Applied Intelligence (2021) 51:7233–7244
https://doi.org/10.1007/s10489-021-02257-x

Dynamic feature selection algorithm based on Q-learning mechanism

Ruohao Xu [1,2] · Mengmeng Li [1,2] · Zhongliang Yang [1,2] · Lifang Yang [1,2] · Kangjia Qiao [1] · Zhigang Shang [1,2]

Accepted: 3 February 2021 / Published online: 1 March 2021

Abstract
Feature selection is a technique to improve the classification accuracy of classifiers and a convenient data visualization method. As an incremental, task oriented, and model-free learning algorithm, Q-learning is suitable for feature selection, this study proposes a dynamic feature selection algorithm, which combines feature selection and Q-learning into a framework. First, the Q-learning is used to construct the discriminant functions for each class of the data. Next, the feature ranking is achieved according to the all discrimination functions vectors for each class of the data comprehensively, and the feature ranking is doing during the process of updating discriminant function vectors. Finally, experiments are designed to compare the performance of the proposed algorithm with four feature selection algorithms, the experimental results on the benchmark data set verify the effectiveness of the proposed algorithm, the classification performance of the proposed algorithm is better than the other feature selection algorithms, meanwhile the proposed algorithm also has good performance in removing the redundant features, and the experiments of the effect of learning rates on the our algorithm demonstrate that the selection of parameters in our algorithm is very simple.

Keywords Feature selection · Q-learning · Dynamic · Markov decision process

1 Introduction

The data sets in many applications such as computer vision, data mining, and pattern recognition always have thousands of variables or features. High dimensional data significantly increases the time and space requirements for data processing. In addition, the irrelevant and redundant features may also give rise to a series of problems including overfitting and poor prediction performance, resulting in the inefficiency of the learning tasks [1–5]. Therefore, dimensionality reduction has become an important stage of data preprocessing in such applications [6, 7].

As the dimension of information increases, the prediction performance, learning efficiency, data visualization, and information intelligibility decrease due to the presence of irrelevant, redundant, and noisy features. There are usually two kinds of approaches to reduce dimensions: feature selection and feature extraction [8]. The former selects the feature subset from the original set, while the later transforms the original feature into a new feature space. Contrary to feature extraction, feature selection does not change the original representation of the variables. So feature selection can retain the original semantics of the variables, thereby providing better interpretability. Another advantage of feature selection is that it only needs to collect or calculate the selected features, while in the feature extraction method, all original features are needed to obtain a low-dimensional representation. Therefore, in the past few years, many studies have focused on feature selection approach to solve the dimensionality reduction problem.

Our work focuses on introducing Reinforcement Learning (RL) mechanisms to feature selection. RL [9, 10] and supervised learning are two independent learning paradigms in machine learning. RL is a learning method driven by reward signals. In recent years, with the combination of the deep network [11] and RL, deep reinforcement learning (DRL) has achieved excellent results in many aspects, such as computer games [12], robot control [13], etc. Feature selection with RL has been previously addressed [14], and the difference of our approach lies in its sequential decision process and feature ranking process, and [15] extend the work done in [14] by introducing RL-

✉ Zhigang Shang
zhigang_shang@zzu.edu.cn

[1] School of Electrical Engineering, Zhengzhou University, Zhengzhou 450001, China

[2] Henan Key Laboratory of Brain Science and Brain-Computer Interface Technology, Zhengzhou University, Zhengzhou 450001, China

based frameworks to conduct an optimal search in the state space, but the reward function depends on the classifier. In [16] reinforcement learning is developed to more systematically handle the sequential feature selection problem at the instance level, formulate a partially observable Markov Decision Process (POMDP) to treat each instance with missing entries as the partial observation of state, however, they assume pre-learned classifier is given, and the classifier will influence the performance of their algorithm. In 2014, Ba et al. [17] used the RL to attend to the most relevant feature of the input image, and J. Feng et al. [18] used RL to select high-quality sentences from noisy text to improve the classification accuracy of classifiers for noisy data in 2018. However, they trained both classifiers and RL agents sharing the input embeddings, and may have higher computing costs.

Inspired by online feature selection [19–21] and RL, we integrate the learning mechanism of Q-learning into feature selection and propose the dynamic feature selection algorithm based on Q-Learning mechanism (QLFS). In our work samples are used to constructed Q function to choose more valuable actions and get more rewards, in which features are ranked according to the parameters of the Q function. Specifically, we consider the problem of FS as an RL problem, in which any sample is considered as a state and an action is to predict the label of sample. We use the linear function of features to construct Q function, helping to reflect the importance of each feature.

In our RL framework, the reward function does not depend on extra selected classifiers. That is to say, there is no classifier in our algorithm, so selecting and training classifiers are avoided, making the structure of our algorithm simpler. The states and actions are encoded separately to construct a Q function for each class of the data, increasing the flexibility.

The main contributions of this paper are as follows:1) We formulate the feature selection problem as a sequential decision-making process and combine feature selection and Q-learning into a framework. 2) Our Q-learning feature selection framework mainly includes the definition of interaction rules between agent and environment, and the design of reward function. 3) We encode states and actions separately to construct a Q-function for each class of the data, and learn the characteristics of each class of the data separately, in which a dynamic update method is used to update the parameters, increasing the flexibility of the algorithm framework. 4) We evaluate the performance of QLFS systematically on various types of data sets.

The rest of this paper is organized as follows. In Section 2, the feature selection, the Markov decision process and the Q-learning algorithm are introduced. QLFS is described in Section 3. In Section 4, the experiments and results are reported. Finally, we provide some conclusion and discussion in Section 5.

2 Preliminary knowledge

2.1 Feature selection

Feature selection can be applied to both supervised [22], unsupervised [23–25] and semi-supervised learning [26]. This study focuses on supervised learning problems. The family of supervised feature selection algorithms can be divided into filters [27–29], wrappers [30, 31] and embedding methods [32–34] according to whether the classifier is integrated into the algorithm [35]. For the filtering method, there is no classifier in the algorithm. For wrapper methods, the feature subset search algorithm is wrapped in the classification model, and the feature subset is scored according to its prediction ability. In addition, the core idea of the embedding method is to search for the best feature subset in the process of constructing the classifier. Compared with the filtering method, the wrapper method and embedded method are closely coupled with specific classifiers, so they have better performance, but also higher computing costs. Therefor in this study, we will focus on the filtering method of feature selection.

2.2 Markov decision process

For RL, the agent continuously interacts with the environment, and learns through the reward given by the environment, that is, “the trial and error”. In the learning process, the agent “exploits” and “explores”, and adjusts the evaluation value of each action based on the feedback of the environment, learning the optimal strategy that can obtain the maximum reword finally. The main difference between RL and supervised learning is that the agent gets a reward signal through environmental feedback, which indicates the quality of the action, rather than to be told the correct answer. The trial and error learning mode of RL is more in line with the biological learning mechanism.

Markov decision process is a mathematical model of RL, which can be represented by a tuple (s, a, p, r, γ), where s is the set of states in the decision process, a is the set of actions, p is the transition probability, r is the reward value, and γ is the discount factor. The characteristic of this model is that the reward and the next state are only related to the current state and action, rather than to the earlier state and action.

A classification problem can be described as a Markov decision process. The state s can be defined as a sample, and a can be defined as the action to predict which class the sample

belongs to. At each step, the environment will give the agent a sample, and the agent chooses an action. Then the environment feeds back the reward to the agent based on the action selected by the agent, as shown in formula (1):

$$r_t = \begin{cases} r & if\ a_t = l_t \\ -r & \text{else} \end{cases} \tag{1}$$

where l_t is the label of the *t-th* sample.

The goal of the agent is finding the optimal strategy to obtain more rewards. The overall process of the Markov decision process for classification problem is shown in Fig. 1.

State *S* Representative state space. The state of the environment is determined by the training samples. At the beginning of training, the agent receives the first sample X_1 as its initial state s_1, and state s_j at the time j corresponds to the *j-th* sample X_j.

Action *A* Action is associated with the label of the sample. The action a_t taken by the agent is to predict a class label.

Transition probability *P* The probability of the current state transitioning to another state after acting. In this paper, the transition probability p is deterministic. The next state of the agent is determined according to the order of the data set.

Policy*π* The probability distribution function represents the probability of taking various actions at state s_t.

Reward *R* The immediate reward given by the environment, which can measure the success or failure of an agent's actions. Agents find the best policy driven by the reward.

Return *G* The cumulative reward represents a long-term benefit, $G_t = \sum_{k=0}^{n} \gamma^k R_{k+1}$, where γ is discount factor. The purpose of RL is to maximize returns.

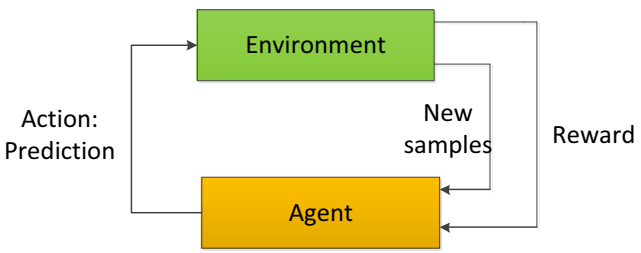

Fig. 1 Markov decision process for the classification problem

2.3 Q-learning

There is a Q function in Q-learning, which represents the expected reward of the agent performing the action a_t at the state s_t:

$$Q(s_t, a_t) = E_\pi(G_t | s_t, a_t) \tag{2}$$

According to the Bellman equation, the Q function can be expressed as:

$$Q(s_t, a_t) = E_\pi(r_t + \gamma Q(s_{t+1}, a_{t+1}) | s_t, a_t) \tag{3}$$

We can obtain the optimal strategy π^* by solving the optimal Q function. If the optimal Q function is known, the optimal strategy can be obtained through the greedy strategy. The strategy π^* is updated as follows:

$$\pi(a|s) = \begin{cases} 1 & if\ a = \text{argmax}_a Q^*(s, a) \\ 0 & else \end{cases} \tag{4}$$

So we can get the updated formula of Q-learning:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha * (r_{t+1} + \alpha * \max Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)) \tag{5}$$

where α is the learning rate, and $r_{t+1} + \alpha * \max Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)$ is the Temporal-Difference Learning(TD) error, $r_{t+1} + \alpha * \max Q_t(s_{t+1}, a_{t+1})$ is the target estimate of the Q value. The update direction of the Q function is to make $Q_t(s_t, a_t)$ approximate the target estimate of Q.

3 Dynamic feature selection algorithm based on Q-learning mechanism

3.1 Q-learning based discriminant function construction

In this study, we use the Q-learning algorithm to build a discriminant function for each class of data. As mentioned, the action is to predict the sample label, and the agent selects the action based on the Q function in Markov decision process. Here, we try to generate a discriminant function for each class of data as a Q function, in which the value of the discriminant function of class A for the data labeled A is positive, while the value for the data labeled B is negative.

For the low-dimensional finite state space, Q function are recorded by a table. The Q values of all state-action combinations are listed in Q-table, through which the optimal strategy

will be obtained. However, there are many state-action combinations, and a large Q-table needs to be established for classification problems. For the above kind of problem, we usually use a function to approximate the Q-table, that is, the function approximation method.

The traditional function approximation method uses the joint vector of state and action to represent the state-action combination, $(s_t, a_t) = [s_t, a_t]$. The Q function is jointly represented by the joint vector $[s_t, a_t]$ and the vector w, where the vector w has many forms, such as polynomial, neural network, etc. In this paper, in order to increase the flexibility of decision space, instead of expressing state-action combinations in above way, we encode the state and actions separately, and the different polynomial functions vector are utilized to represent different actions to get multiple decision functions finally. For example, the action a_i indicates that the sample is predicted to be class i, and action a_j indicates that the sample is predicted to be class j. The Q function corresponding to action a_i is polynomial (6), and the Q function corresponding to action a_j is polynomial (7):

$$Q(s, a_i) = w_i{}^T * \varphi(X) \tag{6}$$

$$Q(s, a_j) = w_j{}^T * \varphi(X) \tag{7}$$

where $\varphi(X)$ is the expression of state s corresponding to a sample $X = [x_1, x_2 ... x_m]^T$, and $\varphi(X) = [1, x_1, x_2 ... x_m]^T$.

The agent adopts a greedy strategy to select actions, shown as formula (8):

$$a = \begin{cases} \text{argmax}_a Q(s, a) & 1-\varepsilon \\ \text{selecting actions randomly} & \varepsilon \end{cases} \tag{8}$$

If the action a_i is selected, the parameter w_i will be updated according to the feedback of the environment. Similarly, if the action a_j is selected, the parameter w_j will be updated according to the feedback of the environment. Finally, for the sample labeled i, $w_i{}^T * \varphi(X_t)$ tends to be positive and $w_j{}^T * \varphi(X_t)$ tends to be negative. In contrast, for the sample labeled j, $w_i{}^T * \varphi(X_t)$ tends to be negative and $w_j{}^T * \varphi(X_t)$ tends to be positive. Therefore, the value of the discriminant function corresponding to the parameter w_i for the data labeled i is positive, the value of the discriminant function corresponding to the parameter w_j for the data labeled i is negative. So for the samples labeled i, we choose the action a_i. The flow chart to construct the discriminant functions based on the Q-learning algorithm is shown in Fig. 2.

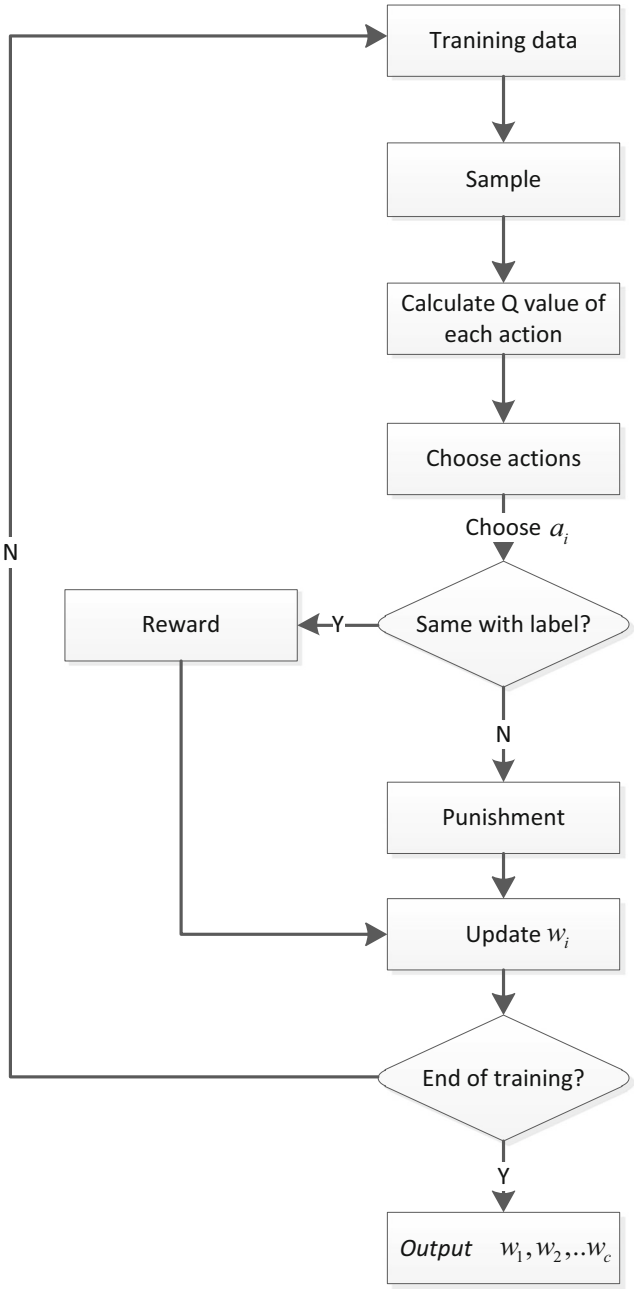

Fig. 2 The flow chart of constructing the discriminant functions based on the Q-learning algorithm

3.2 Feature ranking

$$y_t = \begin{cases} r_t & if \quad terminal = ture \\ r_t + \gamma \max Q_n(s_{t+1}, a_{t+1}) & else \end{cases} \tag{9}$$

For a certain sample, the agent acts a_i, and the environment feedbacks the reward signal r_t according to whether the action corresponds to the sample label. Then the target estimate value y can be calculated through r_t:

In this paper, the discount factor γ is set to 0.1.

According to formula (6), the loss function can be obtained as shown in formula 10:

$$L(w_i) = \left(y_t - {w_i}^T \varphi(X_t)\right)^2 \tag{10}$$

We use the stochastic semi-gradient descent method with fast convergence and simple calculation to solve w_i, in which we only differentiating ${w_i}^T * \varphi(X_t)$ as shown in formula 11:

$$\frac{\partial L(w_i)}{\partial w_i} = -2*\left(y_t - {w_i}^T \varphi(X_t)\right)\varphi(X_t) \tag{11}$$

So we get the updated formula of discriminant function vector w_i as shown in formula 12:

$$w_i(t)^T = w_i(t-1)^T + \alpha\left(y_t - w_i(t-1)^T \varphi(X_t)\right)\varphi(X_t) \tag{12}$$

Suppose we are given n training samples $\{X_i, l_i\}$, $i = 1, \cdots, n$, and $X_i \in R^m$ is the *i-th* data point with the label represented as $l_i \in \{1, 2, 3, ..., c\}$. Our goal is to select d features from the original m features for classification.

Here, a direct method is to find a 0–1 selection matrix $W(m * d)$, where each column in W has only one component equal to 1. Then we can calculate $x' = W^T * x$ and x' is the new d-dimensional sub-vector of x. However, directly finding such a 0–1 selection matrix is proven to be an NP-hard problem [21]. As mentioned, we use the Q-learning algorithm to build a discriminant function for each class of data, each element of the discriminant function vectors can represent the contribution of each feature to the function construction well, reflecting the importance of each feature. We use the function vectors obtained by the Q-learning algorithm to replace the selection matrix for feature selection in this study.

First, the Q-learning algorithm will generate more stable discriminant function vectors for each class of data after several iterations, and combine these discriminant function vectors into a matrix W'. The Q-values of each action is calculated as given in formula 13:

$$W' = \begin{bmatrix} b_1, b_2, ..., b_c \\ w_{11}, w_{21}..., w_{c1} \\ w_{12}, w_{22}..., w_{c2} \\ ... \\ w_{1m}, w_{2m}..., w_{cm} \end{bmatrix} \begin{bmatrix} b_1, w_{11}, w_{12}...w_{1m} \\ b_2, w_{21}, w_{22}...w_{2m} \\ ... \\ b_c, w_{c1}, w_{c2}...w_{cm} \end{bmatrix} * \begin{bmatrix} 1 \\ x_1 \\ ... \\ x_m \end{bmatrix} = \begin{bmatrix} Q_1 \\ Q_2 \\ ... \\ Q_c \end{bmatrix} \tag{13}$$

Next, the elements in W' except the first row are extracted as a matrix W for feature selection. The first row in W' corresponds to the constants, which is only used to better construct the discriminant function. The matrix W is given in formulas 14 and 15:

$$W = \begin{bmatrix} w_{11}, w_{21}, ..., w_{c1} \\ w_{12}, w_{22}, ..., w_{c2} \\ ... \\ w_{1m}, w_{2m}, ..., w_{cm} \end{bmatrix} \tag{14}$$

$$\text{Set } \overline{W} = \left[\overline{W}_1, \overline{W}_2 ... \overline{W}_m\right] \tag{15}$$

where $\overline{W}_i$ is the $\ell 2$-norm of the *i-th* row vector in W, $\overline{W}_i = \sqrt{{w_{1i}}^2 + {w_{2i}}^2 ... + {w_{ci}}^2}$. We can rank the features according to the element in $\overline{W}$.

As mentioned, Q-learning is an online algorithm and updates the parameters each time when a sample is entered iteratively. Therefore, we propose a dynamic feature selection algorithm based on Q-Learning mechanism in this paper, which is described as follows. First, we update $\overline{W}$ through iteration until it is stable, then the features are ranked according to the corresponding element of the stable $\overline{W}$; Next, the worst part of features are removed according to the criterion we set, and a new sub data set is obtained, then the $\overline{W}$ of the new sub set will be updated iteratively; The iteration updating process will be repeated until stop criterion is reached finally.

Detailed steps are as follows:

(1). The discriminant function vector w_i, the transformation matrix W and $\overline{W}$ are obtained according to the formulas (13), (14) and (15); $\overline{w}$ is the average of the elements in $\overline{W}$;

(2). The features are ranked according to the corresponding element of the $\overline{W}$, the ranking of the features which corresponding elements in $\overline{W}$ are smaller than $\overline{w}$ will be recorded as their final ranking, then the values of these features in the original data are replaced with 0, and a new sub data set is obtained.

(3). Update $\overline{W}$ and $\overline{w}$ according to the step (1), repeat the step (2).

(4). Repeat the step (3) until the non-zero items in $\overline{W}$ are greater than $\overline{w}$, then combine the feature ranking of non-zero items in $\overline{W}$ with the previously saved feature ranking to form the final feature ranking.

Algorithm 1 Dynamic feature selection algorithm based on Q-Learning mechanism

Input: training set D, number of training set T, learning rate α ,maximum number of iterations K, discount factor γ .

output: feature ranking

1:Initialize w, W, $\overline{W}$, $\overline{w}$,feature rank P

2: **while** there are non-zero elements in $\overline{W}$ less than $\overline{w}$ **do**

3: $w0=w$, then Initialize w

4: **For** k=1 to K do

5: **For** t=1 to T do

6: Calculate the Q value of all actions in the state s_t : $w_i^T * \varphi(x)$

7: Choose actions based on ε-*greed* strategy, and feedback rewards r_t and s_{t+1} .

8: $r_t = \begin{cases} 1 & if\ a_t = l_t \\ -1 & else \end{cases}$

9: Calculate target estimate y: $y = \begin{cases} r_t & if\ s_{t+1} = end \\ r_t + \gamma \max Q_n(s_{t+1}, a_{t+1}) & \text{else} \end{cases}$

10: Update parameters $w_i : w_i(t+1) = w_i(t) + a(y_t - w_i(t)^T * \varphi(x_t)) * \varphi(x_t)$

11: **end for**

12: **if** $|w - w0| < 0.00001$

13: **Break**

14: **end if**

15: **end for**

16: According to the new parameter vector w_i , calculate W, $\overline{W}$ and $\overline{w}$.

17: Ranking the features according to $\overline{W}$, the ranking of the features corresponding to the elements less than $\overline{w}$ in $\overline{W}$ are recorded and stored to P, and the corresponding elements in D are replaced with 0.

18: **End while**

19:combine the feature ranking of non-zero items in $\overline{W}$ with P to form the final feature ranking

4 Experiments

4.1 Data set description

In our experiment, we collected 12 different benchmark open data sets to illustrate the performance of different feature selection methods. All datasets are from UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/index.html). The brief introduction of the data sets is shown in Table 1.

Table 1 Dataset

Dataset	Dimension	Sample	Class
Segment	18	2310	7
vehicle	18	846	4
Waveform-1	21	5000	3
Satellite	36	6435	6
Optdigits	64	5620	10
Musk-2	166	6589	2
Dna	180	2000	3
Minst	784	10,000	10
Lung	3312	204	5
Gisette	5000	7000	2
Prostate_GE	5966	102	2
Arcene	10,000	200	2

4.2 Classification performance comparison

We compared QLFS with four feature selection algorithms, including Information Gain(IG) [36], Fisher Score(FS) [29], Discriminative Feature Selection(DFS) [22], Infinite Latent

Feature Selection(ILFS) [37]. DFS imposes row sparsity on the transformation matrix of Linear Discriminant Analysis (LDA) through ℓ2,1-norm regularization for feature selection, and ILFS performs the ranking step while considering all the possible subsets of features, as paths on a graph, by passing the combinatorial problem analytically. To compare all kinds of algorithms fairly, according their paper, in ILFS the number of latent variables (Z = 2) and the number of tokens used (T = 6), in DFS $p = 1$ and γ is chosen from {0.01,0.1,1,10,100} in each experiment, 5-fold cross-validation is used, partition one dataset into 5 folds, randomly take 4 folds to perform feature selection, and train some classifiers, remained 1 fold to test.

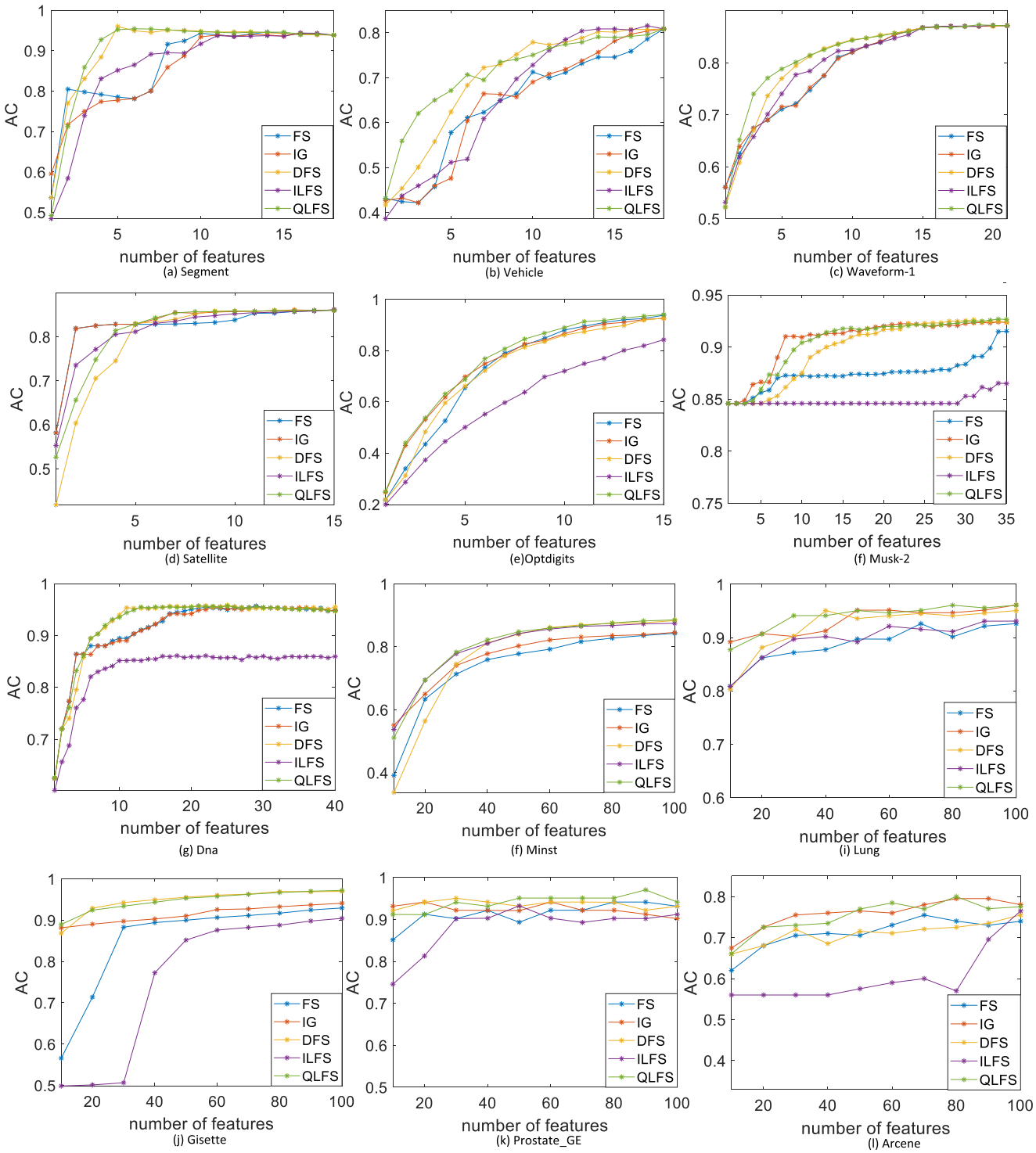

Fig. 3 Comparison between QLFS and other feature selection methods w.r.t classification accuracy

Table 2 Classification average accuracy and standard deviation of different algorithms

Dataset	QLFS	ILFS	DFS	IG	FS
Segment	0.8589±0.0108	0.7394±0.1249	0.8307±0.0239	0.7498±0.0394	0.7978±0.0197
Vehicle	0.6207±0.0573	0.4598±0.0324	0.5012±0.0566	0.4220±0.0260	0.4220±0.0260
Waveform-1	0.7882±0.0206	0.7404±0.0270	0.7698±0.0163	0.7154±0.0101	0.7106±0.0101
Satellite	0.8542±0.0096	0.8351±0.0262	0.8399±0.0108	0.8553±0.0136	0.8291±0.0110
Optdigits	0.9180±0.0057	0.7698±0.0792	0.8879±0.0202	0.9038±0.0134	0.9096±0.0128
Musk-2	0.9257±0.0047	0.8592±0.0123	0.9244±0.0033	0.9244±0.0061	0.8989±0.0210
Dna	0.9520±0.0132	0.8595±0.0088	0.9540±0.0130	0.9540±0.0176	0.9505±0.0170
Minst	0.8757±0.0128	0.8675±0.0106	0.8741±0.0063	0.8352±0.0107	0.8274±0.0052
Lung	0.9607±0.0285	0.9113±0.0379	0.9408±0.0288	0.9467±0.0460	0.9015±0.0752
Gisette	0.9669±0.0029	0.8879±0.0073	0.9694±0.0012	0.9326±0.0048	0.9170±0.0069
Prostate_GE	0.9510±0.0488	0.9024±0.0765	0.9410±0.0645	0.9219±0.0877	0.9414±0.0628
Arcene	0.8001±0.0632	0.5700±0.1214	0.7252±0.0376	0.7949±0.0276	0.7402±0.0358

The support vector machine (SVM) with linear kernel and parameter C = 1 was used as classifier. Figure 3 shows the classification accuracy curves of four feature selection algorithms on datasets.

As shown in Fig. 3, as the number of selected features increases, the trend of classification accuracy of different feature selection methods on different datasets varies. For most datasets, all feature selection approaches achieved higher classification accuracy with more selected features. However, for the *Arcene* dataset, we can see that selecting more features does not achieve higher classification accuracy. Compared with the algorithms FS, IG and ILFS, it can be seen from Fig. 3 that our algorithm achieves a higher average classification accuracy on almost all data sets. Especially on the *Segment, Vehicle, Dna, Gisette* data sets, the performance of our algorithm has been improved by about 10%. And on the *Waveform-1, Optdigits,* and *Minst* datasets, the performance of our algorithm has improved by about 5%. Only in a few cases, the performance of our algorithm is similar to those of FS, IG and ILFS. Moreover, compared with the DFS, our algorithm has better performance in some datasets. For example, in *Vehicle, Satellite, Optdigit, Musk-2, Arcene* our algorithm is superior to DFS. In the other data sets, they offer comparable performance.

In terms of the classification accuracy, our algorithm outperforms the other three methods on most data sets. For clarity, Table 2 listed the classification accuracy and standard deviation when selecting the top 20% of features (where the top 80 features are selected for data larger than 500 dimensions). The results in Table 2 show that the algorithm proposed in this paper achieves the highest performance on most datasets.

Table 3 listed the rank-sum test results for the average accuracy of classification between our algorithm and other algorithms. Here, the results were represented by "1" and "0", "-1", "1" means that the classification (mean) accuracy obtained by QLFS is significantly higher than those of the other methods. "0" means that there is no significant difference between them. "-1" means that the performance of QLFS method is significantly worse than those of the other methods. Combined with the results reported in Tables 2 and 3, it can be seen that the classification accuracy results of QLFS on most data are significantly better than those of the FS, IG and ILFS. Compared with DFS, QLFS is slightly better on most data sets. In conclusion, compared with other algorithms, QLFS can select the most discriminative features.

4.3 Comparison of de-redundancy

To compare the ability to remove redundant features of QLFS and the other methods, we extracted the top 20% of features and measured the redundancy of them. In this paper, we used

Table 3 Rank sum test results between different algorithms

Dataset	QLFS vs ILFS	QLFS vs DFS	QLFS vs IG	QLFS vs FS
Segment	1	0	1	1
Vehicle	1	1	1	1
Waveform-1	1	1	1	1
Satellite	0	0	0	1
Optdigits	1	1	1	0
Musk-2	1	0	0	1
Dna	1	0	0	0
Minst	0	0	1	1
Lung	1	0	0	1
Gisette	1	0	1	1
Prostate_GE	0	0	0	0
Arcene	1	1	0	1
+/≈/−	9/3/0	4/8/0	6/6/0	9/3/0

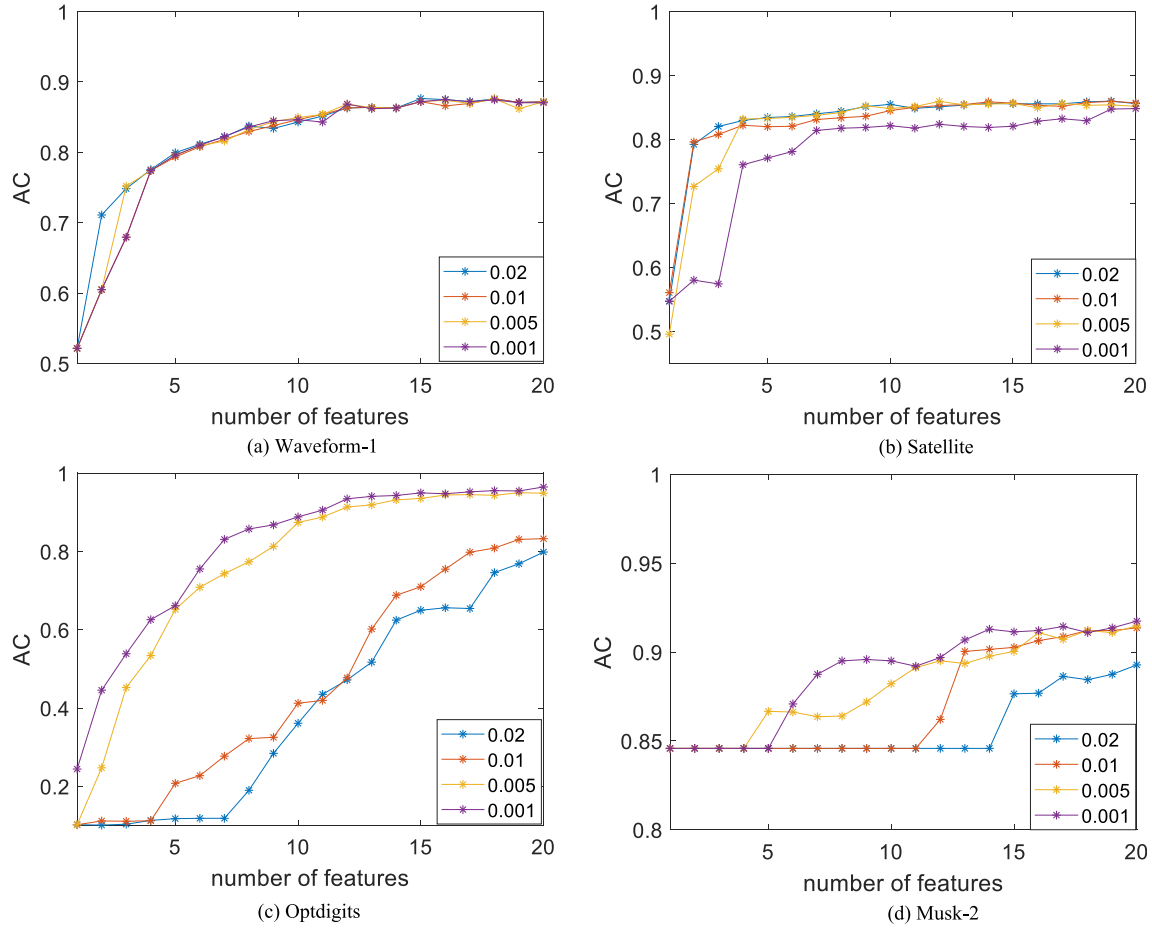

Fig. 4 Comparison of classification accuracy of QLFS with different learning rates

the concept of representing entropy to measure the de-redundancy performance of the algorithm, which is proposed by Devijver et al. [38].

Assuming the eigenvalues of its covariance matrix can be expressed as λ_i (i = 1,2,..d) for the selected d-dimensional feature set, we let $\widetilde{\lambda}_i = \lambda_i / \sum_{i=1}^{d} \lambda_i$, in which $\sum_{i=1}^{d} \widetilde{\lambda}_i = 1$ and $0 < \widetilde{\lambda}_i < 1$. An entropy function can be defined as:

$$H_R = -\sum_{i=1}^{d} \widetilde{\lambda}_i \log \widetilde{\lambda}_i \tag{16}$$

It can be seen from the above formula that when all the eigenvalues are the same, H_R can obtain the maximum value, reflecting that the information between the features is evenly distributed in the feature space. That is to say, the uncertainties involved in feature selection are evenly distributed, and the redundancy between the features is minimized. However, when all the other eigenvalues are zero except for anon-zero one, H_R takes the minimum value of zero, reflecting the features are highly redundant. Table 4 shows the H_R values of the four algorithms on different data sets, the maximum value is bold and the second largest value is marked blue.

It can be seen from Table 4 that QLFS has good de-redundancy performance on most data sets, the H_R of QLFS is the largest or second in most data sets.

4.4 The effect of different gradient learning rates on the QLFS

The gradient learning rate represents the optimization rate of the semi-gradient descent method. The smaller the learning rate, the smaller the effect of a single sample on the parameter update. In this section, we compared the performance of QLFS with different learning rates. Figure 4 shows that the classification performance of QLFS with different learning rates on four datasets: *Waveform-1, Satellite, Optdigits,* and *Musk-2*. The value of the learning rate was set to {0.001, 0.005, 0.01, 0.02}.

For the data set *Waveform-1*, the results of QLFS are different for different learning rates when the number of selected features is less than 5, while the performance is the best when

Table 4 Comparison of H_R of different algorithms

Dataset	QLFS	ILFS	DFS	IG	FS
Segment	**0.9183**	0.4750	0.7520	0.6048	0.6005
Vehicle	**0.6449**	0.6336	0.5890	0.0858	0.0858
Waveform-1	1.1277	0.9202	**1.1492**	0.8185	0.8185
Satellite	1.0700	1.0497	**1.1761**	0.8435	0.5504
Optdigits	**2.2413**	2.2089	2.2145	2.1349	2.1553
Musk-2	**2.5856**	1.7241	2.5685	2.3661	1.836
Dna	3.4807	**3.5234**	3.4779	3.3953	3.3915
Minst	3.4498	3.6306	**3.6878**	3.0911	3.1195
Lung	3.4113	2.5309	**3.5725**	2.1921	1.6491
Gisette	3.5147	**3.9262**	3.5687	2.4780	2.8725
Prostate_GE	2.8510	2.6524	**3.1977**	2.2980	2.3621
Arcene	1.9133	**3.6177**	0.8933	1.6441	1.1850

the learning rate is 0.02. For the *Satellite* data set, the QLFS performance is the best when the learning rate is 0.02, while the performance is very poor when the learning rate is 0.001. On *Optdigits* data set, when the learning rate is 0.001 and 0.005, the performances of QLFS are better than those of 0.02 and 0.01. On the *Musk-2* dataset, QLFS performs best when the learning rate is 0.001. The above results show that as the dimension of the data increases, the smaller learning rate will be more appropriate for QLFS to get better performance.

5 Conclusion and discussion

In this study, we proposed a dynamic feature selection algorithm based on Q-learning mechanism. We formulate the feature selection problem as a sequential decision-making process and combine feature selection and Q-learning into a framework, and states and actions are encoded separately to construct a Q-function for each class of the data, increasing the flexibility of the algorithm framework. First, we use the Q-learning algorithm to build a discriminant function for each class of data. Then the vector $\overline{W}$ is obtained by calculating the 2-norm of the all row vector of the matrix W, then the features are ranked according to the corresponding element of the $\overline{W}$, meanwhile we rank the features during the process of updating W. Finally, we compared our algorithm with the others on different datasets. The results showed that QLFS not only shows stable and high classification accuracy on most datasets compared with other algorithms, but also can remove redundant features better.

Here, we think there are two reasons for the high accuracy of our proposed. First, in our algorithm the same sample is used to update the discriminant function parameters of all classes of data, so the sample information is fully utilized. Second, our algorithm ranks the features while updating the matrix W, which may reduce the interference of the lower ranking features on the ranking of the top features, make us choose the most distinctive feature.

In addition, our algorithm in this paper also has good performance in removing feature redundancy. We think the main reason is that QLFS selects features by using the Q-learning mechanism, the interactions among the whole set of features are considered. In the process of interacting with the samples, QLFS constantly adjusts the weights to make the model more suitable for choosing the correct action, so the interactions among the whole set of features are considered comprehensively. And for vector w_i, the data of class i are transformed into scalar positive number by w_i, while the data of class j are transformed into scalar negative number by w_i, resulting in the data of class i better distinguished from the data of other classes. This just like projecting the data to a new feature space, our algorithm is similar to feature extraction to some degree, so our algorithm can remove the redundant.

In the future we need to solve some problems to extend the proposed dynamic feature selection model. On the one hand, determining a suitable stopping criterion will likely further reduce the operation time and improve the classification performance. On the other hand, the learning rate used in our algorithm is fixed for certain data. In fact, the learning rate should be adjusted according to the progress of parameter updating, so dynamic learning rate will be exploring next step.

And we have used a linear function to build the discriminant function in this study, which does not fit well with complex nonlinear data, so we also hope to introduce the kernel method into our algorithm. Finally, the source code is available at https://github.com/xrh196/qlfs/tree/master to provide the material needed to replicate our experiments.

Acknowledgements The work was supported in part by National Natural Science Foundation of China (61,673,353, and U1304602).

References

1. Liu H, Motoda H (1998) Feature selection for knowledge discovery and data mining. Springer Press, New York
2. Wang L, Zhou N, Chu F (2008) A general wrapper approach to selection of class-dependent features. IEEE Trans Neural Netw 19: 1267–1278
3. Chandrashekar G, Sahin F (2014) A survey on feature selection methods. Comput Electr Eng 40:16–28
4. Cao X, Wei Y, Wen F, Sun J (2014) Face alignment by explicit shape regression. Int J Comput Vis 107:177–190
5. Liu X, Wang L, Zhang J et al (2013) Global and local structure preservation for feature selection. IEEE Trans Neural Netw 25: 1083–1095
6. Hou C, Wang J, Wu Y, Yi D (2009) Local linear transformation embedding. Neurocomputing 72:2368–2378
7. Hou C, Zhang C, Wu Y, Nie F (2010) Multiple view semi-supervised dimensionality reduction. Pattern Recogn 43:720–730
8. Cai J, Luo J, Wang S, Yang S (2018) Feature selection in machine learning: a new perspective. Neurocomputing 300:70–79
9. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction. MIT Press, Cambridge
10. Li Z, Li S, Yue C et al (2019) Differential evolution based on reinforcement learning with fitness ranking for solving multimodal multiobjective problems. Swarm Evol Comput 49:234–244
11. Lecun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521: 436–444
12. Li Y, Fang Y, Akhtar Z (2020) Accelerating deep reinforcement learning model for game strategy. Neurocomputing 408:157–168
13. Won DO, Müller KR, Lee SW (2020) An adaptive deep reinforcement learning framework enables curling robots with human-like performance in real-world conditions. Sci Robot 5:eabb9764
14. Gaudel R, Sebag M (2010) Feature selection as a one-player game. In: Proceedings of 27th international conference on machine learning, Haifa, pp 359–366
15. Fard SMH, Hamzeh A, Hashemi S (2013) Using reinforcement learning to find an optimal set of features. Comput Math Appl 66: 1892–1904
16. Rückstieß T, Osendorfer C, van der Smagt P (2013) Minimizing data consumption with sequential online feature selection. Int J Mach Learn Cybern 4:235–243
17. Ba JL, Mnih V, Kavukcuoglu K (2015) Multiple object recognition with visual attention. In: Proceeding of 3rd international conference on learning representations, San Diego, pp 1–10
18. Feng J, Huang M, Zhao L, et al (2018) Reinforcement learning for relation classification from noisy data. In: Proceedings of 32th AAAI Conference on Artificial Intelligence, New Orleans, pp 5779–5786
19. Wu X, Yu K, Wang H, et al (2010) Online streaming feature selection. In: Proceedings of 27th international conference on machine learning, Haifa, pp 1159–1166
20. Zhou P, Hu X, Li P, Wu X (2017) Online feature selection for high-dimensional class-imbalanced data. Knowl-Based Syst 136:187–199
21. Wang JL, Zhao PL et al (2014) Online feature selection and its applications. IEEE Trans Knowl Data Eng 26:698–710
22. Tao H, Hou C, Nie F, Jiao Y, Yi D (2016) Effective discriminative feature selection with nontrivial solution. IEEE Trans Neural Netw Learn Syst 27:796–808
23. Zhou P, Du L, Li X et al (2020) Unsupervised feature selection with adaptive multiple graph learning. Pattern Recogn 105:107375
24. Shen HT, Zhu Y, Zheng W, et al (2020) Half-quadratic minimization for unsupervised feature selection on incomplete data. IEEE Transactions onNeural Networks and Learning Systems, pp 1–14. https://doi.org/10.1109/TNNLS.2020.3009632
25. Zhang Y, Wang Q, Gong DW, Song XF (2019) Nonnegative Laplacian embedding guided subspace learning for unsupervised feature selection. Pattern Recogn 93:337–352
26. Liu K, Yang X, Yu H, Mi J, Wang P, Chen X (2019) Rough set based semi-supervised feature selection via ensemble selector. Knowl-Based Syst 165:282–296
27. Peng H, Long F, Ding C (2005) Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Trans Pattern Anal Mach Intell 27:1226–1238
28. Estévez PA, Tesmer M, Perez CA et al (2009) Normalized mutual information feature selection. IEEE Trans Neural Netw 20:189–201
29. Bishop C (1995) Neural networks for pattern recognition. Oxford University Press, Cambridge
30. Weston J, Mukherjee S, Chapelle O, et al (2001) Feature selection for SVMs. In: Advances in Neural Information Processing Systems, Denver, pp. 668–674
31. Guyon I, Weston J, Barnhill S et al (2002) Gene selection for cancer classification using support vector machines. Mach Learn 46:289–422
32. Fung G, Mangasarian OL (2000) Data selection for support vector machine classifiers. In: Proceeding of 6th knowledge discovery and data mining, Boston, pp 64–70
33. Chan TM, Zhang J, Pu J, Huang H (2009) Neighbor embedding based super-resolution algorithm through edge detection and feature selection. Pattern Recogn Lett 30:494–502
34. Hou C, Nie F, Li X, Yi D, Wu Y (2014) Joint embedding learning and sparse regression: a framework for unsupervised feature selection. IEEE Trans Cybern 44:793–804
35. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182
36. Raileanu LE, Stoffel K (2004) Theoretical comparison between the Gini index and information gain criteria. Ann Math Artif Intell 41: 77–93
37. Roffo G, Melzi S, Castellani U, et al. (2017) Infinite latent feature selection: a probabilistic latent graph-based ranking approach. In: IEEE International Conference on Computer Vision, Venice, pp 1407–1415
38. Devijver PA, Kittler J (1982) Pattern recognition: a statistical approach. Prentice Hall, London

Ruohao Xu received his Bachelor degree from Jianghan University, Wuhan, China, in 2018. He is currently pursuing the Master degree in Zhengzhou University. His research interests include neural signal processing and reinforcement learning.

Mengmeng Li received his Bachelor degree, Master from Zhengzhou University in 2014 and 2017 respectively, where he is currently pursuing

the Ph.D. degree. His current research interests include biological neural signal processing, machine learning and brain-inspired computation model.

Zhongliang Yang received his Bachelor degree from Zhoukou Normal University in 2018. He is currently pursuing the Master degree in Zhengzhou University. His research interests include neural signal processing and machine learning.

Lifang Yang received his Bachelor degree, Master from Henan University of Technology in 2016 and Zhengzhou University in 2019 respectively. She is currently pursuing the Ph.D. degree Zhengzhou University. Her research interests include neural signal processing and reinforcement learning.

Kangjia Qiao received the B.E. degree from Jianghan University, Wuhan, China, in 2018. Now, he is currently pursuing the B.S. degree at Zhengzhou University, Zhengzhou, China. His current research interests include evolutionary computing, differential evolution, multitasking optimization, and multimodal multiobjective optimization.

Dr. Zhigang Shang received the Ph.D. degree from Tongji University in 2006. He is currently a Professor with the School of Electrical Engineering, Zhengzhou University. His current research interests include neural signal processing and pattern recognition.