phi_i^{(t+1)} \rangle \langle (\mathbf{W}_i^{(t+1)})^T \mathbf{W}_i^{(t+1)} \rangle \right\}^{-1} \\ \mu_{\mathbf{z}}^{(l)} &= \Sigma_{\mathbf{z}}^{(l)} \left\{ \sum_{i=1}^{N_v} \langle \phi_i^{(t+1)} \rangle \langle (\mathbf{W}_i^{(t+1)})^T \rangle (\mathbf{x}_i^{(l)} - \langle \boldsymbol{\mu}_i^{(t+1)} \rangle) \right. \\ &\quad \left. + \{ C(1 + C \langle (\lambda^{(l)})^{-1} \rangle) y^{(l)} \langle \tilde{\boldsymbol{\eta}}^{(t+1)} \rangle \right. \\ &\quad \left. - C^2 \langle (\lambda^{(l)})^{-1} \rangle \langle \boldsymbol{\eta}_{(m+1)}^{(t+1)} \tilde{\boldsymbol{\eta}}^{(t+1)} \rangle \right\} \\ \mathbb{E}_{\mathbf{z}^{(l)}} [\mathbf{z}^{(l)}] &= \mu_{\mathbf{z}}^{(l)}, \end{aligned} \tag{15}$$

where  $\tilde{\boldsymbol{\eta}}$  denotes the first  $m$  dimensions of  $\boldsymbol{\eta}$ , i.e.,  $\boldsymbol{\eta}_c = [\tilde{\boldsymbol{\eta}}_c, \eta_c^{(m+1)}]$ .

For regression, only the variables  $\mathbf{Z}^{(t+1)}$ ,  $\boldsymbol{\eta}^{(t+1)}$ ,  $\lambda^{(t+1)}$  and  $\boldsymbol{\theta}^{(t+1)}$  are different from the classification model, the updating formulas for  $\mathbf{Z}^{(t+1)}$ ,  $\boldsymbol{\eta}^{(t+1)}$ ,  $\lambda^{(t+1)}$  and  $\boldsymbol{\theta}^{(t+1)}$  can be found in the ‘‘Appendix’’. A full description of the proposed OBM<sup>2</sup>SMVL model is given in Algorithm 2. Here we use  $T$  to represent the total number of mini-batches. So the total number of training data is  $N = T \times L$ . Obviously, by limiting  $L$  to 1, the algorithm can handle the case we first assumed, i.e., learning from samples one-by-one.

### 4.3 Prediction on unseen data

Suppose we have a set of test data that is unseen during the model training phase. To apply our models learned above, we have to first project the new data to the same low-dimensional feature space as that for training data. Given the optimal variational distributions  $V^*(\boldsymbol{\eta})$ ,  $V^*(\mathbf{W}_i)$ ,  $V^*(\boldsymbol{\mu}_i)$ , and  $V^*(\phi_i)$  learned in the training phase, we use a single step variational method to approximate the posterior latent representation  $p(\mathbf{z}_{new} | \mathbf{x}_{new})$  for test data  $\mathbf{x}_{new}$ :

$$\begin{aligned} V^*(\mathbf{z}_{new}) &= \mathcal{N}(\mathbf{z}_{new} | \mu_{\mathbf{z}}^{new}, \Sigma_{\mathbf{z}}^{new}) \\ \Sigma_{\mathbf{z}}^{new} &= \left\{ \mathbf{I}_m + \sum_{i=1}^{N_v} \mathbb{E}_{\phi_i} [\phi_i] \mathbb{E}_{\mathbf{W}_i} [\mathbf{W}_i^T \mathbf{W}_i] \right\}^{-1} \\ \mu_{\mathbf{z}}^{new} &= \Sigma_{\mathbf{z}}^{new} \left\{ \sum_{i=1}^{N_v} \mathbb{E}_{\phi_i} [\phi_i] \mathbb{E}_{\mathbf{W}_i} [\mathbf{W}_i^T] (\mathbf{x}_i^{new} - \mathbb{E}_{\boldsymbol{\mu}_i} [\boldsymbol{\mu}_i]) \right\}, \end{aligned}$$

**Algorithm 2** OBM<sup>2</sup>SMVL

**Input:**

the multi-view data  $\{\mathbf{X}_i\}_{i=1}^{N_v}$ , the response vector  $\mathbf{y}$ , the subspace dimension  $m$ , the total number of mini-batches  $T (N = T \times L)$  and the maximal number of iterations  $maxIter$ .

**Output:**

$\mathbb{E}_\eta^{(T)}[\eta], \mathbb{E}_{\mu_i}^{(T)}[\mu_i], \mathbb{E}_{\mathbf{W}_i}^{(T)}[\mathbf{W}_i], \mathbb{E}_{\phi_i}^{(T)}[\phi_i]$

**Method:**

- 1: Initialize  $\mathbb{E}_v[v], \mathbb{E}_\lambda[\lambda], \mathbb{E}_\eta[\eta], \mathbb{E}_{\mu_i}[\mu_i], \mathbb{E}_{\mathbf{W}_i}[\mathbf{W}_i], \mathbb{E}_{\phi_i}[\phi_i], \mathbb{E}_{\alpha_i}[\alpha_i] (i = 1, \dots, N_v), \mathbb{E}_Z[\mathbf{Z}], \mathbb{E}_\theta[\theta]$  (only for regression);
- 2: **for**  $t = 0 \rightarrow T - 1$  **do**
- 3:  $\mathbf{X}_{mini-batch} = \{\mathbf{X}_i^{(t+1)}\}_{i=1}^{N_v}$   
 $\mathbf{y}_{mini-batch} = \mathbf{y}^{(t+1)}$
- 4: **for**  $iter = 1$  to  $maxIter$  **do**
- 5: Update  $\mathbb{E}_{\mu_i}^{(t+1)}[\mu_i], \mathbb{E}_{\mathbf{W}_i}^{(t+1)}[\mathbf{W}_i]$  and  $\mathbb{E}_{\phi_i}^{(t+1)}[\phi_i]$  by Eq.(28), Eq.(27) and Eq.(26) respectively;
- 6: **if** Task is classification **then**
- 7: Update  $\mathbb{E}_Z^{(t+1)}[\mathbf{Z}], \mathbb{E}_\lambda^{(t+1)}[\lambda]$  and  $\mathbb{E}_\eta^{(t+1)}[\eta]$  according to Eq.(15), Eq.(14), Eq.(29) respectively for classification
- 8: **else** {Task is regression}
- 9: Update  $\mathbb{E}_Z^{(t+1)}[\mathbf{Z}], \mathbb{E}_\lambda^{(t+1)}[\lambda], \mathbb{E}_\theta^{(t+1)}[\theta]$  and  $\mathbb{E}_\eta^{(t+1)}[\eta]$  according to Eq.(33), Eq.(30), Eq.(31), Eq.(32) respectively for regression
- 10: **end if**
- 11: **end for**
- 12: **end for**
- 13: **return**  $\mathbb{E}_\eta^{(T)}[\eta], \mathbb{E}_{\mu_i}^{(T)}[\mu_i], \mathbb{E}_{\mathbf{W}_i}^{(T)}[\mathbf{W}_i], \mathbb{E}_{\phi_i}^{(T)}[\phi_i]$

where the expectations are taken over the optimal variational distributions of  $\eta, \mathbf{W}_i, \mu_i$  and  $\phi_i$ .

Then with the optimal variational approximation  $V^*(\eta)$  for the posterior distribution of classification parameter  $\eta$ , we can predict the class label of  $\mathbf{x}^{new}$  for classification by

$$\begin{aligned} \tilde{\mu}_z^{new} &= [(\mu_z^{new})^T, 1]^T \\ y_{new} &= \max_c (\mathbb{E}_{\eta_c, \mathbf{z}_{new}} [\eta_c^T \tilde{\mathbf{z}}_{new}]) \\ &= \max_c (\mu_{\eta_c}^T \tilde{\mu}_z^{new}). \end{aligned}$$

For regression, we can directly predict the response value:

$$\begin{aligned} y_{new} &= \mathbb{E}_{\eta, \mathbf{z}_{new}} [\eta^T \tilde{\mathbf{z}}_{new}] \\ &= \mu_\eta^T \tilde{\mu}_z^{new}. \end{aligned}$$

**4.4 Computational complexity**

For each iteration of parameter updating in our batch learning BM<sup>2</sup>SMVL, we need  $O(NN_v \bar{d}m^2)$  computation, where  $\bar{d}$  is the average dimension of all  $N_v$  views. The most computation load is spent on the calculation of  $\Sigma_z^{(n)}, n = 1, \dots, N$  where the matrix multiplication  $\langle \mathbf{W}_i^T \mathbf{W}_i \rangle$  consumes  $d_i m^2$  computation. And each iteration of parameter updating in our online learning OBM<sup>2</sup>SMVL consumes  $O(LN_v \bar{d}m^2)$  with a batch of  $L$  new arrival data.

**Table 1** Statistics of the multiclass data sets

Datasets	Trecvid	Washington	Cornell	Texas	Wisconsin	News4Gv	NUS-WIDE
#size	1078	230	195	187	265	1500	21,935
#class	5	5	5	5	5	3	3
#D <sub>1</sub>	1894	1703	1703	1703	1703	6783	64
#D <sub>2</sub>	165	230	195	187	265	6307	144
#D <sub>3</sub>	–	–	–	–	–	7717	73
#D <sub>4</sub>	–	–	–	–	–	9336	128
#D <sub>5</sub>	–	–	–	–	–	–	225

## 5 Experiments

We evaluate the proposed batch learning model  $BM^2SMVL$  and online learning model  $OBM^2SMVL$  on various classification and regression tasks. For regression, we use the root-mean-square error (RMSE) to evaluate the results. RMSE is formulated as follows:

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N_{test}} (\hat{y}^{(n)} - y^{(n)})^2}{N_{test}}},$$

where  $y^{(n)}$  is the ground truth of the  $n$ -th sample,  $\hat{y}^{(n)}$  is the corresponding predicted value, and  $N_{test}$  is the total number of the testing samples.

### 5.1 Real data sets

There are seven data sets for classification tasks and three data sets for regression tasks in our experiments. Trecvid contains 1078 manually labeled video shots that belong to five categories (Chen et al. 2012). And each shot is represented by a 1894-dim binary vector of text features and a 165-dim vector of HSV color histogram. WebKB data set has two views, including the content features of the web pages and the link features exploited from the link structures. This data set consists of 877 web pages from computer science departments in four universities, i.e., Cornell, Washington, Wisconsin and Texas. And each university has five document classes, i.e., course, faculty, student, project and staff. We select the web pages from these four universities as our experimental data.<sup>3</sup> These four data sets have five classes with two views. 20Newsgroups data set is widely used for classification. This data set has approximately 20,000 newsgroup documents, which are divided into 20 categories. We follow the way in Long et al. (2008) to construct multi-view learning problems. The NUS-WIDE dataset is a subset selected from Chua et al. (2009). NUS-WIDE dataset contains 21,935 web images that belongs to three categories ('water', 'vehicle', 'flowers'). Each image includes six types of low-level features (64-D color histogram, 144-D color correlogram, 73-D edge direction histogram, 128-D wavelet texture, 225-D block-wise color moments). We use the tf-idf weighting scheme to represent the document, and the document frequency with the value of 5 is adopted to cut down the number of word features. The details of these data sets are shown in Table 1.

<sup>3</sup> <http://www-2.cs.cmu.edu/~webkb/>.

**Table 2** Statistics of the regression data sets

Datasets	HotelReview	MovieLens	CT-Image
#size	5000	502	53,500
#D <sub>1</sub>	12,000	445	240
#D <sub>2</sub>	14	498	144
#D <sub>3</sub>	–	51	–

The hotel review dataset<sup>4</sup> consists of 5000 hotel reviews randomly collected from TripAdvisor.<sup>5</sup> Each review document is associated with two-view features (i.e., 12,000-dim bag-of-word features and 14-dim contextual features) as well as a global rating score which ranks from 1 to 5. In our experiment, we predict the global rating scores for reviews.

Another regression task for rating prediction is studied on this MovieLens dataset by following the way in Lu et al. (2017). Each rating in this dataset has three views, i.e., users, movies and tags. The user view consists of binary feature vectors for user ids, and thus for each rating there is only one non-zero feature in the user view, i.e., the associated user id; the same for the movie view. The tags of the movies are used for the tag view.

The data CT-Image<sup>6</sup> was retrieved from a set of 53,500 CT images from 74 different patients (43 male, 31 female). Each CT slice is described by two histograms in polar space. The first histogram (240-dim) describes the location of bone structures in the image and the second histogram (144-dim) describes the location of air inclusions inside of the body. The predicting values are the relative location of the image on the axial axis which are in the range (0-180) where 0 denotes the top of the head and 180 denotes the soles of the feet. The details of these regression data sets are shown in Table 2.

## 5.2 Competitors

We compare our classification model with the following five competitors:

- VMRML (Quang et al. 2013): it is a vector-valued manifold regularization multi-view learning model. The regularization parameters are set as the default value in their paper, and we choose the best parameter  $\sigma$  for ‘*rbf*’ from  $10^{[-5:5]}$  by fivefold cross-validation in each data set;
- MVMED (Sun and Chao 2013): it presents a multi-view maximum entropy discrimination model. We use the model with one-VS-rest strategy for multiclass problem. According to the paper, we choose the best parameter from  $2^{[-5:5]}$  by executing fivefold cross-validation for each data set;
- MMH (Chen et al. 2012): it is a large-margin predictive latent subspace learning method for multi-view data. Based on the parameters given in its code,<sup>7</sup> we tune the four parameters carefully to choose the best parameters for each data set;
- SVM-FULL: it concatenates all views to form a new single view, and applies SVM for classification. We choose the linear kernel and execute fivefold cross-validation on training sets to decide the cost parameter  $c$  from  $10^{[-3:3]}$ ;

<sup>4</sup> <http://bigml.cs.tsinghua.edu.cn/~ningchen/data.htm>.

<sup>5</sup> <http://www.tripadvisor.com>.

<sup>6</sup> <http://archive.ics.uci.edu/ml/datasets>.

<sup>7</sup> <http://bigml.cs.tsinghua.edu.cn/~ningchen/MMH.htm>.

- OPMV (Zhu et al. 2015): it is an online multi-view learning. According to the paper, the learning rate parameter are chose from  $2^{[-8:8]}$ , the regularization parameter are chose from  $1e^{[-16:0]}$ , and the penalty parameters is pre-defined as 1. The parameters are set according to the above rules.

And the compared regression models are as follows:

- CoR-LS (Lan et al. 2016): it is a co-regularized least square regression model (CoR-LS) for multi-view data;
- LCFS (Wang et al. 2013): it unifies coupled linear regressions,  $\ell_{21}$ -norms and trace norm into a generic minimization formulation so that subspace learning and coupled feature selection can be performed simultaneously;
- SVR-FULL: it concatenates all views to form a new single view, and applies SVR<sup>8</sup> for regression. We use fivefold cross-validation on training sets to decide the regularization parameter  $c$  from  $10^{[-3:3]}$  and the precision parameter  $\epsilon$  from  $10^{[-1:1]}$ ;
- OSVR-FULL: it is the online SVR method. We concatenate all views to form a new single view and applies OSVR. The code provided by Parrella<sup>9</sup> for OSVR. For each data set, we use fivefold cross-validation on training sets to decide the regularization parameter  $c$  from  $10^{[-3:3]}$  and the precision parameter  $\epsilon$  from  $10^{[-1:1]}$ .

### 5.3 Parameter setting

In our batch learning, the regularization parameter  $C$  is chosen from the integer set  $\{1, 2, 3\}$  and the subspace dimension  $m$  from the integer set  $\{20, 30, 50\}$  for each data set by performing fivefold cross validation on training data. While in our online learning, the regularization parameter  $C$  is chosen from the integer set  $\{1, 5, 15\}$  and the subspace dimension  $m$  from the integer set  $\{30, 50, 70\}$ . For the hyperparameters, both our batch and online learning are set as the same, i.e.,  $a_\alpha = b_\alpha = 1e-3$ ,  $a_\phi = 1e-2$ ,  $a_\nu = 1e-1$ ,  $b_\phi = b_\nu = \beta = 1e-5$ . And we set the maximum iterations to 200.

For regression task, the regularization parameter  $C$  is chosen from the integer set  $\{2^{(-5)}, \dots, 2^{(10)}\}$  and the subspace dimension  $m$  from the integer set  $\{50, 100, 150\}$  for each data set by performing fivefold cross validation on training data. The hyperparameters are set as the same, i.e.,  $a_\alpha = b_\alpha = 1e-3$ ,  $a_\phi = 1e-2$ ,  $a_\nu = 1e-1$ ,  $b_\phi = b_\nu = \beta = 1e-5$ ,  $\epsilon = 0.01$ . And we set the maximum iterations to 20 for each mini-batch.

### 5.4 Experimental results

Since a normal prior with zero mean is imposed on the observation data, we normalize the observation data to have zero mean and unit variance. In batch learning experiments, the results of all models on all data sets are averaged over 20 independent runs. We adopt two evaluating metrics accuracy and F1 score for classification tasks. So the results about accuracy are shown in Table 3 and the results about F1 score are shown in Table 4. The ratio sampled for training data is 0.5 in the six data sets Trecvid, Washington, Cornell, Texas, Wisconsin, NUS-WIDE and 0.05 in News4Gv. Since MMH can only deal with two views in its code<sup>10</sup> so its result is missing for News4Gv and NUS-WIDE in Table 3. And it provides a software

<sup>8</sup> <https://www.csie.ntu.edu.tw/~cjlin/liblinear/>.

<sup>9</sup> <http://onlinesvr.altervista.org/>.

<sup>10</sup> <http://bigml.cs.tsinghua.edu.cn/~ningchen/MMH.htm>.



**Table 3** Batch and online learning comparisons on multiclass data sets

	Washington	Cornell	Texas	Wisconsin	News4Gv	NUS-WIDE	Trevid
MMH	80.98 ± 2.94	74.01 ± 0.19	83.86 ± 3.00	87.46 ± 2.62	-	-	83.37 ± 3.91
MVMED	73.86 ± 2.78	72.27 ± 2.97	78.37 ± 4.35	84.73 ± 2.03	94.26 ± 0.83	N/A	78.02 ± 1.34
VMRML	79.44 ± 2.61	76.96 ± 4.03	83.70 ± 2.97	85.91 ± 1.94	93.34 ± 0.98	81.39 ± 0.20	79.59 ± 1.34
SVM-FULL	82.91 ± 3.33	76.14 ± 2.40	82.39 ± 3.45	85.30 ± 1.89	<b>99.21 ± 0.30</b>	80.55 ± 0.28	65.03 ± 5.19
BM <sup>2</sup> SMVL	<b>83.48 ± 3.03</b>	<b>78.87 ± 3.63</b>	<b>86.52 ± 2.19</b>	<b>87.54 ± 1.71</b>	97.99 ± 0.57	<b>82.68 ± 0.26</b>	<b>89.12 ± 1.55</b>
OPMV	73.44 ± 2.06	66.40 ± 3.83	72.46 ± 3.12	78.75 ± 3.76	-	-	73.38 ± 2.02
OBM <sup>2</sup> SMVL	<b>77.96 ± 3.87</b>	<b>74.18 ± 4.63</b>	<b>81.58 ± 3.39</b>	<b>81.86 ± 3.19</b>	98.084 ± 0.95	74.74 ± 0.16	<b>75.43 ± 2.45</b>

Listed results are test averaged accuracies (%). Bold face indicates highest accuracy. 'N/A' means that no result returns after two weeks

**Table 4** Batch and online learning comparisons on multiclass data sets

	Washington	Cornell	Texas	Wisconsin	News4Gv	NUS-WIDE	Trecvid
MVMED	68.54 ± 2.25	68.98 ± 3.52	74.87 ± 5.52	81.93 ± 2.41	94.24 ± 0.84	N/A	77.74 ± 1.40
VMRML	76.04 ± 2.99	73.19 ± 4.65	81.73 ± 3.75	83.92 ± 2.29	93.31 ± 0.99	81.77 ± 00.22	79.03 ± 1.46
SVM-FULL	80.21 ± 3.69	74.96 ± 3.31	80.76 ± 4.03	83.29 ± 2.13	<b>99.21 ± 0.29</b>	80.39 ± 0.28	61.57 ± 8.12
BM <sup>2</sup> SMVL	<b>80.56 ± 3.14</b>	<b>76.82 ± 4.51</b>	<b>86.05 ± 2.57</b>	85.87 ± 1.77	97.87 ± 97.87	<b>82.41 ± 0.26</b>	<b>88.91 ± 1.70</b>
OPMV	67.53 ± 2.40	63.66 ± 4.77	77.02 ± 4.66	70.95 ± 3.70	-	-	72.89 ± 3.61
OBM <sup>2</sup> SMVL	<b>72.73 ± 4.88</b>	<b>70.91 ± 4.18</b>	<b>79.66 ± 3.59</b>	<b>78.76 ± 4.44</b>	98.05 ± 0.95	74.52 ± 1.58	<b>74.74 ± 01.78</b>

Listed results are test F1 Score (%). Bold face indicates highest accuracy. 'N/A' means that no result returns after two weeks

for MMH, but the software doesn't provide the F1 score, so its result is missing in Table 4. In online learning experiments, we use the same training/testing split of the above batch learning experiments. We sample 0.1 of the training data as the batch training, and the rest come one by one. Since OPMV can only deal with two-view data, so its result is missing for News4Gv and NUS-WIDE in Tables 3 and 4. From Tables 3 and 4, we have the following insightful observations:

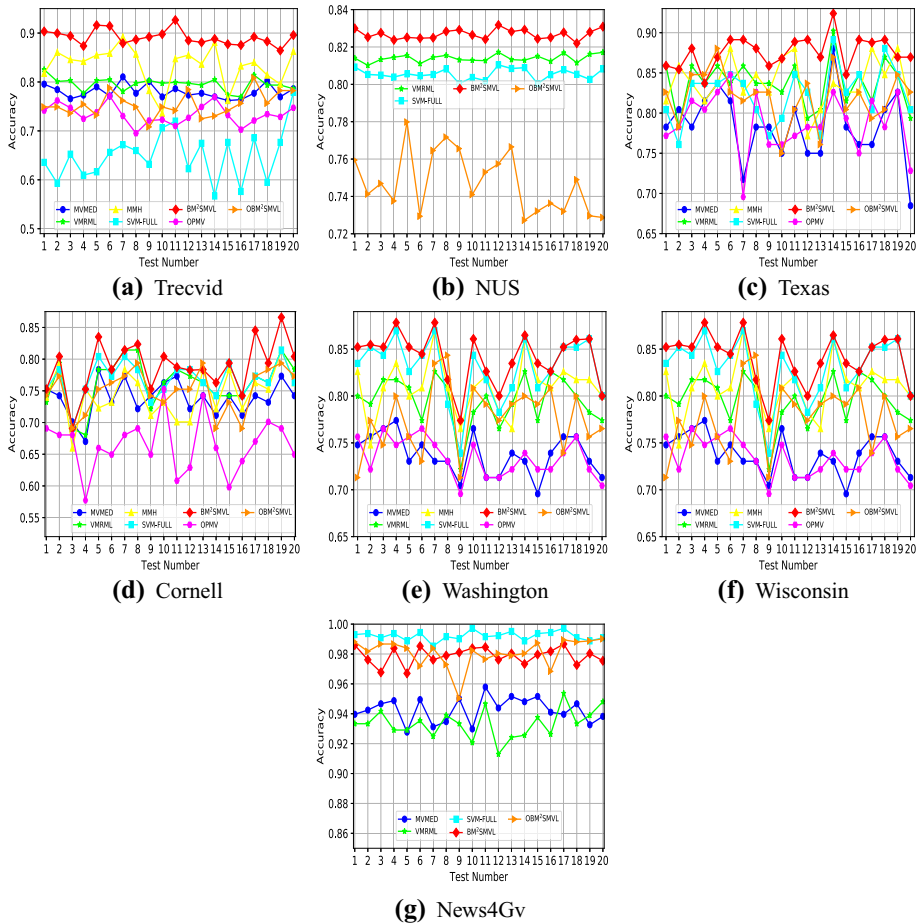
- Our  $BM^2SMVL$  achieves the best performance on the Trecvid, NUS-WIDE, Washington, Cornell, Texas and Wisconsin data sets and performs just a little worse than the SVM-FULL in the News4Gv data. We attribute it to that our method can automatically infer the penalty parameter of max-margin model based on the data augmentation idea, while MVMED and MMH are both under the maximum entropy discrimination framework and cannot infer the penalty parameter. SVM-FULL makes full use of all the information from the observations by concatenating all views to form a new single view. This maybe the reason why it performs better than our  $BM^2SMVL$  in the News4Gv. But some information from the observations is not helpful for the classification in other data sets. In this case, SVM-FULL cannot achieve a good performance.
- Our method infers a posterior under the Bayesian framework instead of a point estimate as in VMRML. With Bayesian model averaging over the posterior, we can make more robust predictions than VMRML.
- We also find that  $OBM^2SMVL$  performs better than OPMV on all data sets and just a little worse than  $BM^2SMVL$ . Unlike OPMV, which seeks a point estimate by optimizing some deterministic objective function, our online model infers a posterior under the Bayesian framework. The point estimate can be affected seriously by inappropriate regularization, outliers and noises, especially when the training data arrive sequentially.

We show every independent experimental run about all models and dataset in Figs. 1, 2 and 3.

For regression task, the results of all models are averaged over 5 independent runs on each data set. All the results are shown in Table 5. The ratio sampled for training data is 0.5 in the data set HotelReview and CT-Image and MovieLens. Since CoR-LS and LCFS can only deal with two-view data, so their results are missing for MovieLens in Table 5.

In online regression learning experiments, we use the same training/testing split of the above batch learning experiments. We sample 0.1 of the training data as the batch training, and the rest samples come with a number of 30. From Table 5, we have the following insightful observations:

- $BM^2SMVL$  consistently outperforms SVR-FULL. The reason may be that SVR-FULL just concatenates all views to form a new single view but some information from the observations is not helpful for regression on some data sets. And SVR-FULL doesn't take the relevance of different views into account.
- Our  $BM^2SMVL$  achieves the best performance on the HotelReview, MovieLens and CT-Image data sets. We attribute it to that our method can automatically infer the penalty parameter of max-margin model based on the data augmentation idea, while CoR-LS and LCFS both cannot infer the penalty parameter. What's more, Our method infers a posterior under the Bayesian framework instead of a point estimate as in CoR-LS and LCFS. With Bayesian model averaging over the posterior, we can make more robust predictions than CoR-LS and LCFS.
- We also find that  $OBM^2SMVL$  performs better than OSVR-FULL on all data sets. Unlike OSVR-FULL, which seeks a point estimate by optimizing some deterministic objective function, our online model infers a posterior under the Bayesian framework. The point



**Fig. 1** Classification (accuracy) results on different datasets

estimate can be affected seriously by inappropriate regularization, outliers and noises, especially when the training data arrive sequentially. What’s more, OSVR-FULL doesn’t consider the relevance of different views.

As we can see, our model performs better on most of the experiment runs. For some data sets, the split of the training/testing data influences the performance of all models. That is why the standard deviations of some data sets is a little big. And we can see from Figs. 1, 2 and 3, our model statistically performs the best on most of the data sets.

### 5.5 Computation complexity analysis

We compare efficiency of different algorithms on classification and regression tasks, the results are reported in Tables 6, 7 and 8. From Table 6, we can see VMRML costs the least time on most of small scale data sets. To further analysis the computational complexity, we conduct experiments on NUS-WIDE with different numbers of the training data (1000, 2000, 3000, 5000, 10,000) which is reported in Table 7. From Table 7, we find our models BM<sup>2</sup>SMVL

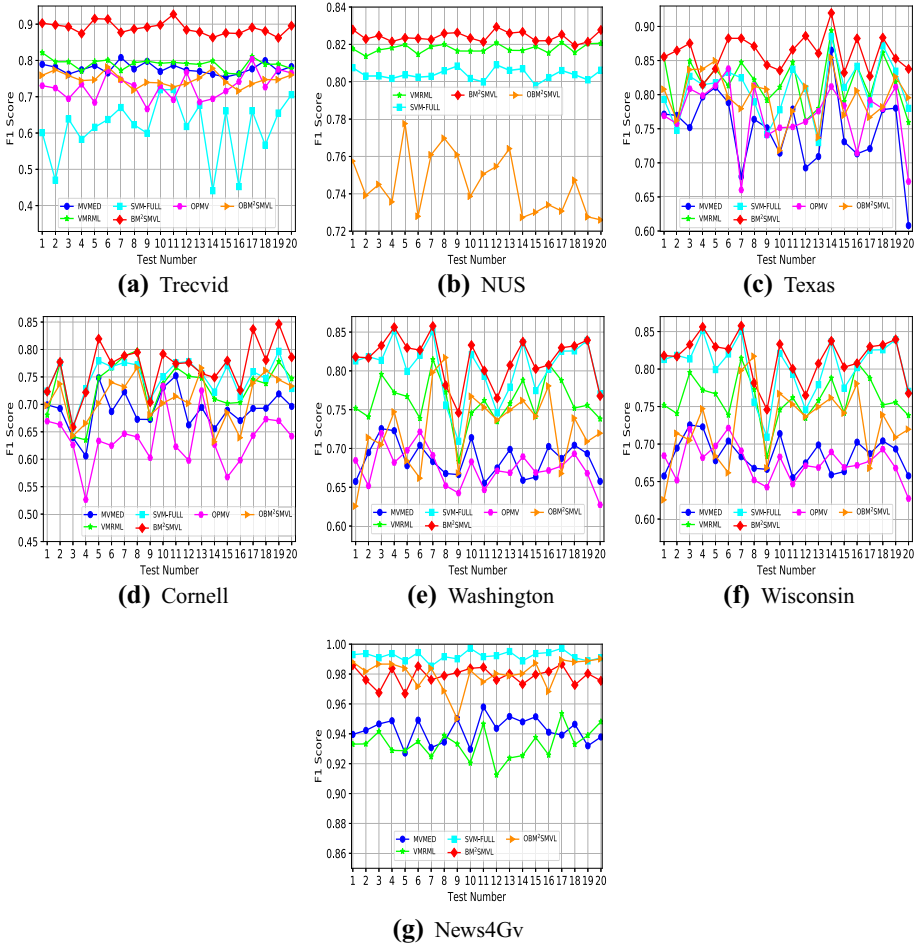


Fig. 2 Classification (F1 score)

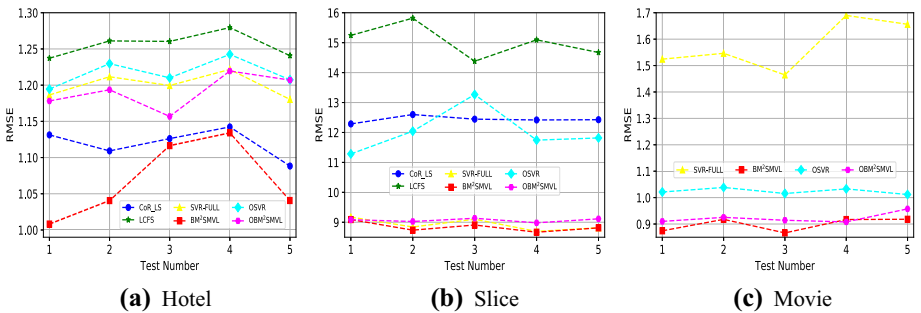


Fig. 3 Regression (RMSE) results on different datasets

**Table 5** Comparisons on regression data sets

	HotelReview	CT-Image	MovieLens
CoR-LS	1.12 ± 0.02	12.43 ± 0.11	-
LCFS	1.26 ± 0.02	15.04 ± 0.55	-
SVR-FULL	1.20 ± 0.02	8.92 ± 0.20	1.58 ± 0.10
BM <sup>2</sup> SMVL	<b>1.08 ± 0.05</b>	<b>8.87 ± 0.14</b>	<b>0.90 ± 0.02</b>
OSVR-FULL	1.20 ± 0.01	12.03 ± 0.74	1.02 ± 0.01
OBM <sup>2</sup> SMVL	<b>1.19 ± 0.02</b>	<b>9.07 ± 0.06</b>	<b>0.92 ± 0.02</b>

Listed results are averaged RMSE. Bold face indicates lowest RMSE

**Table 6** Training time (s) on classification tasks

Dataset	Cornell	Texas	Washington	Wisconsin	Trecvid	News4Gv
MVMED	48.49	45.09	44.10	63.46	804.35	1860.41
VMRML	<b>0.02</b>	<b>0.02</b>	<b>0.02</b>	<b>0.02</b>	<b>0.03</b>	<b>0.97</b>
SVM-FULL	0.10	0.05	0.07	0.08	0.59	4.11
B <sup>2</sup> SMVL	7.29	10.91	12.24	13.49	46.65	110.87
OPMV	6.54	6.30	7.81	9.18	47.93	–
OB <sup>2</sup> SMVL	11.24	14.30	14.14	15.07	50.65	337.90

Bold face indicates the least time

**Table 7** Training time (s) on classification data set NUS

Algorithm	Metric (s)	$N_{train} = 1000$	$N_{train} = 2000$	$N_{train} = 3000$	$N_{train} = 5000$	$N_{train} = 10,000$
VMRML	Train-time	<b>4</b>	18	50	210	1335
MVMED	Train-time	1363	9721	N/A	N/A	N/A
SVM-FULL	Train-time	7	<b>13</b>	<b>19</b>	<b>34</b>	<b>78</b>
BM <sup>2</sup> SMVL	Train-time	234	277	340	490	776
OBM <sup>2</sup> SMVL	Train-time	280	312	409	567	928

‘N/A’ means that no result returns after 24h. ‘–’ means out of memory. All experiments were conducted in Matlab

Bold face indicates the least time

**Table 8** Training time (s) on regression Tasks

Dataset	HotelReview	MovieLens	CT-Image
CoR-LS	27,949.34	–	25.14
LCFS	4956.47	–	<b>23.54</b>
SVR-FULL	<b>62.42</b>	<b>0.08</b>	37.00
B <sup>2</sup> SMVL	1083.40	13.49	2643.25
OSVR	<b>726.56</b>	20.42	96,223.27
OB <sup>2</sup> SMVL	1335.69	<b>16.21</b>	<b>2631.83</b>

Bold face indicates the least time

and OBM<sup>2</sup>SMVL scale linearly with the number of training data  $N$  which coincides with the computational complexity discussed in Section Computational Complexity. Although the training time of VMRML is shorter than that of BM<sup>2</sup>SMVL and OBM<sup>2</sup>SMVL in Table 7,

it seems that VMRML scales squarely with the number of training data and VMRML needs to store 5 Gram matrixes for both training and testing data on NUS-WIDE data set. Further more, we conduct the experiment with 20,000 training data, VMRML is out of memory while  $BM^2SMVL$  and  $OBM^2SMVL$  still work. Besides,  $BM^2SMVL$  performs better than VMRML on all considered data. From Table 7, we can also find the training time of SVM-FULL is the shortest, because it adopts the ‘linear’ kernel with low algorithm complexity. But SVM-FULL achieves the lowest accuracy and F1 score on NUS-WIDE compared to other offline multi-view methods.

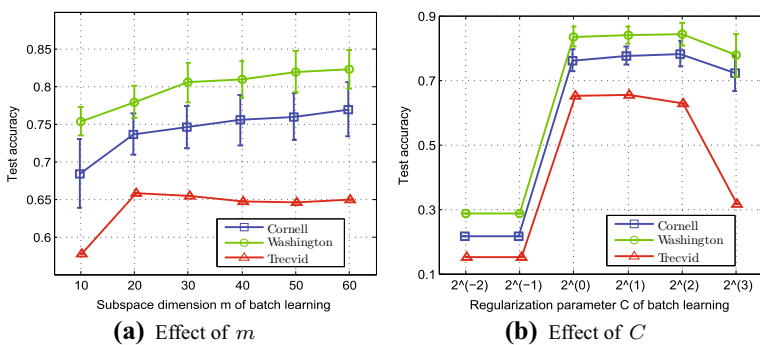
For regression, our online model  $OBM^2SMVL$  takes less time than OSVR-FULL on MovieLens and CT-Image. In batch learning, SVR-FULL takes the least time on MovieLens and Hotel, LCFS takes the least time on CT-Image.  $BM^2SMVL$  takes less time than CoRLS and LCFS on HotelReview, it is because the dimensions of the views are very high on HotelReview.  $BM^2SMVL$  learns low-dimension representations from multiple views, so  $BM^2SMVL$  shows advantages when the dimensions of views are high. What’s more, our model achieves the lowest RMSE on all considered data sets. Although our model can not cost the least time on all data sets, we believe it’s an acceptable and reasonable trade-off between the model complexity and performance for our models.

### 5.6 Sensitivity analysis

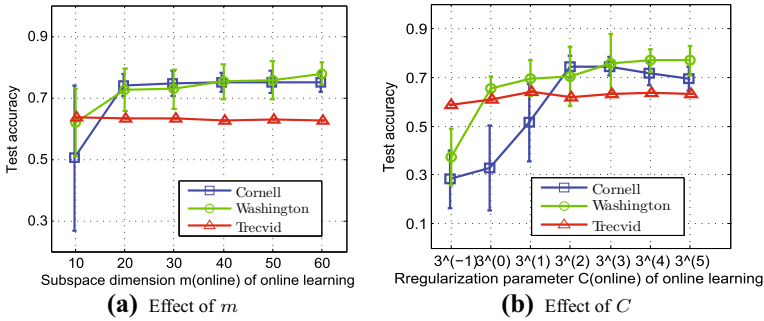
We study the sensitivity of  $BM^2SMVL$  and  $OBM^2SMVL$  with respect to the subspace dimension  $m$ , and the regularization parameter  $C$ .

When we study the influence of  $m$ ,  $C$  (batch) is set as 2 for  $BM^2SMVL$  and  $C$  (online) is set as 15 for  $OBM^2SMVL$ . The averaged results are shown in Figs. 4a and 5a. We find that the test accuracy increases when  $m$  becomes larger. And when  $m$  is large enough, the test accuracy remains stable.

When we study the influence of  $C$ ,  $m$  is set as 30 for both batch and online learning. From the results in Figs. 4b and 5b, we can find that different data sets may prefer different values of  $C$ . In batch learning,  $C$  (batch) balances the classification model and subspace learning model, so our model cannot get the best performance when  $C$  (batch) is too large or too small.  $C$  (online) reflects the importance of new arrival data in our online model. When  $C$  (online) is too small, the new arrival data plays a tiny role in the online model and offers little help to improve the performance of our online model. For some data sets like Cornell,



**Fig. 4** (a) Results on different data sets with different parameters  $m$  in  $BM^2SMVL$ ; (b) Results on different data sets with different regularization parameters  $C$  in  $BM^2SMVL$



**Fig. 5** (a) Results on different data sets with different subspace dimensions  $m$  (online) in  $OBM^2SMVL$ ; (b) Results on different data sets with different regularization parameters  $C$  (online) in  $OBM^2SMVL$

when  $C$  (online) is too large, the performance of  $OBM^2SMVL$  would become bad because the online model doesn't take full advantage of the historical knowledge. For some other data sets like Trecvid and Washington, they are less sensitive to  $C$  (online) when  $C$  (online) is large enough.

### 6 Conclusion

We propose an online Bayesian method to learn predictive subspace for multi-view data. Specifically, the proposed method is based on the data augmentation idea for max-margin learning, which allows us to automatically infer the weight and penalty parameter and find the most appropriate predictive subspace simultaneously under the Bayesian framework. Experiments on various classification and regression tasks show that both our batch model  $BM^2SMVL$  and online model  $OBM^2SMVL$  can achieve superior performance, compared with a number of state-of-the-art competitors.

**Acknowledgements** This work was supported by the National Key Research and Development Program of China under Grant No. 2018YFB1004300, the National Natural Science Foundation of China under Grant No. U1811461, 61602449, U1836206, 61773361, the Project of Youth Innovation Promotion Association CAS under Grant No. 2017146.

### Appendix

For  $BM^2SVML$ , we calculate the updating formula of rest parameters which are omitted in the main body as follows:

$$\begin{aligned}
 V^*(\phi) &= \prod_{i=1}^{Nv} \Gamma(\phi_i | a_{\phi_i}, b_{\phi_i}) \\
 \tilde{a}_{\phi_i} &= a_{\phi_i} + \frac{Nd_i}{2} \\
 \tilde{b}_{\phi_i} &= b_{\phi_i} + \frac{1}{2} \sum_{n=1}^N \langle \|\mathbf{x}_i^{(n)} - \mathbf{W}_i \mathbf{z}^{(n)} + \boldsymbol{\mu}_i\| \rangle
 \end{aligned}$$



$$\mathbb{E}_{\phi_i}[\phi_i] = \tilde{a}_{\phi_i} / \tilde{b}_{\phi_i} \tag{16}$$

$$V^*(\mathbf{W}) = \prod_{i=1}^{Nv} \prod_{j=1}^{d_i} \mathcal{N}(w_{ij} | \mu_{w_{ij}}, \Sigma_{w_{ij}})$$

$$\Sigma_{w_{ij}} = \left[ \text{diag}(\alpha_i) + \sum_{k=1}^N (\langle \phi_i \rangle \langle \mathbf{z}^{(k)} (\mathbf{z}^{(k)})^T \rangle) \right]^{-1}$$

$$\mu_{w_{ij}} = \langle \phi_i \rangle \sum_{k=1}^N \{ (\mathbf{x}_i^{(k)} - \langle \boldsymbol{\mu}_i \rangle) \langle \mathbf{z}^{(k)} \rangle^T \} \Sigma_{w_{ij}}$$

$$\mathbb{E}_{w_{ij}}[w_{ij}] = \mu_{w_{ij}} \tag{17}$$

$$V^*(\boldsymbol{\mu}) = \prod_{i=1}^{Nv} \mathcal{N}(\boldsymbol{\mu}_i | \mu_{\boldsymbol{\mu}_i}, \Sigma_{\boldsymbol{\mu}_i})$$

$$\Sigma_{\boldsymbol{\mu}_i} = (\beta_i + N \langle \phi_i \rangle)^{-1} \mathbf{I}_{d_i}$$

$$\mu_{\boldsymbol{\mu}_i} = \Sigma_{\boldsymbol{\mu}_i} \langle \Phi_i \rangle \sum_{n=1}^N (\mathbf{x}_i^{(n)} - \langle W_i \rangle \langle \mathbf{z}^{(n)} \rangle)$$

$$\mathbb{E}_{\boldsymbol{\mu}_i}[\boldsymbol{\mu}_i] = \mu_{\boldsymbol{\mu}_i} \tag{18}$$

$$V^*(\alpha) = \prod_{i=1}^{Nv} \prod_{j=1}^m \Gamma(\alpha_{ij} | a_i, b_{ij})$$

$$a_i = a + d_i / 2$$

$$b_{ij} = b + \langle \|W_{i(\cdot, j)}\|^2 \rangle / 2$$

$$\mathbb{E}_{\alpha_{ij}}[\alpha_{ij}] = a_i / b_{ij} \tag{19}$$

$$V^*(v) = \prod_{c=1}^{Nc} \Gamma(v_c | \tilde{a}_{v_c}, \tilde{b}_{v_c})$$

$$\tilde{a}_{v_c} = a_{v_c} + (m + 1) / 2,$$

$$\tilde{b}_{v_c} = b_{v_c} + \langle \|\boldsymbol{\eta}_c\|^2 \rangle / 2$$

$$\mathbb{E}_{v_c}[v_c] = \tilde{a}_{v_c} / \tilde{b}_{v_c} \tag{20}$$

$$V^*(\boldsymbol{\eta}) = \prod_{c=1}^{Nc} \mathcal{N}(\boldsymbol{\eta}_c | \mu_{\boldsymbol{\eta}_c}^c, \Sigma_{\boldsymbol{\eta}_c}^c)$$

$$\Sigma_{\boldsymbol{\eta}_c} = \left\{ C^2 \sum_{n=1}^N \tilde{\mathbf{Z}} (\text{diag}(\langle \lambda_c^{-1} \rangle)) \tilde{\mathbf{Z}}^T + \langle v_c \rangle \mathbf{I}_{(m+1)} \right\}^{-1}$$

$$\mu_{\boldsymbol{\eta}_c} = \Sigma_{\boldsymbol{\eta}_c} \sum_{n=1}^N C (1 + C \langle (\lambda_c^{(n)})^{-1} \rangle) y_c^{(n)} \langle \tilde{\mathbf{z}}^{(n)} \rangle$$

$$\mathbb{E}_{\boldsymbol{\eta}_c}[\boldsymbol{\eta}_c] = \mu_{\boldsymbol{\eta}_c} \tag{21}$$

For regression, only the variables  $\mathbf{Z}$ ,  $\boldsymbol{\eta}$ ,  $\boldsymbol{\lambda}$  and  $\boldsymbol{\theta}$  are different from the classification model, so we only provide the updating formulas for  $\mathbf{Z}$ ,  $\boldsymbol{\eta}$ ,  $\boldsymbol{\lambda}$  and  $\boldsymbol{\theta}$  here:

$$\begin{aligned}
 V^*(\mathbf{Z}) &= \prod_{n=1}^N \mathcal{N}(\mathbf{z}^{(n)} | \boldsymbol{\mu}_{\mathbf{z}}^{(n)}, \boldsymbol{\Sigma}_{\mathbf{z}}^{(n)}) \\
 \boldsymbol{\Sigma}_{\mathbf{z}}^{(n)} &= \left\{ C_R^2 (\langle \boldsymbol{\lambda}^{(n)} \rangle^{-1} + \langle \boldsymbol{\theta}^{(n)} \rangle^{-1}) \langle \tilde{\boldsymbol{\eta}} \tilde{\boldsymbol{\eta}}^T \right. \\
 &\quad \left. + \mathbf{I}_m + \sum_{i=1}^{Nv} \langle \phi_i \rangle \langle \mathbf{W}_i^T \mathbf{W}_i \rangle \right\}^{-1} \\
 \boldsymbol{\mu}_{\mathbf{z}}^{(n)} &= \boldsymbol{\Sigma}_{\mathbf{z}}^{(n)} \left\{ \sum_{i=1}^{Nv} \langle \phi_i \rangle \langle \mathbf{W}_i^T \rangle (\mathbf{x}_i^{(n)} - \langle \boldsymbol{\mu}_i \rangle) \right. \\
 &\quad + C_R^2 \{ \langle \boldsymbol{\lambda}^{(n)} \rangle^{-1} (\mathbf{y}^{(n)} - \epsilon) + \langle \boldsymbol{\theta}^{(n)} \rangle^{-1} (\mathbf{y}^{(n)} + \epsilon) \} \langle \tilde{\boldsymbol{\eta}} \rangle \\
 &\quad \left. - C_R^2 (\langle \boldsymbol{\lambda}^{(n)} \rangle^{-1} + \langle \boldsymbol{\theta}^{(n)} \rangle^{-1}) \langle \boldsymbol{\eta}_{(m+1)} \tilde{\boldsymbol{\eta}} \rangle \right\} \\
 \mathbb{E}_{\mathbf{z}^{(n)}}[\mathbf{z}^{(n)}] &= \boldsymbol{\mu}_{\mathbf{z}}^{(n)} \tag{22}
 \end{aligned}$$

$$\begin{aligned}
 V^*(\boldsymbol{\lambda}) &= \prod_{n=1}^N \mathcal{GIG}(\boldsymbol{\lambda}^{(n)} | \frac{1}{2}, 1, \chi_{\boldsymbol{\lambda}}^{(n)}) \\
 \chi_{\boldsymbol{\lambda}}^{(n)} &= C_R^2 \langle [\boldsymbol{\eta}^T \tilde{\mathbf{z}}^{(n)} - (\mathbf{y}^{(n)} - \epsilon)]^2 \rangle \\
 \mathbb{E}_{\boldsymbol{\lambda}^{(n)}}[(\boldsymbol{\lambda}^{(n)})^{-1}] &= 1/\sqrt{\chi_{\boldsymbol{\lambda}}^{(n)}} \tag{23}
 \end{aligned}$$

$$\begin{aligned}
 V^*(\boldsymbol{\theta}) &= \prod_{n=1}^N \mathcal{GIG}(\boldsymbol{\theta}^{(n)} | \frac{1}{2}, 1, \chi_{\boldsymbol{\theta}}^{(n)}) \\
 \chi_{\boldsymbol{\theta}}^{(n)} &= C_R^2 \langle [\boldsymbol{\eta}^T \tilde{\mathbf{z}}^{(n)} - (\mathbf{y}^{(n)} + \epsilon)]^2 \rangle \\
 \mathbb{E}_{\boldsymbol{\theta}^{(n)}}[(\boldsymbol{\theta}^{(n)})^{-1}] &= 1/\sqrt{\chi_{\boldsymbol{\theta}}^{(n)}} \tag{24}
 \end{aligned}$$

$$\begin{aligned}
 V^*(\boldsymbol{\eta}) &= \mathcal{N}(\boldsymbol{\eta} | \boldsymbol{\mu}_{\boldsymbol{\eta}}, \boldsymbol{\Sigma}_{\boldsymbol{\eta}}) \\
 \boldsymbol{\Sigma}_{\boldsymbol{\eta}} &= \{ C_R^2 \tilde{\mathbf{Z}} (\text{diag}(\langle \boldsymbol{\lambda}^{-1} \rangle + \langle \boldsymbol{\theta}^{-1} \rangle)) \tilde{\mathbf{Z}}^T + \langle v \rangle \mathbf{I}_{(m+1)} \}^{-1} \\
 \boldsymbol{\mu}_{\boldsymbol{\eta}} &= \boldsymbol{\Sigma}_{\boldsymbol{\eta}} \{ C_R^2 \tilde{\mathbf{Z}} \{ \langle \boldsymbol{\lambda}^{-1} \rangle \cdot (\mathbf{y} - \epsilon) + \langle \boldsymbol{\theta}^{-1} \rangle \cdot (\mathbf{y} + \epsilon) \} \} \\
 \mathbb{E}_{\boldsymbol{\eta}}[\boldsymbol{\eta}] &= \boldsymbol{\mu}_{\boldsymbol{\eta}} \tag{25}
 \end{aligned}$$

Similarly, we can get the updating formulas for OBM<sup>2</sup>SMVL as follows:

$$\begin{aligned}
 V^*(\phi^{(t+1)}) &= \prod_{i=1}^{Nv} \Gamma(\phi_i^{(t+1)} | a_{\phi_i}^{(t+1)}, b_{\phi_i}^{(t+1)}) \\
 \tilde{a}_{\phi_i}^{(t+1)} &= a_{\phi_i} + \frac{Ld_i}{2} \\
 \tilde{b}_{\phi_i}^{(t+1)} &= b_{\phi_i} + \frac{1}{2} \sum_{l=1}^L \langle \|\mathbf{x}_i^{(l)} - \mathbf{W}_i^{(t+1)} \mathbf{z}^{(l)} + \boldsymbol{\mu}_i^{(t+1)}\| \rangle \\
 \mathbb{E}_{\phi_i}^{(t+1)}[\phi_i] &= \tilde{a}_{\phi_i}^{(t+1)} / \tilde{b}_{\phi_i}^{(t+1)}
 \end{aligned}$$

$$V^*(\mathbf{W}^{(t+1)}) = \prod_{i=1}^{Nv} \prod_{j=1}^{d_i} \mathcal{N}(w_{ij}^{(t+1)} | \mu_{w_{ij}}^{(t+1)}, \Sigma_{w_{ij}}^{(t+1)}) \tag{26}$$

$$\Sigma_{w_{ij}}^{(t+1)} = \left[ (\Sigma_{w_{ij}}^{(t)})^{-1} + (\langle \phi_i^{(t+1)} \rangle \sum_{l=1}^L \langle \mathbf{z}^{(l)} (\mathbf{z}^{(l)})^T \rangle) \right]^{-1}$$

$$\mu_{w_{ij}}^{(t+1)} = \left\{ \langle \phi_i^{(t+1)} \rangle \sum_{l=1}^L [\langle \mathbf{x}_i^{(l)} - \langle \mu_i^{(l)} \rangle \langle \mathbf{z}^{(l)} \rangle^T] + \mu_{w_{ij}}^{(t)} (\Sigma_{w_{ij}}^{(t)})^{-1} \right\} \Sigma_{w_{ij}}^{(t+1)}$$

$$\mathbb{E}_{w_{ij}}^{(t+1)}[w_{ij}] = \mu_{w_{ij}}^{(t+1)} \tag{27}$$

$$V^*(\boldsymbol{\mu}^{(t+1)}) = \prod_{i=1}^{Nv} \mathcal{N}(\mu_i^{(t+1)} | \mu_{\mu_i}^{(t+1)}, \Sigma_{\mu_i}^{(t+1)})$$

$$\Sigma_{\mu_i}^{(t+1)} = ((\Sigma_{\mu_i}^{(t)})^{-1} + L \langle \phi_i^{(t+1)} \rangle)^{-1} \mathbf{I}_{d_i}$$

$$\mu_{\mu_i}^{(t+1)} = \Sigma_{\mu_i}^{(t+1)} \langle \phi_i^{(t+1)} \rangle \left[ \sum_{l=1}^L \langle \mathbf{x}_i^{(l)} - \langle \mathbf{W}_i^{(t+1)} \rangle \langle \mathbf{z}^{(l)} \rangle \right] + (\Sigma_{\mu_i}^{(t)})^{-1} \mu_{\mu_i}^{(t)}$$

$$\mathbb{E}_{\mu_i}^{(t+1)}[\mu_i] = \mu_{\mu_i}^{(t+1)} \tag{28}$$

$$V^*(\boldsymbol{\eta}^{(t+1)}) = \prod_{c=1}^{Nc} \mathcal{N}(\eta_c^{(t+1)} | \mu_{\eta_c}^{(t+1)}, \Sigma_{\eta_c}^{(t+1)})$$

$$\Sigma_{\eta_c}^{(t+1)} = \left\{ C^2 \sum_{l=1}^L \langle (\tilde{\mathbf{z}}^{(l)} (\tilde{\mathbf{z}}^{(l)})^T) \langle \lambda_c^{(l)-1} \rangle + (\Sigma_{\eta_c}^{(t)})^{-1} \right\}^{-1}$$

$$\mu_{\eta_c}^{(t+1)} = \Sigma_{\eta_c}^{(t+1)} \sum_{l=1}^L \{ C(1 + C \langle \lambda_c^{(l)-1} \rangle) y_c^{(l)} \langle \tilde{\mathbf{z}}^{(l)} \rangle + (\Sigma_{\eta_c}^{(t)})^{-1} \mu_{\eta_c}^{(t)} \}$$

$$\mathbb{E}_{\eta_c}^{(t+1)}[\eta_c] = \mu_{\eta_c}^{(t+1)} \tag{29}$$

For regression, only the variables  $\mathbf{Z}^{(t+1)}$ ,  $\boldsymbol{\eta}^{(t+1)}$ ,  $\boldsymbol{\lambda}^{(t+1)}$  and  $\boldsymbol{\theta}^{(t+1)}$  are different from the classification model, so we only provide the updating formulas for  $\mathbf{Z}^{(t+1)}$ ,  $\boldsymbol{\eta}^{(t+1)}$ ,  $\boldsymbol{\lambda}^{(t+1)}$  and  $\boldsymbol{\theta}^{(t+1)}$  here:

$$V^*(\boldsymbol{\lambda}^{t+1}) = \prod_{l=1}^L \mathcal{GIG}(\lambda^{(l)} | \frac{1}{2}, 1, \chi_{\lambda}^{(l)})$$

$$\chi_{\lambda}^{(l)} = C_R^2 \langle [\boldsymbol{\eta}^T \tilde{\mathbf{z}}^{(l)} - (y^{(l)} - \epsilon)]^2 \rangle$$

$$\mathbb{E}_{\lambda}[(\lambda^{(l)})^{-1}] = 1/\sqrt{\chi_{\lambda}^{(l)}} \tag{30}$$

$$V^*(\boldsymbol{\theta}^{t+1}) = \prod_{l=1}^L \mathcal{GIG}(\theta^{(l)} | \frac{1}{2}, 1, \chi_{\theta}^{(l)})$$

$$\chi_{\theta}^{(l)} = C_R^2 \langle [\boldsymbol{\eta}^T \tilde{\mathbf{z}}^{(l)} - (y^{(l)} + \epsilon)]^2 \rangle$$

$$\mathbb{E}_{\theta}[(\theta^{(l)})^{-1}] = 1/\sqrt{\chi_{\theta}^{(l)}} \tag{31}$$

$$V^*(\boldsymbol{\eta}) = \mathcal{N}(\boldsymbol{\eta} | \mu_{\boldsymbol{\eta}}^{(t+1)}, \Sigma_{\boldsymbol{\eta}}^{(t+1)})$$

$$\begin{aligned}
 \Sigma_{\eta}^{(t+1)} &= \left\{ (\Sigma_{\eta}^{(t)})^{-1} + \sum_{l=1}^L C^2 \{ (\lambda^{(l)})^{-1} + (\theta^{(l)})^{-1} \} \mathbf{z}^{(l)} (\mathbf{z}^{(l)})^T \right\}^{-1} \\
 \mu_{\eta}^{(t+1)} &= \Sigma_{\eta}^{(t+1)} \left\{ (\Sigma_{\eta}^{(t)})^{-1} \mu_{\eta}^{(t)} + \sum_{l=1}^L C^2 \{ (\lambda^{(l)})^{-1} \right. \\
 &\quad \left. (y^{(l)} - \epsilon) + (\theta^{(l)})^{-1} (y^{(l)} + \epsilon) \} \mathbf{z}^{(l)} \right\} \\
 \mathbb{E}_{\eta}^{(t+1)}[\eta] &= \mu_{\eta}^{(t+1)} \tag{32} \\
 V^*(\mathbf{Z}^{(t+1)}) &= \prod_{l=1}^L \mathcal{N}(\mathbf{z}^{(l)} | \mu_{\mathbf{z}}^{(l)}, \Sigma_{\mathbf{z}}^{(l)}) \\
 \Sigma_{\mathbf{z}}^{(l)} &= \{ C_R^2 \{ (\lambda^{(l)})^{-1} + (\theta^{(l)})^{-1} \} \langle \tilde{\eta}^{(t+1)} (\tilde{\eta}^{(t+1)})^T \rangle \\
 &\quad + \mathbf{I}_m + \sum_{i=1}^{N_v} \langle \phi_i^{(t+1)} \rangle \langle (\mathbf{W}_i^{(t+1)})^T \mathbf{W}_i^{(t+1)} \rangle \}^{-1} \\
 \mu_{\mathbf{z}}^{(l)} &= \Sigma_{\mathbf{z}}^{(l)} \left\{ \sum_{i=1}^{N_v} \langle \phi_i^{(t+1)} \rangle \langle (\mathbf{W}_i^{(t+1)})^T \rangle (\mathbf{x}_i^{(l)} - \langle \mu_i^{(t+1)} \rangle) \right. \\
 &\quad + C_R^2 \{ (\lambda^{(l)})^{-1} (y^{(l)} - \epsilon) + (\theta^{(l)})^{-1} (y^{(l)} + \epsilon) \} \langle \tilde{\eta} \rangle \\
 &\quad \left. - C_R^2 \{ (\lambda^{(l)})^{-1} + (\theta^{(l)})^{-1} \} \langle \eta_{(m+1)} \tilde{\eta} \rangle \right\} \\
 \mathbb{E}_{\mathbf{z}^{(l)}}[\mathbf{z}^{(l)}] &= \mu_{\mathbf{z}}^{(l)} \tag{33}
 \end{aligned}$$

## References

Beal, J. M. (2003). *Variational algorithms for approximate bayesian inference*. London: University College London.

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th annual conference on computational learning theory* (pp. 92–100).

Brić, M., & Kopriva, I. (2018). Multi-view low-rank sparse subspace clustering. *Pattern Recognition*, 73, 247–258.

Cesa-Bianchi, N., & Lugosi, G. (2006). *Prediction, learning, and games*. Cambridge: Cambridge University Press.

Chechik, G., Sharma, V., Shalit, U., & Bengio, S. (2010). Large scale online learning of image similarity through ranking. *The Journal of Machine Learning Research*, 11, 1109–1135.

Chen, K., & Jie, Y. (2014). Short-term wind speed prediction using an unscented Kalman filter based state-space support vector regression approach. *Applied Energy*, 113(6), 690–705.

Chen, N., Zhu, J., Sun, F., & Xing, E. P. (2012). Large-margin predictive latent subspace learning for multiview data analysis. *Pattern Analysis and Machine Intelligence*, 34(12), 2365–2378.

Chen, Z., Yang, L. F., Li, C. J., & Zhao, T. (2017). Online partial least square optimization: Dropping convexity for better efficiency and scalability. In *Proceedings of the 34th international conference on machine learning* (Vol. 70, pp. 777–786). JMLR. org.

Chen, Z., & Zhou, J. (2018). Collaborative multiview hashing. *Pattern Recognition*, 75, 149–160.

Chiang, D., Marton, Y., & Resnik, P. (2008). Online large-margin training of syntactic and structural translation features. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 224–233).

- Chua, T.-S., Tang, J., Hong, R., Li, H., Luo, Z., & Zheng, Y.-T. (2009). Nus-wide: A real-world web image database from National University of Singapore. In *CIVR*. Santorini, Greece., July 8–10.
- Cramer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., & Singer, Y. (2006). Online passive-aggressive algorithms. *The Journal of Machine Learning Research*, 7, 551–585.
- Deng, S., Gao, K., Du, C., Ma, W., Long, G., & Li, Y. (2016). Online variational bayesian support vector regression. In *International joint conference on neural networks* (pp. 3950–3957).
- Du, C., Zhe, S., Zhuang, F., Qi, Y., He, Q., & Shi, Z. (2015). Bayesian maximum margin principal component analysis. In *The 29th AAAI conference on artificial intelligence*.
- Gilks, W. R. (2005). *Markov chain monte carlo*. New York: Wiley.
- Gönen, M., & Alpaydm, E. (2011). Multiple kernel learning algorithms. *The Journal of Machine Learning Research*, 12, 2211–2268.
- Grangier, D., & Bengio, S. (2008). A discriminative kernel-based approach to rank images from text queries. *Pattern Analysis and Machine Intelligence*, 30(8), 1371–1384.
- Guo, Y., & Xiao, M. (2012). Cross language text classification via subspace co-regularized multi-view learning. *Computer Science—Computation and Language*. In *Proceedings of the 29th International Conference on Machine Learning* (915–922). Omnipress.
- Hardoon, D. R., Szedmak, S., & Shawe-Taylor, J. (2004). Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12), 2639–2664.
- Hazan, E., Agarwal, A., & Kale, S. (2007). Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2–3), 169–192.
- He, J., Du, C., Zhuang, F., Yin, X., He, Q., & Long, G. (2016). Online bayesian max-margin subspace multi-view learning. In *IJCAI* (pp. 1555–1561).
- Jiang, H., & He, W. (2012). Grey relational grade in local support vector regression for financial time series prediction. *Expert Systems with Applications*, 39(3), 2256–2262.
- Kalteh, A. M. (2013). Monthly river flow forecasting using artificial neural network and support vector regression models coupled with wavelet transform. *Computers and Geosciences*, 54(4), 1–8.
- Kazem, A., Sharifi, E., Hussain, F. K., Saberi, M., & Hussain, O. K. (2013). Support vector regression with chaos-based firefly algorithm for stock market price forecasting. *Applied Soft Computing*, 13(2), 947–958.
- Lan, C., Deng, Y., Li, X., & Huan, J. (2016). Co-regularized least square regression for multi-view multi-class classification. In *2016 International joint conference on neural networks (IJCNN)* (pp. 342–347). IEEE.
- Li, Y., Yang, M., & Zhang, Z. (2016). Multi-view representation learning: A survey from shallow methods to deep methods. arXiv preprint [arXiv:1610.01206](https://arxiv.org/abs/1610.01206).
- Lin, K.-P., Lu, Y.-M., Pai, P.-F., & Chang, P.-T. (2013). Revenue forecasting using a least-squares support vector regression model in a fuzzy environment. *Information Sciences*, 220(1), 196–209.
- Liu, Y., Zheng, Y., Liang, Y., Liu, S., & Rosenblum, D. S. (2016). Urban water quality prediction based on multi-task multi-view learning. In *Proceedings of the 25th international joint conference on artificial intelligence*.
- Long, B., Yu, P. S., & Zhang, Z. (2008). A general model for multiple view unsupervised learning. In *SIAM international conference on data mining* (pp. 822–833).
- Lu, C.-T., He, L., Shao, W., Cao, B., & Yu, P. S. (2017). Multilinear factorization machines for multi-task multi-view learning. In *Proceedings of the 10th ACM international conference on web search and data mining* (pp. 701–709). ACM.
- Merugu, S., Rosset, S., & Perlich, C. (2006). A new multi-view regression approach with an application to customer wallet estimation. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 656–661). ACM.
- Nguyen-Tuong, D., Peters, J. R., & Seeger, M. (2009). Local gaussian process regression for real time online model learning. In *Advances in neural information processing systems* (pp. 1193–1200).
- Parrella, F. (2007). Online support vector regression. Master's Thesis, Department of Information Science, University of Genoa, Italy.
- Polson, N. G., & Scott, S. L. (2011). Data augmentation for support vector machines. *Bayesian Analysis*, 6(1), 43–47.
- Quang, M. H., Bazzani, L., & Murino, V. (2013). A unifying framework for vector-valued manifold regularization and multi-view learning. In *International conference on machine learning* (pp. 100–108).
- Reents, G., & Urbanczik, R. (1998). Self-averaging and on-line learning. *Physical Review Letters*, 80(24), 5448.
- Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), 386.
- Shao, W., He, L., Lu, C.-T., Wei, X., & Philip, S Y. (2016). Online unsupervised multi-view feature selection. In *2016 IEEE 16th international conference on data mining (ICDM)* (pp. 1203–1208). IEEE.

- Sharma, A., Kumar, A., Daume III, H., & Jacobs, D. W. (2012). Generalized multiview analysis: A discriminative latent space. In *IEEE conference on computer vision and pattern recognition* (pp. 2160–2167).
- Shi, T., & Zhu, J. (2013). Online Bayesian passive-aggressive learning. In *International conference on machine learning* (pp. 378–386).
- Smola, A. J., & Lkocf, B. (2004). *A tutorial on support vector regression*. Dordrecht: Kluwer Academic Publishers.
- Sun, S., & Chao, G. (2013). Multi-view maximum entropy discrimination. In *International joint conference on artificial intelligence* (pp. 1706–1712).
- Ting, H. (2011). Online regression with varying gaussians and non-identical distributions. *Analysis and Applications*, 9(04), 395–408.
- Wang, K., He, R., Wang, W., Wang, L., & Tan, T. (2013). Learning coupled feature spaces for cross-modal matching. In *IEEE international conference on computer vision* (pp. 2088–2095).
- Xie, L., Shen, J., Han, J., Zhu, L., & Shao, L. (2017). Dynamic multi-view hashing for online image retrieval. In *IJCAI international joint conference on artificial intelligence* (pp. 3133–3139). AAAI Press.
- Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on association for computational linguistics* (pp. 189–196).
- Ye, G., Liu, D., Jhuo, I.-H., Chang, S.-F. et al. (2012). Robust late fusion with rank minimization. In *IEEE conference on computer vision and pattern recognition* (pp. 3021–3028).
- Ye, H.-J., Zhan, D.-C., Miao, Y., Jiang, Y., & Zhou, Z. (2015). Rank consistency based multi-view learning: A privacy-preserving approach. In *Proceedings of the 24th ACM international on conference on information and knowledge management* (pp. 991–1000). ACM.
- Zhang, C., Hu, Q., Fu, H., Zhu, P., & Cao, X. (2017). Latent multi-view subspace clustering. In *IEEE conference on computer vision and pattern recognition* (pp. 4333–4341).
- Zhao, H., Ding, Z., & Fu, Y. (2017). Multi-view clustering via deep matrix factorization. In *AAAI* (pp. 2921–2927).
- Zheng, S., Cai, X., Ding, C., Nie, F., & Huang, H. (2015). A closed form solution to multi-view low-rank regression. In *29th AAAI conference on artificial intelligence*.
- Zhu, J., Ahmed, A., & Xing, E. P. (2009). Medlda: Maximum margin supervised topic models for regression and classification. In *Proceedings of the 26th annual international conference on machine learning* (pp. 1257–1264). ACM.
- Zhu, J., Chen, N., Perkins, H., & Zhang, B. (2014). Gibbs max-margin topic models with data augmentation. *Journal of Machine Learning Research*, 15(1), 1073–1110.
- Zhu, J., Chen, N., & Xing, E. P. (2014). Bayesian inference with posterior regularization and applications to infinite latent svms. *Journal of Machine Learning Research*, 15, 1799.
- Zhu, Y., Gao, W., & Zhou, Z.-H. (2015). One-pass multi-view learning. In *Asian conference on machine learning* (pp. 407–422).

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

## Affiliations

Jia He<sup>1,2,4</sup> · Changying Du<sup>3,5</sup> · Fuzhen Zhuang<sup>1,4</sup>  · Xin Yin<sup>1</sup> · Qing He<sup>1,4</sup> · Guoping Long<sup>5</sup>

✉ Changying Du  
duchangying@huawei.com

Jia He  
hejia0149@gmail.com

Fuzhen Zhuang  
zhuangfuzhen@ict.ac.cn

Xin Yin  
yinxin@ics.ict.ac.cn

Qing He  
heqing@ict.ac.cn

Guoping Long  
guoping@iscas.ac.cn

- <sup>1</sup> The Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China
- <sup>2</sup> Huawei EI Innovation Lab, Beijing 100085, China
- <sup>3</sup> Huawei Noah's Ark Lab, Beijing 100085, China
- <sup>4</sup> The University of Chinese Academy of Sciences, Beijing 100049, China
- <sup>5</sup> The Lab of Parallel Software and Computational Science, Institute of Software, CAS, Beijing, China