IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 14, 2020, accepted March 23, 2020, date of publication March 30, 2020, date of current version April 14, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2984271

Multiple-Instance Learning Approach via Bayesian Extreme Learning Machine

PEIPEI WANG[1], XINQI ZHENG[1,2], JUNHUA KU[3], AND CHUNNING WANG[4]

[1]School of Information Engineering, China University of Geosciences, Beijing 100083, China
[2]Technology Innovation Center of Territory Spatial Big-Data, MNR of China, Beijing 100036, China
[3]School of Mathematics, Yibin University, Yibin 644000, China
[4]National Geological Library of China, Beijing 100083, China

Corresponding author: Xinqi Zheng (zhengxq@cugb.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 71673256, in part by the Funds for Basic Scientific Research of Central Universities under Grant 2652020001, in part by Key Research and Development Project of Hainan Province under Grant ZDYF2018234, and in part by the China Geological Survey under Grant DD20190413.

ABSTRACT Multiple-instance learning (MIL) can solve supervised learning tasks, where only a bag of multiple instances is labeled, instead of a single instance. It is considerably important to develop effective and efficient MIL algorithms, because real-world datasets usually contain large instances. Known for its good generalization performance, MIL based on extreme learning machines (ELM–MIL) has proven to be more efficient than several typical MIL classification methods. ELM–MIL selects the most qualified instances from each bag through a single hidden layer feedforward network (SLFN) and trains modified ELM models to update the output weights. This learning approach often performs susceptible to the number of hidden nodes and can easily suffer from over-fitting problem. Using Bayesian inferences, this study introduces a Bayesian ELM (BELM)-based MIL algorithm (BELM–MIL) to address MIL classification problems. First, weight self-learning method based on a Bayesian network is applied to determine the weights of instance features. The most qualified instances are then selected from each bag to represent the bag. Second, BELM can improve the classification model via regularization of automatic estimations to reduce possible over-fitting during the calibration process. Experiments and comparisons are conducted with several competing algorithms on Musk datasets, images datasets, and inductive logic programming datasets. Superior classification accuracy and performance are demonstrated by BELM–MIL.

INDEX TERMS Multiple-instance learning, Bayesian extreme learning machine, instance selection, classification.

I. INTRODUCTION

Multiple-instance learning (MIL) was firstly proposed by Dietterich *et al.* [1] as an approach to predict drug activities. Afterwards, MIL enjoyed many successful applications, such as improved drug activity prediction [2], text categorization [3], image classification [4], object detection [5], and stock prediction [6]. Different from other machine-learning frameworks, MIL is novel, because it contains bags with labels, instead of labeled instances. There are unfixed quantities of instances in each bag, and the label of each bag is observable, whereas the instances in each bag are not. By rule, if there is at least one positive instance in the bag, the label is positive. Otherwise, it is negative. This is important, because traditional single-instance learning (SIL) approaches no longer solve MIL problems.

Over the past few decades, substantial MIL methods have emerged to solve multi-instance problems. All can be generalized into three types:

- *Specialized algorithms combined with the characteristics of MIL.* This includes the axis-parallel rectangle approach, which finds the most appropriate axis parallel rectangle by taking the property values of the instance [1]. In 1998, Maron and Lozano–Perez presented the diverse density (DD) method [7], which took the instance having the largest diverse density from each bag as the based instance, then calculated the distance between this based instance and the instance in test bag to predict the bag label. In 2002, Zhang and Goldman [8] extended DD with an expectation

The associate editor coordinating the review of this manuscript and approving it for publication was Jenny Mahoney.

maximized (EM) model by calculating the based instance using the expectation function.

- *MIL algorithms based on classification algorithms.* Researchers have added relevant MIL constraints in the objective function of SIL, including citation K-nearest neighbor (k-NN) [9], which extends k-NN, and Andrews *et al.* [10] developed a support vector machine (SVM) by adding MIL constraints to its object function. Zhou and Zhang [11] added a bag constraint from the traditional back-propagation (BP) neural network and created a BP-mixed-integer programming algorithm by defining a new error function.
- *Algorithms based on the idea of problem transformation.* These include MIL via embedded instance selection (MILES), proposed by Yixin *et al.* [12]. One embeds the bag into the instance space, obtains the characteristics of the bag, and then trains the instance using 1-SVM classifier. Fu *et al.* [13] developed a novel method based on adaptive instance selection called MIL with instance selection (MILIS) by combining the instance selection and the classifier learning steps, providing an alternate optimization framework to ensure iterative convergence. In MILIS, The initial instance selection on a negative bag is based on a simple and efficient kernel density estimate (KDE) method. In 2015, Wang *et al.* [14] first developed an instance selection method based on the extreme learning machine (ELM).

In real-world applications, MIL algorithms are usually time-consuming and can hardly achieve high performance when confronting with potentially large amounts of instances. Therefore, it becomes essential to select the most informative and contributing instance from each bag by the most efficient way. Selecting the most effective classifier is crucial, too. Fig. 1 illustrates the main idea of instance selection-based MIL.

Over the past few years, ELM [15] has been increasingly applied to various machine-learning problems and has shown outstanding generalization performance and extremely fast learning speeds [16]–[20]. ELM proposes the parameters of random initialization hidden layer, and the output weights are determined by a least mean-squares method based on applying the Moore—Penrose generalized inverse [21]. Thus, the computational cost is much lower than those of other classical algorithms (e.g., gradient descent, global search, genetic algorithm, particle swarm). To address MIL classification problems, a multi-kernel extension of ELM (MKELM) provides an elegant way to circumvent calculation of the hidden layer outputs and inherently encode it in a kernel matrix [22]. However, the potential issue of MKELM is how to find an optimal balance between different kernels for a specific application since the kernel balance parameter is typically data dependent. Wang *et al.* [14] modified the specific error function for the characteristics of multiple instance problems and proposed ELM-MIL. Although ELM-MIL can learn much faster, the effectiveness of ELM largely depends

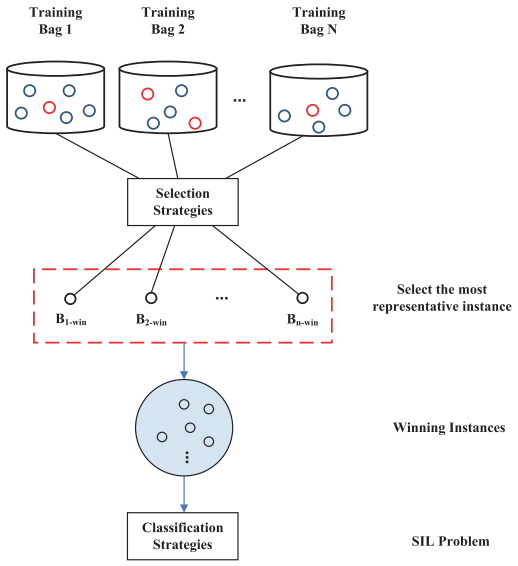


FIGURE 1. Framework of instance selection-based MIL. There are N training bags, and the red circles in each bag represent positive instances, whereas the blue circles represent negative ones. We first select the most representative instance from each bag using selection strategies. Then, the winning instances can be used to train the classifier. Finally, MIL problems can be converted to SIL.

on the number of hidden nodes selected (i.e., network structure). Its classification accuracy is slightly worse and more susceptible to hidden nodes than other benchmark MIL algorithms. ELM-MIL can easily suffer from over-fitting because of the calculation of output weights of the single-hidden layer feed-forward neural network proposed by Moore—Penrose, which generalizes the inverse minimum mean method.

Based on the above observation, we introduce a Bayesian ELM (BELM) [23] based MIL algorithm (BELM-MIL) to further improve classification performance. Using Bayesian inferences, BELM-MIL can alleviate over-fitting problems with an appropriate prior and an automatic regularization, showing good classification performance with many learning tasks [24]. On the premise of instance selection, this method uses the weight self-learning method based on Bayesian network to learn the weight of each feature in the instance and calculate the mathematical expectation of each instance in the bag according to the weight. Then, a representative instance is selected from each bag and we train the classifier using BELM. Compared to existing problem transformation-based MIL methods, BELM-MIL effectively enhances the accuracy of classification by combining BELM with MIL. To the best of our knowledge, this is the first work combining both BELM and MIL in a unified work. The main contribution of this paper can be summarized as follows:

- We propose BELM-MIL for classification.
- We use a weight self-learning method based on Bayesian networks for instance-feature weight optimization.
- We improve instance selection strategies for optimizing the representativeness of instances.

- We conduct experiments on 13 real-world datasets, and the experimental results reveal superior BELM-MIL performance.

The rest of this paper is structured as follows. In Section II, we review the methods related to our work. Section III describes the details of our proposed method (i.e., BELM-MIL). Section IV presents the experiments of the method and discusses its comparison with other baseline approaches. Finally, in Section V, the main idea of our algorithm is concluded.

II. RELATED WORKS

A. ELM

ELM is a single hidden layer feed-forward network (SLFN) first proposed by Huang [15]. Because the input-weights (i.e., connections between the input layer and the hidden layer) and the hidden biases are chosen arbitrarily, ELM only needs to set the number of hidden-layer nodes for algorithm execution. Thus, it can achieve extremely fast learning speeds, better than BP, and it has better generalization performance. However, it has difficulty generating local optimal solutions. The structure of ELM is illustrated in Fig. 2.

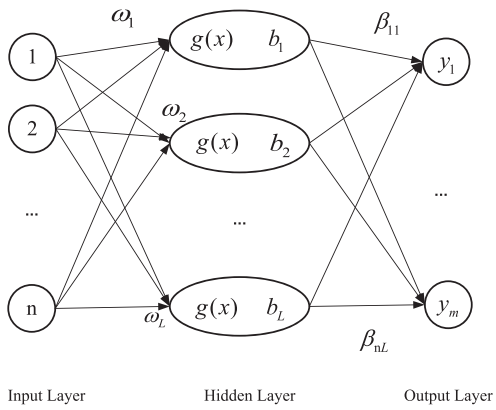


FIGURE 2. ELM, which consists of an input layer, a single hidden layer, and an output layer. The input and hidden layers are connected by input weight ω_i, and the hidden layer and output layer are connected by output weight β. Each hidden neuron has a bias of b_i.

Suppose that there is a training set, $(\boldsymbol{x_1}, \ldots, \boldsymbol{x_N})^{\boldsymbol{T}}$, with label $(y_i, \ldots, y_N)^T$. The output of SLFNs having L nodes can be simply formulated as $\boldsymbol{o}(\boldsymbol{x}) = [o_1(\boldsymbol{x}), \ldots, o_L(\boldsymbol{x})]$, where $\boldsymbol{x}$ is the input instance. The structure of SLFNs can be denoted as follows:

$$\boldsymbol{o_i} = \sum_{j=1}^{L} \boldsymbol{\beta_j} g(\boldsymbol{\omega_j} \cdot \boldsymbol{x_i} + b_j), \quad i = 1, \ldots, N \tag{1}$$

where $g(x)$ is the sigmoid active function that can be denoted as follows:

$$g(\boldsymbol{x}) = \frac{1}{1 + exp(-(\boldsymbol{\omega} \cdot \boldsymbol{x} + \boldsymbol{b}))} \tag{2}$$

where input weight $\boldsymbol{\omega}$ is an $L \times N$ matrix, of which the ith row vector can be presented as: $\boldsymbol{\omega_i} = [\omega_{i1}, \omega_{i2}, \ldots, \omega_{in}]$, and $\boldsymbol{b}$ is an $L \times 1$ hidden bias vector, where b_i is the bias of the ith hidden layer unit. Suppose that $\boldsymbol{\beta}$ is the output weight. The structure of SLFNs can then be simply written as

$$\boldsymbol{O} = \boldsymbol{H\beta} \tag{3}$$

where $\boldsymbol{O}$ is the output matrix: $\boldsymbol{O} = [\boldsymbol{o_1}, \boldsymbol{o_2}, \ldots, \boldsymbol{o_N}]^{\boldsymbol{T}}$, and $\boldsymbol{H}$ is called the $N \times L$ hidden-layer output matrix of the network, whose elements are as follows:

$$\boldsymbol{H} = \begin{bmatrix} g(\boldsymbol{\omega_1} \cdot \boldsymbol{x_1} + b_1), & \cdots & g(\boldsymbol{\omega_L} \cdot \boldsymbol{x_1} + b_L) \\ \vdots & \ddots & \vdots \\ g(\boldsymbol{\omega_1} \cdot \boldsymbol{x_N} + b_1), & \cdots & g(\boldsymbol{\omega_L} \cdot \boldsymbol{x_N} + b_L) \end{bmatrix}_{N \times L} \tag{4}$$

and $\boldsymbol{\beta} = [\boldsymbol{\beta_1}, \ldots, \boldsymbol{\beta_L}]$, where the ith row vector is $\boldsymbol{\beta_i} = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{in}]$, which can be determined by using generalized Moore–Penrose inverse as follows:

$$\hat{\boldsymbol{\beta}} = \boldsymbol{H}^{\dagger}\boldsymbol{Y} \tag{5}$$

$\boldsymbol{H}^{\dagger}$ is the Moore–Penrose generalized inverse of matrix $\boldsymbol{H}$, and $\boldsymbol{H}^{\dagger} = (\boldsymbol{H} \cdot \boldsymbol{H})^{-1}$.

B. BELM

Bayesian inference provides a natural way to automatically optimize model complexity [25], [26]. BELM optimizes the output layer weights based on the Bayesian linear method, which improves the over-fitting problem of ELM [23]. With the wide application of Bayesian theory, we can conclude that the establishment of most models based on Bayesian theory can be divided into the following two steps. The first step finds the posterior distribution of the model parameters. Suppose $\boldsymbol{\omega}$ is the set of parameters, and $\boldsymbol{D}$ is the dataset. Because the prior distribution and likelihood function of the model is proportional, the prior distribution can be simply defined as:

$$P(\boldsymbol{\omega}|\boldsymbol{D}) = P(\boldsymbol{\omega}) \cdot P(\boldsymbol{D}|\boldsymbol{\omega}) \tag{6}$$

The second step is that the output, y_{new}, can be given by the integral of the posterior distribution of $\boldsymbol{\omega}$ for input instances of $\boldsymbol{x_{new}}$:

$$P(y_{new}|\boldsymbol{x_{new}}, \boldsymbol{D}) = \int P(y_{new}|\boldsymbol{x_{new}}, \boldsymbol{\omega}) \cdot P(\boldsymbol{\omega}|\boldsymbol{D}) d\omega \tag{7}$$

The Gaussian normalization of BELM differs from ELM in that the regularization α is a natural consequence of the Gaussian Process. In contrast, other approaches require a term for the cost function being minimized. Therefore, BELM can achieve better generalization than ELM.

ELM learns fast and has a good generalization capability with a simple network structure. The network weights are calculated using the Moore–Penrose generalized inverse matrix, and the weights between the input and hidden layers and the bias of the hidden neuron are assigned randomly, without adjustment during the training process. Thus, we can get the

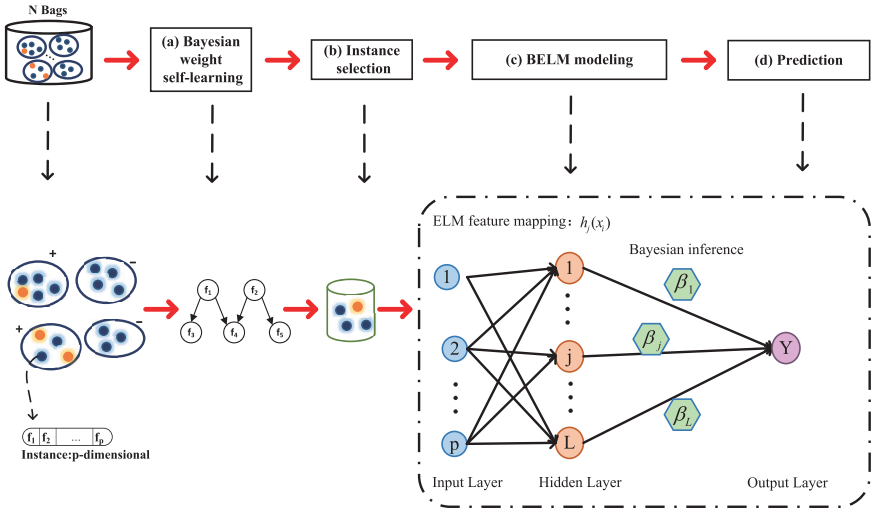


FIGURE 3. A conceptual view of BELM-MIL. Given N training bags, each bag contains several instances, and there are four main steps to classifier training: Bayesian weighted self-learning; instance selection; BELM modeling; and prediction.

optimal solution by only setting the number of hidden nodes. BELM covers these advantages of ELM, and it reduces the computational cost while avoiding establishing confidence intervals through the guidance method.

III. PROPOSED METHOD

Suppose that there are N bags in the training set, $\boldsymbol{B} = \{B_1, \ldots, B_N\}$, the number of instances in the ith bag, B_i, is n_i. The jth instance in B_i is x_{ij}, which belongs to a p-dimensional space (i.e., $[x_{ij1}, \ldots, x_{ijp}]^T$). Suppose that $\boldsymbol{Y} = \{y_1, \ldots, y_N\}$ is the label set where y_i is the label of bag B_i, and $y_i = +1$ denotes the label of ith bag (positive), whereas $y_i = -1$ represents the label of the ith bag (negative). The purpose of our work is to predict whether the label of the new input bag is positive.

The conceptual view of BELM-MIL is shown in Fig. 3. BELM–MIL is achieved through four primary steps: Bayesian weight self-learning; instance selection; BELM modeling; and prediction. Algorithm 1 reports the detailed process of the proposed BELM–MIL.

A. WEIGHT SELF-LEARNING BASED ON BAYESIAN NETWORK

Because it can obtain the feature weight through the study of instances, the learning process is not affected by human factors, and the weight determination is intelligent, Bayesian networks are applied to iteratively determine weights [27], [28]. Bayesian networks are usually defined by two parts [29]. The first part is a directed acyclic graph, where each node represents a random variable, and each arc represents a probability dependency. The second part is the conditional probability table (CPT) for each attribute, which is the conditional probability distribution of the attribute under its parents.

Algorithm 1 BELM–MIL: Multiple Instance Learning via Bayesian Extreme Learning Machine

Require:
The number of hidden nodes, L;
The set of training data, $\boldsymbol{B}$;
The label of the training dataset, $\boldsymbol{Y}$;
The set of testing data, $\hat{\boldsymbol{B}}$;
A constant number, k;

Ensure:
The class label, $\hat{\boldsymbol{Y}}$, of testing bags, $\hat{\boldsymbol{B}}$;

//Training phases:

1: Update feature weight $\boldsymbol{\omega}$ by weight self-learning method (Section III-A) using (9)-(10) :
2: Calculate mathematical expectation Y_{ij} of each instance, x_{ij}, in bag B_i according to (11), and select instance x_{ij} having $max(Y_{ij})$ to represent bag B_i.
3: Suppose y has an independent noise, ξ_i. Apply the probability model, $P(y|\boldsymbol{H}, \beta, \sigma^2)$, to the bag-level instance (Section III-C). Then, define a prior distribution, $p(\boldsymbol{\beta}|\alpha)$, to punish excessive weight.
4: Initialize the shared prior, α, and the variance of y, and update α and σ^2 iteratively according to (16)-(19) until convergence.

//Testing phases:

5: Select instance $\hat{\boldsymbol{x}}_{\boldsymbol{ij}}$ having $max(Y_{ij})$ to represent bag $\hat{B}_i$.
6: Predict $p(\hat{y}|\alpha, \sigma^2, \boldsymbol{h}(\hat{\boldsymbol{x}}))$ based on $\boldsymbol{m}$ in *Step 4*.
7: Apply the classifier on the testing dataset, $\hat{\boldsymbol{B}}$, to predict the class label, $\hat{\boldsymbol{Y}}$.
8: **return** $\hat{\boldsymbol{Y}}$.

For the selection of the representative instance in each bag, we suppose each instance has a label (i.e., positive or negative), and the goal of the Bayesian Network is to maximize

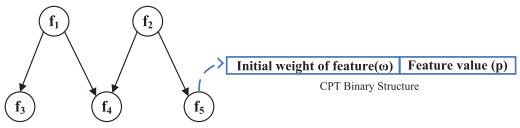


FIGURE 4. Illustration of Bayesian network for weight self-learning problem and CPT Binary structure for feature f_5. Each node represents a feature of instance (i.e., $f_1, f_2, \ldots, f_p$) which is denoted e in the probability presentation of Bayesian network. The line represents the causal relationship between the features of the upper-level and the next level. The CPT of this Bayesian network is a binary structure which stores the initial weight ω and its feature value p.

the total probability obtained from the training set. Thus, we construct a Bayesian network for weight self-learning optimization, as is shown in Fig. 4. The CPT here presents not the conditional probability table of the features, but the initial weight and feature value relative to its parents, i.e., a binary structure. The calculation of CPT items in Bayesian networks is similar to the calculation probability involved in naive Bayesian classification [30].

Here, the Bayesian Network is trained by using a gradient-descent algorithm [31], the purpose of which is to learn the value of CPT. Suppose $\boldsymbol{S}$ is the set of training instances $\boldsymbol{X_1, X_2, \ldots, Xs}$, and $\boldsymbol{\omega} = [\omega_1, \omega_2, \ldots, \omega_p]^T$ is the initial weight of the feature. The prior probability of node e at $\boldsymbol{X_d}$ is $p(e|\boldsymbol{X_d})$, i.e., the feature value in binary structure of CPT. The goal of learning is to maximize $P_\omega(\boldsymbol{S}) = \prod_{d=1}^{S} P_\omega(\boldsymbol{X_d})$, where $P_\omega(\boldsymbol{S})$ is the total probability of training instances $\boldsymbol{S}$, and $P_\omega(\boldsymbol{X_d})$ is the total probability of instance $\boldsymbol{X_d}$. With $\ln P_\omega(\boldsymbol{S})$ gradient calculation, it makes the problem easier:

$$\begin{aligned} lnP_\omega(\boldsymbol{S}) &= \ln \prod_{d=1}^{S} P_\omega(\boldsymbol{X_d}) \\ &= \ln(\frac{\omega}{p(e|\boldsymbol{X_1})} \cdot \ldots \cdot \frac{\omega}{p(e|\boldsymbol{X_d})}) \\ &= \ln \frac{\omega}{p(e|\boldsymbol{X_1})} + \ldots + \ln \frac{\omega}{p(e|\boldsymbol{X_d})} \end{aligned} \tag{8}$$

We initialize $\boldsymbol{\omega}$ randomly and update it by using a gradient-descent algorithm [31]. The iterative process can be thus derived.

(a) Calculate the gradient for each attribute in the network according to (8):

$$\begin{aligned} \frac{\partial \ln P_\omega(\boldsymbol{S})}{\partial \omega} &= \frac{p(e|\boldsymbol{X_1})}{\boldsymbol{\omega}} + \frac{p(e|\boldsymbol{X_2})}{\boldsymbol{\omega}} + \ldots + \frac{p(e|\boldsymbol{X_d})}{\boldsymbol{\omega}} \\ &= \sum_{d=1}^{S} \frac{p(e|\boldsymbol{X_d})}{\boldsymbol{\omega}} \end{aligned} \tag{9}$$

(b) Move a small step along the gradient to update weight $\boldsymbol{\omega}$:

$$\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} + (l)\frac{\partial \ln P_\omega(S)}{\partial \omega} \tag{10}$$

where l is step learning rate, a very small value.

(c) Then, $\boldsymbol{\omega}$ can be normalized to (0, 1).

(d) When $\frac{\partial \ln P_\omega(\boldsymbol{S})}{\partial \omega} < k$, the training of $\boldsymbol{\omega}$ will end, otherwise, go to (a).

B. INSTANCE SELECTION

For each instance, x_{ij}, in bag B_i, the mathematical expectation can be calculated as follows:

$$Y_{ij} = \sum_{k=1}^{p} x_{ijk} * \omega_k \tag{11}$$

Then, we select instance x_{ij}, which has $max(Y_{ij})$ to represent bag B_i. Thus, the MIL problem can be converted to SIL.

C. BELM MODELING

For the selected instances, $(x_1, x_2, \ldots, x_N)$, suppose each label y has an independent noise, ξ, that follows a Gaussian distribution, $\mathcal{N}(0; \sigma^2)$. Then, y can be written as

$$y = \boldsymbol{H\beta} + \xi \tag{12}$$

where $H \in \mathbb{R}^{(N \times L)}$ is the output weight of the hidden layer. The probability model and likelihood function can then be derived as follows:

$$\begin{aligned} P(y|\boldsymbol{H}, \beta, \sigma^2) &= \prod_{i=1}^{N} p(y_i|H, \boldsymbol{\beta}, \sigma^2) \\ &= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(y_i - h(\boldsymbol{x_i})\boldsymbol{\beta})^2}{2\sigma^2}) \end{aligned} \tag{13}$$

To punish excessive weight, the natural priori distribution is defined as follows:

$$p(\boldsymbol{\beta}|\alpha) = \mathcal{N}(0; \alpha^{-1}\boldsymbol{I}) = (\frac{\alpha}{2\pi})^{\frac{L}{2}} exp(-\frac{\alpha}{2}\boldsymbol{\beta^T\beta}) \tag{14}$$

where $\boldsymbol{I}$ is the identity matrix, and α is the shared prior, which is a natural consequence of the Guassian approach. Because the prior distribution and likelihood function follow a Gaussian distribution, the posterior can be derived as follows:

$$p(\boldsymbol{\beta}|\alpha, \sigma^2, y) = \frac{p(y|\beta, \sigma^2)p(\beta|\alpha)}{p(y|\alpha, \sigma^2)} \tag{15}$$

which also follows a Gaussian distribution with a mean value of $\boldsymbol{m}$ and a variance of $\boldsymbol{v}$, as follows:

$$\boldsymbol{m} = \sigma^{-2} \cdot \boldsymbol{v} \cdot \boldsymbol{H^T} Y \tag{16}$$

$$\boldsymbol{v} = (\alpha\boldsymbol{I} + \sigma^{-2} \cdot \boldsymbol{H^T} \cdot \boldsymbol{H})^{-1} \tag{17}$$

Thus, we can get the optimal value of hyper-parameter via the evidence procedure of [32], and optimal conditions can be calculated by following the fixed-point formula:

$$\alpha \leftarrow \frac{L - \alpha \cdot trace[v]}{\boldsymbol{m^T m}} \tag{18}$$

$$\sigma^2 \leftarrow \frac{\sum_{i=1}^{N}(y_i - h(x_i)\boldsymbol{m})^2}{N - L + \alpha \cdot trace[\boldsymbol{v}]} \tag{19}$$

where N is the number of patterns. By initializing α and σ^2, $\boldsymbol{m}$ and $\boldsymbol{v}$ are updated iteratively according to (16)-(19) until convergence. Finally, $\boldsymbol{m}$ can be applied to predict the new output, y_{new}, based on input $\boldsymbol{x_{new}}$.

D. PREDICTION

For the testing dataset, $\hat{\boldsymbol{B}}$, the representative instances can be selected the same way as the training data. Then the label of each bag can be predicted:

$$p(\hat{y}|\alpha, \sigma^2, \boldsymbol{h}(\hat{\boldsymbol{x}})) = \mathcal{N}(\boldsymbol{h}(\hat{\boldsymbol{x}})\boldsymbol{m}; \hat{\sigma^2}) = \int p(\hat{y}|\beta, \sigma^2, \boldsymbol{h}(\hat{\boldsymbol{x}}))p(\beta|\alpha, \sigma^2, y)d\beta \quad (20)$$

where $\boldsymbol{h}(\hat{\boldsymbol{x}}) = [h_1(\hat{\boldsymbol{x}}), \ldots, h_L(\hat{\boldsymbol{x}})]$, and $\sigma^2(\hat{\boldsymbol{x}}) = \sigma^2 + h(\hat{\boldsymbol{x}})^T \cdot \boldsymbol{v} \cdot h(\hat{\boldsymbol{x}})^T$. Equation (20) also follows a Gaussian distribution with the mean value and variance, as follows:

$$\hat{m} = h(\hat{\boldsymbol{x}}) \cdot m \quad (21)$$

$$\hat{\sigma^2} = \sigma^2 + h(\hat{\boldsymbol{x}}) \cdot \boldsymbol{v} \cdot \boldsymbol{h}(\hat{\boldsymbol{x}})^T \quad (22)$$

Finally, the class label, $\hat{\boldsymbol{y}}$, is mapped to (0, 1) using the Sigmoid function [33]. If $\hat{y} > 0.5$, the label of this bag is positive. Otherwise, it is negative.

IV. EXPERIMENTS

A. EXPERIMENTAL SETTING

In this section, two groups of experiments are conducted to analyze the performance of our proposed method. The first experiment reports the three types of MIL algorithms mentioned in Section I and demonstrates the feasibility of studying a MIL algorithm based on the idea of problem transformation. The second experiment analyzes the performance between BELM–MIL and baselines. Our experiments involves two parameters. The number of hidden nodes, L, is set to 50, and the regular parameter, C, in ELM–MIL is set to 0.25 [14]. All experiments are executed on a computer using an Intel(R) Core(TM) i5-2450M 2.50-GHz central processing unit with 8-GB RAM, Matlab R2012a. In our experiments, we use 10-fold cross validation to compare the results. The dataset is divided into 10 parts, among which, nine are taken as training data and one is taken as test data for the experiment. The corresponding correct rate is obtained for each experiment. The average value of 10 results is used to estimate the accuracy of the algorithm. The procedure is repeated 10 times and the averaged accuracy is then computed.

B. BASELINE APPROACHES

For this paper, we choose four baseline approaches for comparison and evaluation, all based on the idea of problem transformation.

- *ELM–MIL*: This algorithm first modifies the specific error function in ELM for the characteristics of MIL problems, then it selects the most qualified instance in each bag using SLFNs. It finally uses selected instances to update the output weight of ELM [14].
- *MILES*: The main idea of MILES is transforming the MIL problem into a supervised learning problem. By defining the similarity measure of the instance in the bag, the training bags can be mapped to the feature space, and the representative features can be selected by the 1-norm SVM to enhance the bag classification [12].
- *MILIS*: This method applies KDE to select an instance for each bag, and then maps each bag to a bag-level feature. It optimizes the classifier formation in an iterative manner that guarantees convergence compared with MILES [13].
- *Bayesian-kNN*: Each bag is considered as an instance of the traditional classification concept. According to the label of the kth nearest neighbors of the bag, the voting method is applied to determine the class of the bag [9].

TABLE 1. Details of experimental datasets, including drug activities, content-based image retrieval and inductive logic programming datasets.

Data set	Bags	Atts	Insts	Pos.bags	Neg.bags	Avg/bag
Musk1	92	166	476	47	45	5.2
Musk2	102	166	6,598	39	63	64.7
Elephant	200	230	1,391	100	100	7.0
Fox	200	230	1,320	100	100	6.6
Tiger	200	230	1,220	100	100	6.1
Cars	800	90	5,600	420	380	7
People	691	90	4,837	311	380	7
Eastwest	20	24	213	10	10	10.7
Westeast	20	24	213	10	10	10.7
Mutagenesis_atoms	188	10	1,618	125	63	8.6
Mutagenesis_bonds	188	16	3,995	125	63	21.3
Mutagenesis_chains	188	24	5,349	125	63	28.5
Suramin	11	22	2,378	7	4	216.2

C. EXPERIMENTAL DATASETS

Three fields of learning tasks across 13 datasets are used to validate BELM–MIL in our experiment. Similar experiments based on those datasets can be found in previous works [34], [35]. The original datasets for drug activity perdition and content-based image retrieval are available online at http://www.miproblems.org. The datasets for the train-bound challenge, and the Surami judgment task can be found at http://www.cs.waikato.ac.nz/~eibe/multi_instance/. Table 1 shows the details of each dataset, and the number of instances and dimensional features are also listed. Moreover, those datasets are introduced briefly as follows:

- **Drug-activity prediction data:** The purpose of drug-activity prediction is to predict the potency of drug molecules. Each drug molecule (i.e., bag) has several low-energy shapes (i.e., instances). Musk1 and Musk2 are the classic datasets in MIL, first proposed by Dietterich *et al.* [1] in their study of drug-active molecular prediction. During the learning procedure, if any of the conformations in the molecule has a musky smell, we define this molecule as a musk. Musk1 consists of 47 positive bags and 45 negative bags with a total of 476 instances. Musk2 has 102 bags, of which 39 are positive and 63 are negative, with 6,598 instances in all. The instances in both datasets are represented using a 166-dimensional feature vector.
- **Content-based image retrieval data:** In content-based image retrieval, the learning task is to determine whether

an image contains the object of interest, such as an elephant or a fox. This kind of retrieval is commonly used for MIL evaluation [36], [37]. Each image represents a bag, and the region of interest in the image is an instance if at least one region (instance) contains the object of interest. Then it is labeled as positive. We use five datasets: Elephant, Fox, Tiger, Cars, and People, corresponding to different types of image recognition. Each of the first three datasets (Elephant, Fox, Tiger) contains 200 bags, consisting of 100 positive bags and 100 negative bags with a total of 1,391, 1,320, and 1,220 instances, respectively. Each instance in those datasets is described with a 230-dimensional feature vector. Cars contains 5,600 instances grouped into 800 bags, of which 420 bags are positive and 380 are negative. People has 691 bags with 4,837 instances in total and consists of 311 positive bags and 380 negative bags. Each instance in Cars and People is represented by a 90-dimensional feature vector.

- **Inductive Logic Programming (ILP) data:** We select six sets of multi-instance concept-related datasets from the ILP: Eastwest, Westeast, Mutagenesis_atoms, Mutagenesis_bonds, Mutagenesis_chains, and Surami. Each bag in the Eastwest and Westeast datasets signifies a train (bags) contains a variable number of cars (instances), because the direction of a train is either west or east. This learning task is under the MIL assumption, and those two datasets are used for MIL performance evaluation [38]–[40]. Each bag in Eastwest and Westeast datasets consists of 10 positive bags and 10 negative bags, and each instance in bags is described by a 20-dimensional feature vector. The positive bags represent eastbound trains in Eastwest dataset, while the positive bags in Westeast dataset denote the opposite direction.

  The Mutagenesis_atoms, Mutagenesis_bonds, and Mutagenesis_chains datasets are applied to predict gene mutations [41], and MIL frameworks have been successfully used to tackle this problem [34]. These three datasets have been converted to MIL problems by using a proper toolbox [42]. A bag represents a compound molecule, atom, and bond in the mutagenesis MIL dataset that contains all 1,618 atoms, all 3,995 atoms, and all 5,349 atomic bond tuples as their instances, respectively. Each bag consists of 125 positive bags and 63 negative bags. The Surami dataset also belongs to the field of medicine, and each drug molecule can be regarded as a bag in the MIL problem [43]. The purpose of our prediction is to judge whether the drug molecule has an anticancer effect. There are 11 drug molecules in total that contain seven positive bags and four negative bags.

D. EVALUATION CRITERION

Classification accuracy (ACC), root mean-squared error (RMES), area-under-the-receiver operating characteristic curve (AUC), and model building time are applied to compare algorithm performance, which are widely used for performance evaluation on domain specific problems [44]–[48]. Finally, we count the number of wins, ties, and losses to evaluate all competitive algorithms.

- *ACC:* As its name suggests, classification accuracy indicates the correct rate: $Accuracy = n/N$, where n is the number of bags whose predicted label is consistent with the original label, and N is the size of the testing bags.
- *AUC:* AUC is the area-under-the-receiver operating characteristic curve, which can be defined as: $AUC = (\sum r_i - M(M+1)/2)/(MN)$, where r_i is the rank of the ith negative instance in the ranked list, and M, N is the number of negative and positive instances, respectively.
- *RMES:* RMES (i.e., the standard error), is very sensitive to very large or very small data in a set of classification results. Thus, it is very suitable for measuring the precision of the algorithm, and it is calculated according to the following formula: $RMES = \sqrt{\sum e^2/N}$, where e is the error between the predicted value and the true value of bags, and N is the size of testing bags. Therefore, the smaller the RMES, the higher the precision of the algorithm and the better the performance.
- *Win/tie/lose:* Win/tie/lose has proven to be an effective measure to help us intuitively understand the victory or defeat of the two competitive algorithms, A and B [49]. Specifically, for a performance measure M, a win/tie/lose record for A and B indicates the number of datasets where A wins, ties, or loses against B on M respectively.

E. RESULTS AND ANALYSIS

1) COMPARISON OF THREE TYPES OF MIL METHODS

Table 2 shows the results of the first group experiments, and we can conclude that the best performing algorithms in these three types of MIL algorithms are MIDD (average classification accuracy is 72.20%), MILR (average classification accuracy is 70.71%), and MILIS (average classification accuracy is 64.30%). The above experimental results show that the development of the first two types of MIL algorithms is quite mature [50], [51], whereas there is still enormous developing space in the research of the third-class MIL algorithm (i.e., the idea of problem transformation-based MIL). Moreover, if the third-class MIL algorithm transforms MIL problem into a SIL problem, the problem is simplified, and the classification efficiency is improved apparently. Therefore, based on the third-class learning algorithm, to learn a method with sufficient classification efficiency and achieve considerable classification accuracy is a feasible research direction [52], [53].

2) ACCURACY COMPARISON BETWEEN BELM–MIL AND BASELINES

Table 3 reports the experimental results in terms of classification accuracy. Moreover, Table 7 illustrates the win/tie/lose

TABLE 2. Comparison of three types of MIL methods.

Data set	DD [a]	MIDD [a]	EMDD [a]	mi–SVM [b]	MI–SVM [b]	MILR [b]	MILES [c]	MILIS [c]	Bayesian-kNN [c]
Muskl	78.64	87.21	87.24	87.41	77.92	72.83	77.78	78.89	89.11
Musk2	75.98	82.45	87.26	83.62	84.36	80.39	72.14	78.15	78.18
Elephant	78.52	77.52	76.88	80.08	73.14	77.54	72.55	74.57	50.36
Fox	67.44	62.89	57.13	57.98	58.85	57.56	58.56	58.11	50.86
Tiger	73.56	75.38	72.10	78.98	82.55	78.54	72.16	71.50	50.90
Cars	69.12	76.84	71.63	59.26	61.88	72.25	52.52	52.53	52.56
People	77.55	77.87	77.71	71.25	72.57	74.67	54.99	54.99	54.99
Eastwest	60.03	65.94	50.00	58.51	60.57	65.35	50.74	50.50	50.43
Westeast	60.16	40.11	25.65	28.73	30.23	50.79	55.28	50.57	50.29
Mutagenesis_atoms	71.28	72.87	71.28	68.57	66.61	72.34	66.49	66.42	73.33
Mutagenesis_bonds	72.34	75.53	71.81	68.43	66.77	75.61	65.42	66.25	69.62
Mutagenesis_chains	76.06	80.32	72.87	65.06	65.52	77.66	66.68	67.56	70.85
Suramin	63.64	63.64	54.55	63.64	63.64	63.64	65.22	65.87	55.68
Average	71.10	**72.20**	67.39	67.04	66.51	**70.71**	63.89	**64.30**	61.32

[a] Specialized algorithms combined with the characteristics of MIL methods.
[b] MIL algorithms based on classification algorithms.
[c] Algorithms based on the idea of problem transformation.

TABLE 3. Classification accuracy (ACC)(%) of different base algorithms. Best results for each dataset are highlighted in bold.

Data set	ELM–MIL	MILES	MILIS	Bayesian-kNN	BELM–MIL
Muskl	86.50	77.78	78.89	89.11	**89.26**
Musk2	84.94	72.14	78.15	78.18	**86.96**
Elephant	76.76	72.55	74.57	50.36	**96.94**
Fox	59.54	58.56	58.11	50.86	**98.10**
Tiger	74.63	72.16	71.50	50.90	**96.82**
Cars	55.12	52.52	52.53	52.56	**70.25**
People	58.31	54.99	54.99	54.99	**78.58**
Eastwest	**55.10**	50.74	50.50	50.43	50.13
Westeast	**55.97**	55.28	50.57	50.29	50.99
Mutagenesis_atoms	67.02	66.49	66.42	**73.33**	66.53
Mutagenesis_bonds	66.49	65.42	66.25	69.62	**78.19**
Mutagenesis_chains	58.51	66.68	67.56	70.85	**80.32**
Suramin	63.64	65.22	**65.87**	55.68	33.36
Average	66.35	63.89	64.30	61.32	**75.11**

TABLE 4. AUC of different base algorithms. Best results for each dataset are highlighted in bold.

Data set	ELM–MIL	MILES	MILIS	Bayesian-kNN	BELM–MIL
Muskl	0.90	0.87	0.86	0.87	**0.92**
Musk2	0.86	0.81	0.87	0.84	**0.89**
Elephant	0.77	0.78	0.82	0.82	**0.92**
Fox	0.59	0.63	0.63	0.64	**0.81**
Tiger	0.73	0.78	0.78	0.82	**0.88**
Cars	0.69	0.58	0.62	0.56	**0.67**
People	0.63	0.57	0.62	0.58	**0.66**
Eastwest	0.60	0.55	0.58	0.56	**0.63**
Westeast	0.63	0.56	0.55	0.56	**0.66**
Mutagenesis_atoms	**0.74**	0.73	0.70	0.56	0.71
Mutagenesis_bonds	0.72	0.72	0.71	0.55	**0.73**
Mutagenesis_chains	0.58	0.72	0.71	0.55	**0.75**
Suramin	0.58	**0.62**	0.62	0.62	0.48
Average	0.69	0.69	0.70	0.66	**0.75**

records of competitive algorithms on classification accuracy. Each win/tie/lose record represents that the number of row-method wins, ties, or losses to the column method on 13 experimental datasets. In summary, we can conclude the results as follows:

- BELM–ELM outperforms ELM–MIL in terms of classification accuracy. To be specifically, BELM–MIL wins 9 datasets and lose 4 datasets compared with ELM–MIL.
- Compared to MILES, MILIS and Bayesian-kNN, BELM–MIL also shows good performance. BELM–MIL wins 10 datasets, ties 1 dataset and loses 2 datasets compared to MILES, and it wins 10 datasets, ties 2 dataset and loses 1 datasets compared to MILIS, while BELM–MIL wins 9 datasets, ties 2 dataset and loses 2 datasets compared to Bayesian-kNN.

3) AUC COMPARISON BETWEEN BELM–MIL AND BASELINES

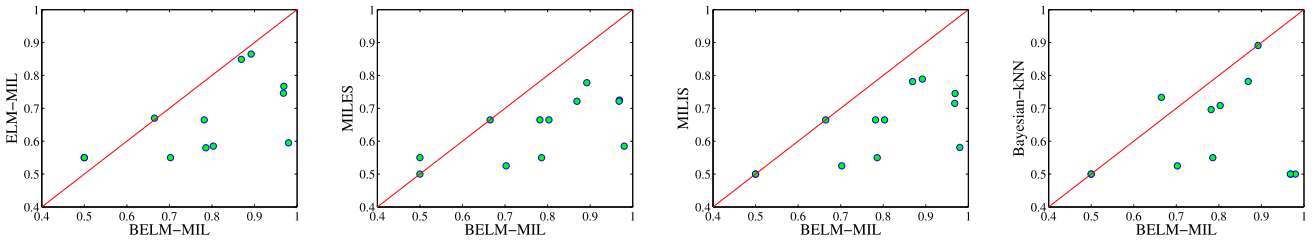
Experimental results in terms of AUC of BELM–MIL and other baseline algorithms are shown in Table 4, and Table 8 illustrates the win/tie/lose records of competitive algorithms on classification accuracy. In details, the results in Table 4 and Table 8 indicate that:

- BELM–ELM significantly outperforms ELM–MIL in terms of AUC. To be exactly, BELM–MIL wins 11 datasets and loses 2 datasets compared with ELM–MIL.
- Compared to MILES, MILIS and Bayesian-kNN, BELM–MIL also shows good performance. BELM–MIL wins 11 datasets, and loses 2 datasets compared to MILES, and it wins 12 datasets, and loses 1 datasets compared to MILIS, while BELM–MIL wins 12 datasets, and loses 1 datasets compared to Bayesian-kNN.

4) RMES COMPARISON BETWEEN BELM–MIL AND BASELINES

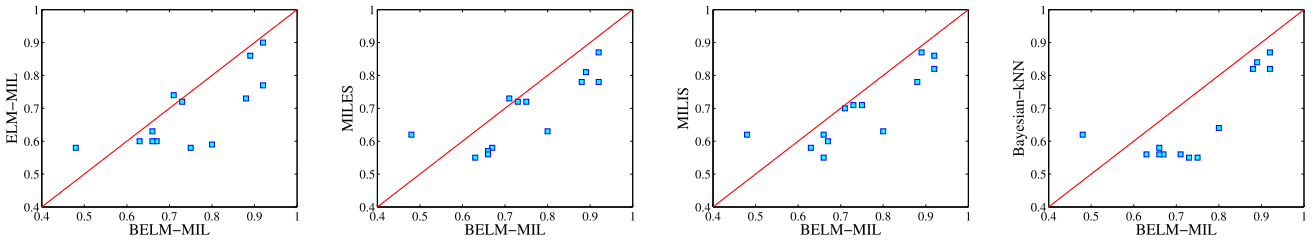
Table 5 and Table 9 illustrate the result details of BELM and other rival algorithms with respect to RMES. Moreover, Fig 7 illustrates the comparison of BELM–ELM and other rival algorithms. Overall, the experimental results can be summarized as follows:

- BELM–ELM significantly outperforms ELM–MIL in terms of RMES. To be precisely, BELM–MIL wins 11 datasets and loses 2 datasets compared with ELM–MIL.
- BELM–MIL shows good precision when compared to MILES, MILIS and Bayesian-kNN. It wins 9 datasets, ties 1 dataset and loses 3 datasets compared to MILES,



(a) BELM–MIL *vs.* ELM–MIL (b) BELM–MIL *vs.* MILES (c) BELM–MIL *vs.* MILIS (d) BELM–MIL *vs.* Bayesian-kNN

FIGURE 5. Comparison between BELM–MIL and rival algorithms (ELM–MIL, MILES, MILIS and Bayesian-kNN) on 13 datasets. Each data point in a figure represents the classification accuracy (ACC) on one dataset.



(a) BELM–MIL *vs.* ELM–MIL (b) BELM–MIL *vs.* MILES (c) BELM–MIL *vs.* MILIS (d) BELM–MIL *vs.* Bayesian-kNN

FIGURE 6. Comparison between BELM–MIL and rival algorithms (ELM–MIL, MILES, MILIS and Bayesian-kNN) on 13 datasets. Each data point in a figure represents the AUC on one dataset.

TABLE 5. RMES of different base algorithms. Best results for each dataset are highlighted in bold.

Data set	ELM–MIL	MILES	MILIS	Bayesian-kNN	BELM–MIL
Musk1	0.35	0.52	**0.33**	0.50	0.38
Musk2	0.40	0.43	**0.32**	0.51	0.44
Elephant	0.39	0.46	0.41	0.47	**0.35**
Fox	0.63	0.64	0.54	0.73	**0.53**
Tiger	0.39	0.50	0.46	0.42	**0.34**
Cars	0.49	0.46	0.46	0.62	**0.44**
People	0.42	0.44	0.44	0.52	**0.39**
Eastwest	0.64	**0.59**	0.65	0.64	0.62
Westeast	0.64	0.71	0.73	0.84	**0.63**
Mutagenesis_atoms	0.58	**0.42**	0.51	0.58	0.45
Mutagenesis_bonds	0.57	0.41	0.44	0.58	**0.40**
Mutagenesis_chains	0.55	0.39	0.43	0.58	**0.39**
Suramin	0.60	0.60	0.68	0.60	**0.52**
Average	0.51	0.51	0.49	0.58	**0.45**

TABLE 6. Modeling time (min) of different base algorithms.

Data set	ELM–MIL	MILES	MILIS	Bayesian-kNN	BELM–MIL
Musk1	0.120	0.520	0.039	1.100	0.960
Musk2	18.000	84.630	1.560	140.000	32.000
Elephant	1.922	9.105	0.168	15.062	3.443
Fox	1.823	8.634	0.159	14.283	3.265
Tiger	1.685	7.980	0.147	13.201	3.017
Cars	1.933	9.158	0.169	15.149	3.463
People	1.933	9.158	0.169	15.149	3.463
Eastwest	2.942	13.933	0.257	23.048	5.268
Westeast	2.942	13.933	0.257	23.048	5.268
Mutagenesis_atoms	2.378	11.264	0.208	18.633	4.259
Mutagenesis_bonds	5.869	27.800	0.512	45.987	10.512
Mutagenesis_chains	7.858	37.220	0.686	61.568	14.073
Suramin	59.711	282.817	5.214	467.832	106.938

wins 11 datasets, and loses 2 datasets compared to MILIS, and wins 13 datasets, and loses 0 datasets compared to Bayesian-kNN.

TABLE 7. Win/tie/lose records of each competing algorithm on classification accuracy (ACC).

Algorithm	ELM–MIL	MILES	MILIS	Bayesian-kNN
MILES	2/2/9			
MILIS	2/1/10	3/7/3		
Bayesian-kNN	4/0/9	5/3/5	5/4/4	
BELM–MIL	**9/0/4**	**10/1/2**	**10/2/1**	**9/2/2**

TABLE 8. Win/tie/lose records of each competing algorithm on AUC.

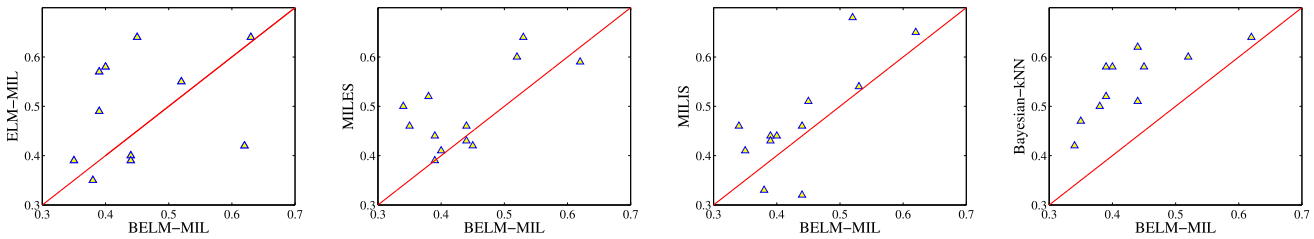
Algorithm	ELM–MIL	MILES	MILIS	Bayesian-kNN
MILES	5/1/7			
MILIS	7/0/5	5/3/5		
Bayesian-kNN	4/0/8	6/3/4	4/2/6	
BELM–MIL	**11/0/2**	**11/0/2**	**12/0/1**	**12/0/1**

TABLE 9. Win/tie/lose records of each competing algorithm on RMES.

Algorithm	ELM–MIL	MILES	MILIS	Bayesian-kNN
MILES	5/1/7			
MILIS	7/0/6	5/2/6		
Bayesian-kNN	0/3/10	1/1/10	3/0/10	
BELM–MIL	**11/0/2**	**9/1/3**	**11/0/2**	**13/0/0**

5) ANALYSIS AND DISCUSSION

Figure 5, 6, and 7 illustrate the ACC, AUC, and RMES comparisons of BELM–ELM and other rival algorithms. For ACC and AUC comparison, if the data point falls below the diagonal line, $y = x$, it means that BELM–ELM has higher performance than the Y-axis algorithm. However, for RMES comparison, if the data points falls above the diagonal line, it signifies higher performance. As those pictures show, most

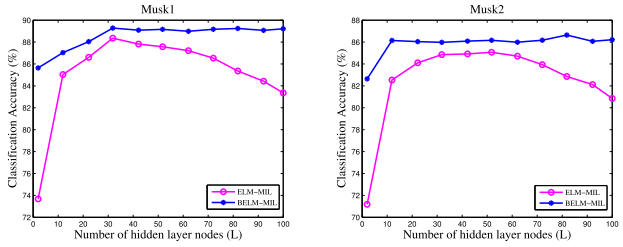


(a) BELM–MIL *vs.* ELM–MIL (b) BELM–MIL *vs.* MILES (c) BELM–MIL *vs.* MILIS (d) BELM–MIL *vs.* Bayesian-kNN

FIGURE 7. **Comparison between BELM–MIL and rival algorithms (ELM–MIL, MILES, MILIS and Bayesian-kNN) on 13 datasets. Each data point in a figure represents the RMES on one dataset.**

data points fall in the desired region, illustrating the superior performance of BELM–MIL.

Among all those instance-selection-based MIL methods, BELM–MIL often outperforms MILES, which is a bag-mapping instance selection approach. For instance, as shown in Tables 3 and 4, BELM–MIL achieves a 89.26% ACC and a 0.92 AUC on Musk1, which is much higher than MILES's 77.87% ACC and 0.87 AUC, and with similar observations for the other dataset, like Elephant, Fox, and Mutagenesis_bonds. Although MILES makes full use of the instances for bag mapping, not all of them contribute to the final classification performance. Thus, BELM—MIL is superior. By contrast, BELM–ELM can select the most valuable one (i.e., instance pruning) and reduce the influence of unfavorable instances.



(a) Musk1 (b) Musk2

FIGURE 8. **The predict accuracy of ELM–MIL and BELM–MIL on MUSK1 and MUSK2 with different number of hidden layer nodes.**

Because the performance of ELM usually depends on the selection of the number of hidden nodes to some extent [54], we compare the classification accuracy of BELM–MIL and ELM–MIL on Musk1 and Musk2 of a varying number of hidden-layer nodes. As is shown in Fig. 8, ELM–MIL easily obtains a small or large value when using a different number of L. Meanwhile, by exploiting probabilistic inferences, BELM–MIL can achieve more stable classification accuracy and control the model complexity successfully with an increase of L. As a result, our proposed BELM–MIL method achieves more robust accuracy than ELM–MIL and further improves MIL classification tasks.

It is worth noting that the ACC and AUC of BELM–MIL in content-based image retrieval are apparently superior, demonstrating that the BELM–MIL can select the most proper instance for model training. Specifically, BELM–MIL gains 96.94% ACC, 0.92 AUC on Elephant, 98.10% ACC, 0.81 AUC on Fox 96.82% AUC, 0.88 AUC on Tiger, 70.25% ACC, 0.67 AUC on Cars, and 78.58% ACC, 0.66 AUC on People. BELM–MIL is effective on content-based image retrieval, which suggests that BELM–MIL can apply those selected instances to optimize image retrieval accurately.

We also conduct experiments to compare time consumptions. Generally, when compared with MILES and ELM–MIL, as Table 6 reports, ELM–MIL and MILIS can run a bit faster than our proposed method, because of the lack of a weight-updating process. Specifically, we analyse the time complexity. For ELM–MIL, the training process mainly includes the following two parts: selecting the representative instance from each bag via SLFNs, and training the network of ELMs. The time complexity of ELM–MIL can easily be calculated:

$$O(N_b \cdot N_i \cdot n \cdot L + N_b \cdot L) \leq O(N_b \cdot N_i \cdot n \cdot L) \tag{23}$$

where N_b is the number of bags, N_i is the number of instances of each bag, L is the number of hidden-layer nodes, and n is the number of attributes in the dataset. According to Fig. 3, BELM–MIL mainly includes four steps, and the time complexity of BELM–MIL can be calculated as

$$\begin{aligned} &O(N_b \cdot N_i \cdot n \cdot N_p + N_b \cdot L \cdot N_q) \\ &\quad \leq O(N_b \cdot (N_i \cdot n \cdot N_p + N_i + L \cdot N_q)) \\ &\quad \leq O(N_b \cdot N_i^2 \cdot n \cdot L \cdot N_p \cdot N_q) \end{aligned} \tag{24}$$

where N_p is the number of iterations of weight self-learning, and N_q is the number of iterations of BELM classifier training. By contrast, MILES and Bayesian-kNN are inferior to our algorithm. The ACC and AUC of BELM–MIL on those benchmark datasets are much higher than ELM–MIL, MILIS, MILES, and Bayesian-kNN, and the RMES of BELM–MIL usually has a smaller value. In other words, BELM–MIL is a better algorithm overall and needs a little more running time, which is attributed to the combination of weight update method and Bayesian classifier training used in BELM–MIL.

V. CONCLUSION

In this paper, a BELM–MIL is proposed. To select the most representative instance in each bag, the Bayesian self-learning method is applied to update the weight of features. Based on new feature weight, the instance is selected from each bag as the input of BELM to train the classifiers. The experiments in Section IV demonstrate that our algorithm performs comparably with baselines on the Musk and other benchmark datasets. Because Bayesian inference provides a natural way to automatically optimize model complexity, by using Bayesian inference, BELM–MIL achieves stable performance with an increase of hidden nodes.

However, as shown in Fig. 8, the accuracy of BELM–MIL is more likely to achieve unstable performance, especially with small values of L. Sparse Bayesian learning has proven to be effective for redundant information reduction by exploiting an automatic relevance determination prior [54]–[56]. Full Bayesian method of ELM also has more advantages than BELM, for example, the variational approximation inference is employed in the Bayesian model to compute the posterior distribution and the independent variational hyperparameters approximately, which can be used to select the hidden nodes automatically, and it can finally achieve more stable performance with more compact architectures [57]. The MIL classification performance of ELM and sparse Bayesian learning and MIL classification based on Full Bayesian method of ELM are worthy of further study. On the other hand, BELM–MIL didn't run enough fast as ELM–MIL and MILIS does (see Table 6), which is attributed to the combination of weight update method and Bayesian classifier training used in BELM–MIL. Over the past few years, deep learning has become a leading trend for feature selection [58], [59]. Extension of the BELM structure into a deeper architecture provides a potentially promising approach to learn more effective features for higher MIL classification accuracy.

To address the aforementioned limitations, we consider that the following three aspects are worthy of our future studies: First, the performance of ELM with sparse Bayesian learning and Full Bayesian method of ELM on multi-instance classification deserve further study [54], [60], [61]. Second, deep learning methods could be combined to reduce the dimension of instance, thus, achieve a better speed while keeping high classification accuracy. Finally, we will extend our work to different real-world problems.

To sum up, we propose a BELM-based algorithm for MIL classification. Under the probabilistic framework, BELM combines the advantages of both ELM and Bayesian learning. The BELM-based method is thus able to improve the classification model through regularization of automatic estimation for reducing possible over-fitting in the calibration process. The effectiveness of BELM for MIL classification is validated on three different fields datasets in comparison with several other rivals. Superior classification accuracy verifies that the proposed BELM–MIL is a promising candidate for performance improvement of MIL classification.

ACKNOWLEDGMENT

The authors would like to thank Prof. Zhangbing Zhou from the China University of Geosciences Beijing for his thoughtful suggestions that have helped improve this article.

REFERENCES

[1] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, "Solving the multiple instance problem with axis-parallel rectangles," *Artif. Intell.*, vol. 89, nos. 1–2, pp. 31–71, Jan. 1997.
[2] Z. Zhao, G. Fu, S. Liu, K. M. Elokely, R. J. Doerksen, Y. Chen, and D. E. Wilkins, "Drug activity prediction using multiple-instance learning via joint instance and feature selection," *BMC Bioinf.*, vol. 14, no. S14, p. S16, Oct. 2013.
[3] B. Liu, Y. Xiao, and Z. Hao, "A selective multiple instance transfer learning method for text categorization problems," *Knowl.-Based Syst.*, vol. 141, pp. 178–187, Feb. 2018.
[4] R. Hong, M. Wang, Y. Gao, D. Tao, X. Li, and X. Wu, "Image annotation by multiple-instance learning with discriminative feature mapping and selection," *IEEE Trans. Cybern.*, vol. 44, no. 5, pp. 669–680, May 2014.
[5] F. Huang, J. Qi, H. Lu, L. Zhang, and X. Ruan, "Salient object detection via multiple instance learning," *IEEE Trans. Image Process.*, vol. 26, no. 4, pp. 1911–1922, Apr. 2017.
[6] X. Zhang, S. Qu, J. Huang, B. Fang, and P. Yu, "Stock market prediction via multi-source multiple instance learning," *IEEE Access*, vol. 6, pp. 50720–50728, 2018.
[7] O. Maron and T. Lozano-Pérez, "A framework for multiple-instance learning," in *Proc. Conf. Adv. Neural Inf. Process. Syst.*, 1997, pp. 570–576.
[8] Q. Zhang and S. A. Goldman, "Em-DD: An improved multiple-instance learning technique," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, pp. 1073–1080.
[9] J. Wang and J. D. Zucker, "Solving the multiple-instance problem: A lazy learning approach," in *Proc. 17th Int. Conf. Mach. Learn.*, 2000, pp. 1–7.
[10] S. Andrews, T. Hofmann, and I. Tsochantaridis, "Multiple instance learning with generalized support vector machines," in *Proc. 18th Nat. Conf. Artif. Intell.*, 2002, pp. 943–944.
[11] Z. H. Zhou and M. L. Zhang, "Neural networks for multi-instance learning," in *Proc. Int. Conf. Intell. Inf. Technol.*, 2002, pp. 455–459.
[12] Y. Chen, J. Bi, and J. Z. Wang, "MILES: Multiple-instance learning via embedded instance selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 1931–1947, Dec. 2006.
[13] Z. Fu, A. Robles-Kelly, and J. Zhou, "MILIS: Multiple instance learning with instance selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 958–977, May 2011.
[14] J. Wang, L. Cai, J. Peng, and Y. Jia, "A novel multiple instance learning method based on extreme learning machine," *Comput. Intell. Neurosci.*, vol. 2015, Feb. 2015, Art. no. 405890.
[15] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew *et al.*, "Extreme learning machine: A new learning scheme of feedforward neural networks," *Neural networks*, vol. 2, pp. 985–990, Jul. 2004.
[16] Y. Bazi, N. Alajlan, F. Melgani, H. AlHichri, S. Malek, and R. R. Yager, "Differential evolution extreme learning machine for the classification of hyperspectral images," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 6, pp. 1066–1070, Jun. 2014.
[17] D. D. Wang, R. Wang, and H. Yan, "Fast prediction of protein–protein interaction sites based on extreme learning machines," *Neurocomputing*, vol. 128, pp. 258–266, Mar. 2014.
[18] A. A. Mohammed, R. Minhas, Q. M. Jonathan Wu, and M. A. Sid-Ahmed, "Human face recognition based on multidimensional PCA and extreme learning machine," *Pattern Recognit.*, vol. 44, nos. 10–11, pp. 2588–2597, Oct. 2011.
[19] F. L. Chen and T. Y. Ou, "Sales forecasting system based on gray extreme learning machine with taguchi method in retail industry," *Expert Syst. Appl.*, vol. 38, no. 3, pp. 1336–1345, Mar. 2011.
[20] Z. M. Yaseen, S. O. Sulaiman, R. C. Deo, and K.-W. Chau, "An enhanced extreme learning machine model for river flow forecasting: State-of-the-art, practical applications in water resource engineering area and future research direction," *J. Hydrol.*, vol. 569, pp. 387–408, Feb. 2019.
[21] K. S. Banerjee, "Generalized inverse of matrices and its applications," *Technometrics*, vol. 15, no. 1, pp. 197, Feb. 1973.
[22] Y. Zhang, Y. Wang, G. Zhou, J. Jin, B. Wang, X. Wang, and A. Cichocki, "Multi-kernel extreme learning machine for EEG classification in brain-computer interfaces," *Expert Syst. Appl.*, vol. 96, pp. 302–310, Apr. 2018.

[23] E. Soria-Olivas, J. Gomez-Sanchis, J. D. Martin, J. Vila-Frances, M. Martinez, J. R. Magdalena, and A. J. Serrano, "BELM: Bayesian extreme learning machine," *IEEE Trans. Neural Netw.*, vol. 22, no. 3, pp. 505–509, Mar. 2011.

[24] Y. Zhang, J. Jin, X. Wang, and Y. Wang, "Motor imagery EEG classification via Bayesian extreme learning machine," in *Proc. 6th Int. Conf. Inf. Sci. Technol. (ICIST)*, May 2016, pp. 27–30.

[25] W. Wu, C. Wu, S. Gao, B. Liu, Y. Li, and X. Gao, "Bayesian estimation of ERP components from multicondition and multichannel EEG," *NeuroImage*, vol. 88, pp. 319–339, Mar. 2014.

[26] R. Zhang, G.-B. Huang, N. Sundararajan, and P. Saratchandran, "Multicategory classification using an extreme learning machine for microarray gene expression cancer diagnosis," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 4, no. 3, pp. 485–495, Jul. 2007.

[27] Z. Hao, Z. Xu, H. Zhao, and H. Fujita, "A dynamic weight determination approach based on the intuitionistic fuzzy Bayesian network and its application to emergency decision making," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 4, pp. 1893–1907, Aug. 2018.

[28] Y. Yang and M. Ding, "Decision function with probability feature weighting based on Bayesian network for multi-label classification," *Neural Comput. Appl.*, vol. 31, no. 9, pp. 4819–4828, Sep. 2019.

[29] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Mach. Learn.*, vol. 29, nos. 2–3, pp. 131–163, 1997.

[30] K. P. Murphy, "Naive Bayes classifiers," *Univ. Brit. Columbia*, vol. 18, p. 60, Oct. 2006.

[31] D. P. Mandic, "A generalized normalized gradient descent algorithm," *IEEE Signal Process. Lett.*, vol. 11, no. 2, pp. 115–118, Feb. 2004.

[32] D. J. MacKay, "Probable networks and plausible predictions—A review of practical Bayesian methods for supervised neural networks," *Netw., Comput. neural Syst.*, vol. 6, no. 3, pp. 469–505, 1995.

[33] H. K. Kwan, "Simple sigmoid-like activation function suitable for digital hardware implementation," *Electron. Lett.*, vol. 28, no. 15, pp. 1379–1380, 1992.

[34] J. Wu, S. Pan, X. Zhu, C. Zhang, and X. Wu, "Multi-instance learning with discriminative bag mapping," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 6, pp. 1065–1080, Jun. 2018.

[35] J. Wu, X. Zhu, C. Zhang, and Z. Cai, "Multi-instance learning from positive and unlabeled bags," in *Proc. Pacific–Asia Conf. Knowl. Discovery Data Mining*. Cham, Switzerland: Springer, 2014, pp. 237–248.

[36] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 577–584.

[37] X.-S. Wei, J. Wu, and Z.-H. Zhou, "Scalable algorithms for multi-instance learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 4, pp. 975–987, Apr. 2017.

[38] D. Michie, S. Muggleton, D. Page, and A. Srinivasan. (1994). *To the International Computing Community: A New East-West Challenge*. [Online]. Available: http://www.doc.ic.ac.uk/~shm/Papers/ml-chall.pdf

[39] L. Dong, "A comparison of multi-instance learning algorithms," Ph.D. dissertation, Dept. Comput. Sci., Univ. Waikato, Hamilton, New Zealand, 2006.

[40] Y. Li, S. Wang, Q. Tian, and X. Ding, "A boosting approach to exploit instance correlations for multi-instance classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 12, pp. 2740–2747, Dec. 2016.

[41] A. Srinivasan, S. Muggleton, R. D. King, and M. J. Sternberg, "Mutagenesis: Ilp experiments in a non-determinate biological domain," in *Proc. 4th Int. workshop Inductive Log. Program.*, vol. 237, 1994, pp. 217–232.

[42] P. Reutemann and D. Adams, "Development of a propositionalization toolbox," M.S. thesis, Albert Ludwigs Univ. Freiburg, Freiburg im Breisgau, Germany, 2004, vol. 8849, no. 24, p. 12678.

[43] P. Braddock, D.-E. Hu, T. Fan, I. Stratford, A. Harris, and R. Bicknell, "A structure-activity analysis of antagonism of the growth factor and angiogenic activity of basic fibroblast growth factor by suramin and related polyanions," *Brit. J. Cancer*, vol. 69, no. 5, pp. 890–898, May 1994.

[44] S.-J. Huang, W. Gao, and Z.-H. Zhou, "Fast multi-instance multi-label learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2614–2627, Nov. 2019.

[45] R. Moazenzadeh, B. Mohammadi, S. Shamshirband, and K.-W. Chau, "Coupling a firefly algorithm with support vector regression to predict evaporation in northern Iran," *Eng. Appl. Comput. Fluid Mech.*, vol. 12, no. 1, pp. 584–597, Jan. 2018.

[46] C.-T. Cheng, J.-Y. Lin, Y.-G. Sun, and K. Chau, "Long-term prediction of discharges in manwan hydropower using adaptive-network-based fuzzy inference systems models," in *Proc. Int. Conf. Natural Comput.* Berlin, Germany: Springer, 2005, pp. 1152–1161.

[47] F. Fotovatikhah, M. Herrera, S. Shamshirband, K.-W. Chau, S. F. Ardabili, and M. J. Piran, "Survey of computational intelligence as basis to big flood management: Challenges, research directions and future work," *Eng. Appl. Comput. Fluid Mech.*, vol. 12, no. 1, pp. 411–437, Jan. 2018.

[48] W.-C. Wang, K.-W. Chau, L. Qiu, and Y.-B. Chen, "Improving forecasting accuracy of medium and long-term runoff using artificial neural network based on EEMD decomposition," *Environ. Res.*, vol. 139, pp. 46–54, May 2015.

[49] M. A. Tahir, J. Kittler, and A. Bouridane, "Multi-label classification using stacked spectral kernel discriminant analysis," *Neurocomputing*, vol. 171, pp. 127–137, Jan. 2016.

[50] K. Das, S. Conjeti, A. G. Roy, J. Chatterjee, and D. Sheet, "Multiple instance learning of deep convolutional neural networks for breast histopathology whole slide classification," in *Proc. IEEE 15th Int. Symp. Biomed. Imag. (ISBI)*, Apr. 2018, pp. 578–581.

[51] F. Herrera, S. Ventura, R. Bello, C. Cornelis, A. Zafra, D. S. Tarrago, and S. Vluymans, *Multiple Instance Learning. Foundations and Algorithms*. Cham, Switzerland: Springer, 2016, p. 10.

[52] M. Jia, Z. Gao, Q. Guo, and X. Gu, "WITHDRAWN: Fast compressive tracking with robust example selection based on multiple instance learning in smart and autonomous systems," *Pattern Recognit.*, vol. 69, pp. 336–351, Sep. 2017.

[53] Y. Zhang, J. Wu, C. Zhou, P. Zhang, and Z. Cai, "Multiple-instance learning with evolutionary instance selection," in *Proc. Int. Conf. Database Syst. Adv. Appl.* Cham, Switzerland: Springer, 2016, pp. 229–241.

[54] Z. Jin, G. Zhou, D. Gao, and Y. Zhang, "Eeg classification using sparse Bayesian extreme learning machine for brain–computer interface," *Neural Comput. Appl.*, vol. 2, pp. 1–9, Oct. 2018.

[55] J. Luo, C.-M. Vong, and P.-K. Wong, "Sparse Bayesian extreme learning machine for multi-classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 4, pp. 836–843, Apr. 2014.

[56] Y. Zhang, G. Zhou, J. Jin, Q. Zhao, X. Wang, and A. Cichocki, "Sparse Bayesian classification of EEG for brain–computer interface," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2256–2267, Nov. 2016.

[57] Y. Chen, J. Yang, C. Wang, and D. Park, "Variational Bayesian extreme learning machine," *Neural Comput. Appl.*, vol. 27, no. 1, pp. 185–196, Jan. 2016.

[58] Q. Zou, L. Ni, T. Zhang, and Q. Wang, "Deep learning based feature selection for remote sensing scene classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 11, pp. 2321–2325, Nov. 2015.

[59] V. B. Semwal, K. Mondal, and G. C. Nandi, "Robust and accurate feature selection for humanoid push recovery and classification: Deep learning approach," *Neural Comput. Appl.*, vol. 28, no. 3, pp. 565–574, 2017.

[60] K. I. Wong, C. M. Vong, P. K. Wong, and J. Luo, "Sparse Bayesian extreme learning machine and its application to biofuel engine performance prediction," *Neurocomputing*, vol. 149, pp. 397–404, Feb. 2015.

[61] Y. Chen, J. Yang, C. Wang, and N. Liu, "Multimodal biometrics recognition based on local fusion visual features and variational Bayesian extreme learning machine," *Expert Syst. Appl.*, vol. 64, pp. 93–103, Dec. 2016.

PEIPEI WANG received the B.S. degree in software engineering from South Central University for Nationalities, Wuhan, China, in 2015, and the M.S. degree in computer science and technology from the China University of Geosciences, Wuhan, in 2018. She is currently pursuing the Ph.D. degree in surveying and mapping with the China University of Geosciences, Beijing.

Her research interests include machine learning, data mining, and spatial-temporal analysis and modeling.

XINQI ZHENG received the B.S. and M.S. degrees from Henan University, Kaifeng, China, in 1985 and 1988, respectively, and the Ph.D. degree from Information Engineering University, Zhengzhou, China, in 2004.

He is currently a Professor with the School of Information Engineering, China University of Geosciences, Beijing. His research interests include spatial analysis and modeling, spatial data mining, spatial optimization, spatial-temporal complex system simulation, and modeling.

JUNHUA KU received the M.S. degree from the Department of Mathematics, Hainan Normal University, in 2007, and the Ph.D. degree in computer science and technology from the School of Computer Science, China University of Geosciences, in 2015. He is currently working with the School of Mathematics, Yibin University, Yibin, Sichuan, China. His research interests include artificial intelligence, machine learning, and computer networks and applications.

CHUNNING WANG received the B.S. degree in computer and applications from Beihang University, Beijing, China, in 1995, and the M.S. degree in library science from Peking University, Beijing, in 2010. She worked at the National Geological Data Library, engaged in the digitization and modern management of geological data from 1995 to 2010. She is currently a Senior Engineer with the Department of Information Technology, National Geological Library of China. Her research interests include geological informatization and information management.