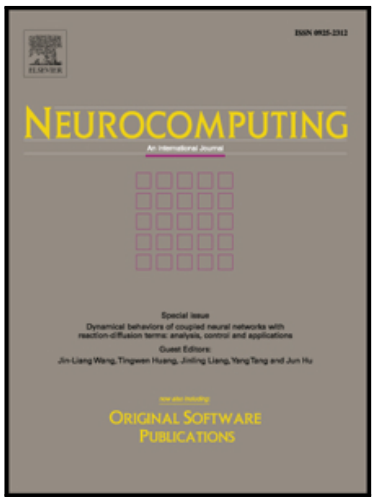

Graph classfication based on sparse graph feature selection and extreme learning machine

Yajun Yu , Zhisong Pan , Guyu Hu , Huifeng Ren

PII: S0925-2312(17)30200-X
DOI: 10.1016/j.neucom.2016.03.110
Reference: NEUCOM 17999

To appear in: *Neurocomputing*

Received date: 30 September 2015
Revised date: 12 March 2016
Accepted date: 15 March 2016

Please cite this article as: Yajun Yu , Zhisong Pan , Guyu Hu , Huifeng Ren , Graph classfication based on sparse graph feature selection and extreme learning machine, *Neurocomputing* (2017), doi: 10.1016/j.neucom.2016.03.110

Graph classfication based on sparse graph feature selection and extreme learning machine

Yajun Yu, Zhisong Pan, Guyu Hu, Huifeng Ren

College of Command Information System, PLA University of Science and Technology, Nanjing, China 210007

Abstract

Identification and classification of graph data is a hot research issue in pattern recognition. The conventional methods of graph classification usually convert the graph data to the vector representation and then using SVM to be a classifier. These methods ignore the sparsity of graph data, and with the increase of the input sample, the storage and computation of the kernel matrix will cost a lot of memory and time. In this paper, we propose a new graph classification algorithm called graph classification based on sparse graph feature selection and extreme learning machine. The key of our method is using the lasso to select features because of the sparsity of graph data, and extreme learning machine(ELM) is introduced to the following classification task due to its good performance. Extensive experimental results on a series of benchmark graph data sets validate the effectiveness of the proposed methods.

Keywords: graph kernel, graph classification, extreme learning machine, lasso

1. Introduction

Most of the existing machine learning algorithms such as support vector machine(SVM) are only applied to deal with vector type data. But in many practical applications such as bioinformatics, drug discovery, web data mining and social networks involves the study of relationships between structured objects[1], which can not be represented in vector forms. In recent years, the research about structured data has become a hot research issue in machine learning and data mining. Graphs are usually employed to represent the structured objects, and the nodes of the graph represent objects while the edges indicate the relationships between objects. To analyse graph data, the most important thing is the similarity measurement of two graphs. In order to address this issue, several methods are proposed such as graph edit distance (GED)[2] and graph kernel. Kernel method is a good way to study graph data, and kernel function can be used to measure the similarity between two graphs.

The general method of graph classification is using the graph kernel to map the graph data into higher dimensional vector described feature space, and computing the kernel matrix K consisting of each similarity of two graphs, then the original graph data set can be classified by SVM[3]. However, on one hand, this method neglects the

sparsity of the graph data. Intuitively, because of the diversity of the structure of the graph, the specific structure contained in a graph doesn't exist in the others, and with the increase of the number of nodes and edges in the graph, the sparsity of the graph will be more evident. In this paper, we will show the sparsity of the graph feature space through experiment, and use the lasso to select sparse feature.

On the other hand, SVM is suitable to solve small-scale samples, with the increase of the sample, the storage and computation of the kernel matrix will cost a lot of memory and time. Since ELM has better identification accuracy and faster speed, we use ELM to classify the graph data set.

The main contributions of this paper include:

- Using Weisfeiler-Lehman graph kernel feature mapping method[4] to test the sparsity of graph feature;
- Proposing graph feature selection via the lasso. Using the lasso, we can select the key nodes and edges of a graph, these nodes and edges form a subgraph, and the subgraph represents the original graph but has smaller size;
- Using ELM to classify graph. It's the first time proposing to use ELM to classify structure data, and this model spend less time and has better accuracy than SVM model.

The rest of this paper is organized as follows. In Section 2, we introduce the concepts of graph kernel. And we introduce the idea of the lasso for sparse graph feature selection, and verify the sparsity of the graph data feature space in Section 3. In Section 4, we describe the extreme learning machine. And we propose graph classification based on sparse graph feature selection and ELM in Section 5. Experimental results are presented in Section 6, and finally, we conclude this paper in Section 7.

2. Graph Kernel

We define a graph G as a four-tuple $(V; E; lv; le)$, where V is the set of vertices, $V = \{v_1, v_2, ..., v_n\}$, and E is the set of undirected edges, $E = \{e_1, e_2, ..., e_m\}$, and $l_v(l_e): V, E \to \Sigma$ are two functions mapping from nodes or edges in the graph to an alphabet Σ.

Kernel method has been applied widely in the analysis of vector data. Informally, a kernel is a function of two objects that quantifies their similarity. Define the kernel function between two graphs called graph kernel. Given a feature mapping φ, the graph in the original space can be mapped to the high-dimensional and even infinite

dimensional vector space, and formula (1) holds.

$$K(G_1, G_2) = \langle \varphi(G_1), \varphi(G_2) \rangle \quad (1)$$

The similarity between two graphs can be measured by the graph kernel. In recent years, several diff erent graph kernels were proposed. They can be categorized into three classes: (1) Graph kernels based on walks and paths, such as shortest-path kernels[5], Joint kernels[6] and kernels based on label pairs[7]; (2) Graph kernels based on limited size subgraphs, such as cyclic pattern kernels[8]; (3) Graph kernels based on subtree patterns such as graph kernels based on tree patterns[9] and fast subtree kernels[10].

This paper uses the Weisfeiler-Lehman subtree kernel. The key idea of the kernel is[4]: Let Σ_0 be the set of original node labels of G and G'. Assume all Σ_i are pairwise disjoint. Assume that every $\Sigma_i = \{\sigma_{i1}, ..., \sigma_{i|\Sigma_i|}\}$ is ordered. Define a map $c_i : \{G, G'\} \times \Sigma_i \to N$, so that $c_i(G, \sigma_{ij})$ is the number of occurrences of the letter σ_{ij} in the graph G. The Weisfeiler-Lehman subtree kernel on two graphs G and G' with h iterations is defined as:

$$k^{(h)}_{WLsubtree}(G, G') = \langle \phi^{(h)}_{WLsubtree}(G), \phi^{(h)}_{WLsubtree}(G') \rangle \quad (2)$$

Where

$\phi^{(h)}_{WLsubtree}(G) = \left(c_0(G, \sigma_{01}), ..., c_0\left(G, \sigma_{0|\Sigma_0|}\right), ..., c_h(G, \sigma_{h1}), ..., c_h\left(G, \sigma_{h|\Sigma_h|}\right)\right)$, and

$\phi^{(h)}_{WLsubtree}(G') = \left(c_0\left(G', \sigma_{01}\right), ..., c_0\left(G', \sigma_{0|\Sigma_0|}\right), ..., c_h\left(G', \sigma_{h1}\right), ..., c_h\left(G', \sigma_{h|\Sigma_h|}\right)\right)$.

See Figure 1, 1-5 for an illustration of the first iteration of the Weisfeiler-Lehman subtree kernel.

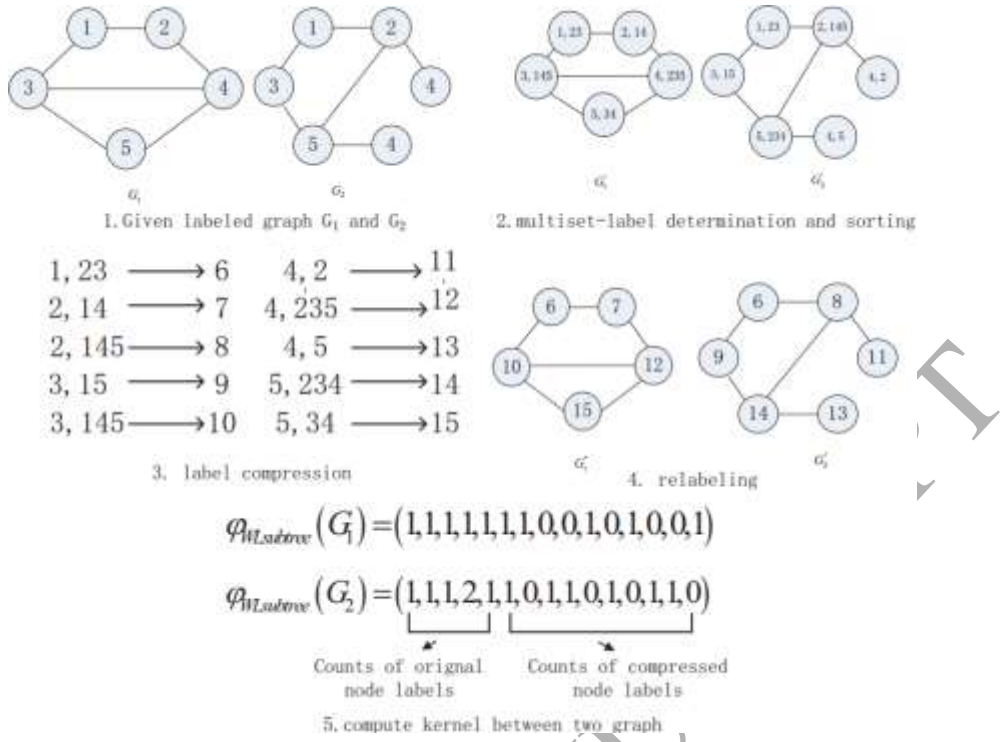

Figure 1: Illustration of the first iteration of the Weisfeiler-Lehman subtree kernel

3. Graph feature selection based on the lasso

Intuitively, if two graphs are not isomorphic, then the set of node labels between them obtained by Weisfeiler-Lehman test of graph isomorphism algorithm[4] is different, and this will result in a large part of the corresponding feature vector of the graph data are zeros. All feature vectors of graph data mapped by Weisfeiler-Lehman graph kernel form a feature matrix. Figure 2 shows the sparsity of graph data sets.

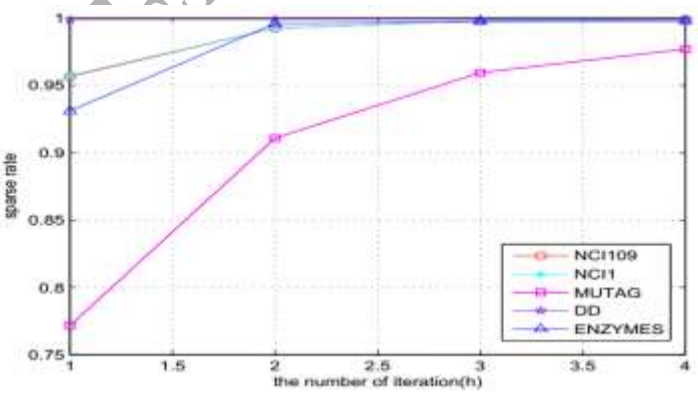

Figure 2: Sparse rate of each data sets

We use the graph data sets used in the literature 4. They are MUTAG, NCI1, NCI109, ENZYMES and DD. We use the Weisfeiler-Lehman graph kernel feature mapping method to get a feature matrix, and then calculate the proportion of the number of zero elements in all elements of each feature matrix. It can be seen from Figure 2 that when h = 4, the sparse rate of these five graph data sets nearly comes close to 99%. It means only a small number of features contribute to its classification, the features of graph data are sparse.

Because of the sparsity of graph data, when classifying graph data we should select key features of them, so that we can get features of the graph data that are most able to distinguish them. And it can not only speed up the whole learning process, but also improve the distinction accuracy rate. In order to find sparse representation of features, we utilize the lasso[11] to select features. The lasso is a classical method of feature selection based on regularization which is proposed by Tibshirani in 1996. Its original definition is:

$$\left(\hat{\alpha},\hat{\beta}\right)=\arg\min\left\{\sum_{i=1}^{N}\left(y_i-\alpha-\sum_j\beta_j x_{ij}\right)^2\right\} subjectto \sum_j|\beta_j|\le t \tag{3}$$

Consider the following classical linear regression problems, the lasso can be restate as follows: Assuming we have data $D=\left\{\left(x_i,y_i\right)\right\}, i=1,2,...N$, where x_i are the predictor variables and each x_i is a p-dimensional vector, and y_i are the responses.

$$X=\begin{pmatrix}x_1^{'}\\ .\\ .\\ x_i^{'}\\ .\\ .\\ x_N^{'}\end{pmatrix}=\begin{pmatrix}x_{11} & \cdots & x_{1j} & \cdots & x_{1p}\\ . & & . & & .\\ . & & . & & .\\ x_{i1} & \cdots & x_{ij} & \cdots & x_{ip}\\ . & & . & & .\\ . & & . & & .\\ x_N & \cdots & x_{Nj} & \cdots & x_{Np}\end{pmatrix} \tag{4}$$

Considering the linear model:

$$y=X\beta+\in \tag{5}$$

Assuming error term $\in^i$ are independently and identically distributed according to a Gaussian distribution with mean zero and some variance σ^2. Then lasso is an estimate which minimizes the residual sum of squares subject to L1 constraint:

$$\hat{\beta}=\arg\min_\beta\left\|y-X\beta\right\|_2^2 subjectto \left\|\beta\right\|_1\le s \tag{6}$$

Where $\beta\in R^P, s\ge 0, \left\|.\right\|_q$ is q-norm. The lasso can also be equivalent to Lagrangian form,

$$\hat{\beta}=\arg\min_\beta\left\|y-X\beta\right\|_2^2+\lambda\left\|\beta\right\|_1\le s \tag{7}$$

Where regularization parameter $\lambda\ge 0$ is equivalent to s in (6).

Because L_1-norm is non-differentiable, the lasso shrinks some coefficients and sets others to 0, and then achieves the purpose of feature selection[11]. λ is a turning parameter, we can control the number of the features by adjusting the λ.

Suppose the data processed by Weisfeiler-Lehman subtree graph kernel are $(x_i, y_i), i = 1, 2, 3, ..., N$, where $x_i = (x_{i1}, x_{i2}, ..., x_{ip})$ are input variables, and y_i are responses. Assume that the x_{ij} are standardized, and $\hat{\omega} = (\hat{\omega}_0, \hat{\omega}_1, ..., \hat{\omega}_p)^T$, then the lasso estimate $\hat{\omega}$ is defined by

$$\hat{\omega} = \arg\min \left\{ \sum_{i=1}^{n} \left(y_i - \sum_j \omega_j x_{ij} \right)^2 \right\} subjectto \sum_j |\omega_j| \le \lambda \qquad (8)$$

Where $\lambda \ge 0$ is a turning parameter, it controls the degree of sparsity. Appropriate parameter λ will cause shrinkage of the solutions towards 0, and some coefficients may be exactly to 0. Then the solution of this problem makes many elements in x set to zero. Formula(8) is difficult to solve since it's a non-smooth convex problem. This problem can be solved by many existing software packages, such as SLEP[12].

4. Extreme Learning Machine

ELM is a fast training algorithm for single hidden feedforward layer neural network(SLFN), it can randomly chooses the input weights and analytically determines the output weights of SLFNs[13]. So, the network parameters can be determined without any iteration step, and the adjustment time of the network parameters is greatly reduced. Compared with the traditional artificial neural networks, this method has not only the advantages of fast learning speed, but also good generalization performance. Figure 3 shows the structure of ELM network. The ELM network has received wide attention in recent years.

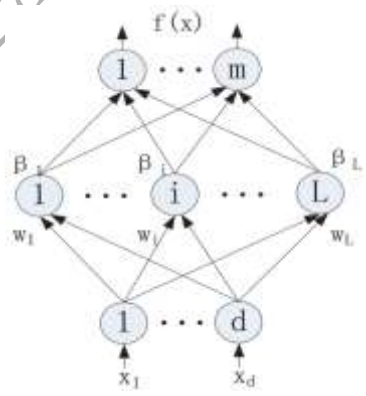

Figure 3: The structure of ELM. It contains three layers: input layer, hidden layer and output layer, where the hidden layer includes L hidden neurons, and the L is much less than N in general, the output of the output layer is a m dimensional vector

Consider *N* arbitrary distinct samples (x_i, t_i), where $x_i = (x_{i1}, x_{i2}, ..., x_{in})^T \in \mathbb{R}^n$ and $t_i = (t_{i1}, t_{i2}, ..., t_{im})^T \in \mathbb{R}^m$, an ELM with L hidden nodes and an activation function

g(x) is mathematically modeled as:

$$\sum_{i=1}^{L} \beta_i g(\omega_i \cdot x_k + b_i) = o_k, k = 1, ..., N \quad (9)$$

where $\omega_i = (\omega_{i1}, \omega_{i2}, ..., \omega_{in})^T$ is the weight vector connecting the i^{th} hidden neuron and the input neurons, $\beta_i = (\beta_{i1}, \beta_{i2}, ..., \beta_{im})^T$ is the weight vector connecting the i^{th} hidden neuron and the output neurons, b_i is the threshold of the i^{th} hidden neuron and $o_k = (\beta_{k1}, \beta_{k2}, ..., \beta_{km})^T$ is the output vector of the SLFN. $\omega_i \cdot x_k$ denotes the inner product of ω_i and x_k. The ELM with L hidden nodes and activation function g(x) reliably approximates N samples with minimum error:

$$\sum_{i=1}^{L} \beta_i g(\omega_i \cdot x_k + b_i) = t_k, k = 1, 2, ..., N \quad (10)$$

The above N equations can be written compactly as:

$$H\beta = T \quad (11)$$

Where

$$H = \begin{pmatrix} g(\omega_1 \cdot x_1 + b_i) & \cdots & g(\omega_L \cdot x_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(\omega_1 \cdot x_N + b_i) & \cdots & g(\omega_L \cdot x_N + b_L) \end{pmatrix}_{N \times m} \quad (12)$$

$$\beta = \begin{bmatrix} \beta_1^T \\ . \\ . \\ . \\ \beta_L^T \end{bmatrix}_{L \times m} \quad \text{and} \quad T = \begin{bmatrix} t_1^T \\ . \\ . \\ . \\ \beta_N^T \end{bmatrix}_{N \times m} \quad (13)$$

where H is called the hidden layer output matrix of the neural network. If the number of neurons in the hidden layer is equal to the number of samples, then H is square and invertible. Otherwise, the system of equations needs to be solved by numerical methods, concretely by solving

$$\min_{\beta} \| H\beta - T \| \quad (14)$$

The result that minimizes the norm of this least squares equation is

$$\hat{\beta} = H^{\dagger} T \quad (15)$$

Where $H^{\dagger}$ is called Moore-Penrose generalized inverse[14].

In recent years, ELM has been well developed, and more and more people pay attention to the study of ELM. Yibing Wang[15] proposed L1-Norm Minimization ELM. According to Yibing Wang's description, L1-Norm Minimization ELM not only maximally inherits the key features of ELM, which includes fast learning speed, good generalization performance and little human intervened tuning of parameters, but also extends the theory of sparse hidden nodes selection. Bo Jia[16] proposed Two-Dimensional Extreme Learning Machine(2DELM), unlike original ELM which handles vectors, 2DELM takes the matrices as input features without vectorization. Ran Yangjun[17] proposed Lasso-ELM, which utilising the regression optimisation of iteration expression Lasso, it not only can significantly decrease the number of the nodes in hidden layer of neural works, but also has a better generalization capability of neural networks

5. Proposed graph classification algorithm

Assume we have graph data set $(G_i)_{i=1}^N$, after the Weisfeiler-Lehman subtree kernel mapping, we can get a kernel matrix $K=\left(k\left(G_i,G_j\right)\right)_{N\times N}$. Then we use the lasso to select sparse features that key to classification. After that, these selected features are used to train and test an ELM classifier. Figure 4 shows the approximate process of graph classification based on sparse graph feature selection and extreme learning machine(GC-LASSO-ELM).

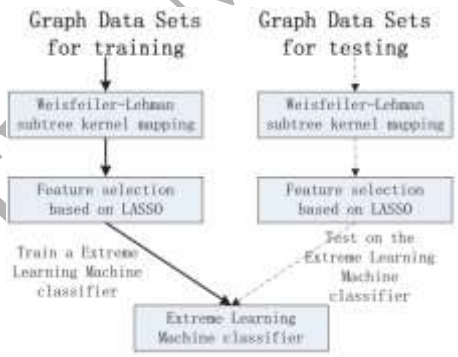

Figure 4: The approximate process of GC-LASSO-ELM

As we can see in Figure 4, our method includes three parts: Weisfeiler-Lehman subtree kernel mapping, sparse feature selection and classification using ELM. We get the optimal parameters through experiments on the training set. The detailed procedure of GC-LASSO-ELM is listed in algorithm 1.

Algorithm 1 GC-LASSO-ELM Algorithm

Require:

A graph dataset of N graphs $G=\{G_1,G_2,...,G_N\}$

Ensure:

The label of graphs.

1. Compute Kernel matrix $K = (k_{ij})_{i,j=1}^{N}$ using Weisfeiler-Lehman subtree graph kernel $k_{WLsubtree}^{(h)}(G, G') = \left\langle \phi_{WLsubtree}^{(h)}(G), \phi_{WLsubtree}^{(h)}(G') \right\rangle$.

2. Use lasso to select sparse features, and represent graph data with these new features.

3. Classify graph data processed through 1 and 2 based on ELM

6. Experimental results

6.1 Data sets

In this section, we validate our method on the following data sets: MUTAG[18], PTC_MM, PTC_FM, PTC_MR and PTC_FR. MUTAG is the data set of mutable molecules, it contains 188 chemical compounds, and it can be divided into two classes according to whether they are mutagenic or not, where 125 of them are positive and 63 are negative. The PTC[19] is the data sets of carcinogenic molecules. It contains 417 chemical compounds, and has four kinds of data sets: Male Mouse(PTC_MM), Female Mouse(PTC_FM), Male Rat(PTC_MR) and Female Rat(PTC_FR). Each molecule is assigned one of the eight labels: EE, IS, E, CE, SE, P, NE, N according to its carcinogenicity. EE, IS and E denote negative, CE, SE and P denote positive, and NE and N is considered as can not discriminant, so not involved in classification. Table 1 gives the detail information of these five graph data sets used in this paper.

Table 1: Graph Datasets

Dataset	Number of Positive	Number of Negative	Number of Total number	Number of Average node
MUTAG	125(66.5%)	63(33.5%)	188	17.93
PTC_MM	66(37.7%)	109(62.3%)	175	25.05
PTC_FM	77(42.4%)	109(58.6%)	186	25.25
PTC_MR	62(35.6%)	112(64.4%)	174	25.56
PTC_FR	61(32.6%)	126(67.4%)	187	26.08

6.2 Experiments Settings

In our method, we use the Weisfeiler-Lehman subtree kernel proposed by Nino Shervashidze because of its significant computer speed on large graph data sets, and we choose h=5 in the Weisfeiler-Lehman subtree kernel[14]. For the lasso, the parameter λ is a tunable parameter that controls the degree of the sparsity. For the ELM, the parameter involves the number of hidden neurons. We select the optimal λ and the number of hidden neurons through contrast experiment, parameters is set as follows: $\lambda = [0.3, 0.5, 0.7, 0.9]$, the number of hidden neurons between 1 and 100. Figure 5 to 9 show the classify accuracy of different λ and the number of hidden neurons on five graph data sets.

As we can see in Figure 5, we choose λ = 0.9 and 30 hidden neurons when apply our method on MUTAG because of its higher accuracy than others.

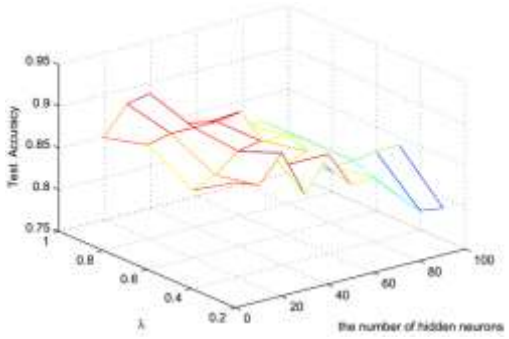

Figure 5: Classification accuracy at varying λ and number of neurons On MUTAG

Figure 6 to Figure 9 are test on PTC data sets, we choose λ = 0.9 and 40 neurons when apply our method on PTC, because it has higher accuracy on all the four data sets.

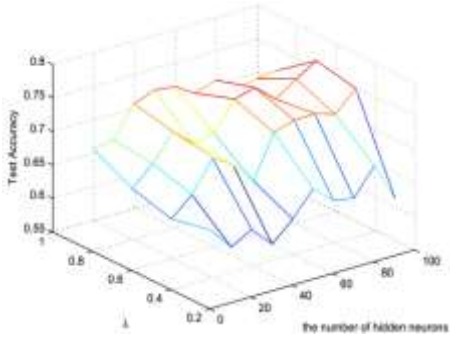

Figure 6: Classification accuracy at varying λ and number of neurons On *PTC MR*

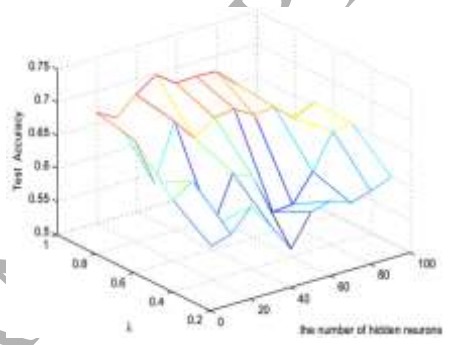

Figure 7: Classification accuracy at varying λ and number of neurons On *PTC FM*

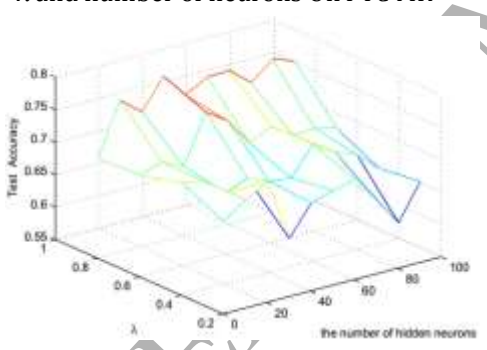

Figure 8: Classification accuracy at varying λ and number of neurons On *PTC FR*

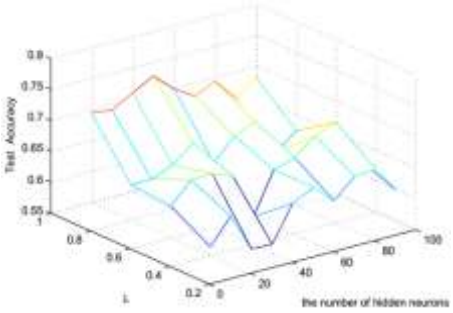

Figure 9: Classification accuracy at varying λ and number of neurons On *PTC MM*

6.3 Evaluation of classification performance

After choose the optimal λ and the number of hidden neurons, we use these chosen parameters to perform 10-fold cross-validation and using 9 folds for training 1 folds for testing on every data sets, repeat each experiment 10 times.

First, we compare four methods on MUTAG, PTC_MM, PTC_FM, PTC_MR and PTC_FR: GC-LASSO-ELM, which using the lasso to select graph features and the ELM to classify, it's the proposed method in the paper; GCKLDA-ELM, which using kernel LDA[20] to reduce dimension and the ELM to classify; GC-KPCA-ELM, which using kernel PCA[21] to reduce dimension and the ELM to classify; GK-DR[3], which using kernel PCA to reduce dimension and the SVM to

classify, it's proposed by Wu Xia. The main ingredient of KLDA is the kernel trick which allows the efficient computation of fisher discriminant in feature space. The linear classification in feature space corresponds to a non-linear decision function in input space. KPCA is an extension of PCA using techniques of kernel method. Using a kernel, the originally linear operation of PCA are done in a reproducing kernel Hilbert space with a non-liner mapping.

See Table 2, it lists the average classification accuracy and the standard deviation on each data set. Comparing GK-DR and GC-KPCA-ELM, we can see that the GC-KPCA-ELM has better accuracy than GK-DR, this indicates using ELM to classify graph data has better performance than using SVM. Comparing GC-LASSO-ELM, GC-KPCA-ELM and GC-KLDA-ELM, we know that when we using the lasso to select graph features, it has better accuracy than using KLDA and KPCA to reduce dimension. More notable is that the classification accuracy on PTC data sets of graph classification using SVM directly after graph kernel mapping are: 61.0%(PTC_MM),61.0%(PTC_FM), 62.8%(PT_MR), 66.7%(PTC_MM)[6] .

Table 2: Average classification accuracy

	GK-DR(%)	GC-KPCA-ELM(%)	GC-KLDA-ELM(%)	GC-LASSO-ELM(%)
MUTAG	83.52±0.4973	84.45±0.5035	84.59±0.9218	**86.12**±1.3126
PTC_MM	62.44±0.4367	65.88±0.3029	66.29±1.3018	**68.74**±1.0621
PTC_FM	61.42±0.6024	62.47±0.4029	65.42±1.1638	**69.43**±0.7069
PTC_MR	64.28±0.2989	65.04±0.5718	67.35±1.2862	**72.80**±2.7886
PTC_FR	67.28±0.1849	68.25±0.4621	68.61±0.6392	**71.01**±0.9501

Figure 10 shows the classification accuracies of GK-DR, GC-KPCA-ELM, GC-KLDA-ELM and GC-LASSO-ELM on MUTAG data set, we choose $\lambda = 0.9$ and 30 hidden neurons for GC-LASSO-ELM, after run the method we selected 49 features while original kernel matrix has 188 features. The dimension reduce to 5 for KPCA, and reduce to 1 for KLDA.

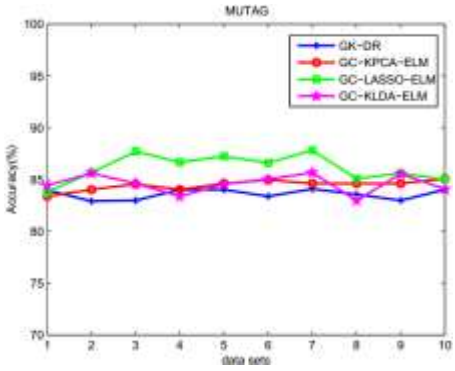

Figure 10: Average classification accuracy on MUTAG

Figure 11 to Figure 14 show the classification accuracies of GK-DR, GC-KPCA-ELM, GCKLDA-ELM and GC-LASSO-ELM on PTC data sets. we choose $\lambda = 0.9$ and 40 hidden neurons for GC-LASSO-ELM, and we selected 63 features while original kernel matrix has 175 features test on PTC_MM, we selected 76 features while original kernel matrix has 174 features test on PTC_MR, we

selected 72 features while original kernel matrix has 187 features test on PTC_FR, we selected 66 features while original kernel matrix has 186 features test on PTC_FM, The dimension reduce to 5 for KPCA, and reduce to 1 for KLDA.

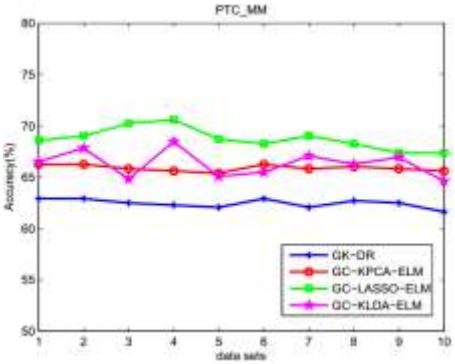

Figure 11: Average accuracy on *PTC MM*

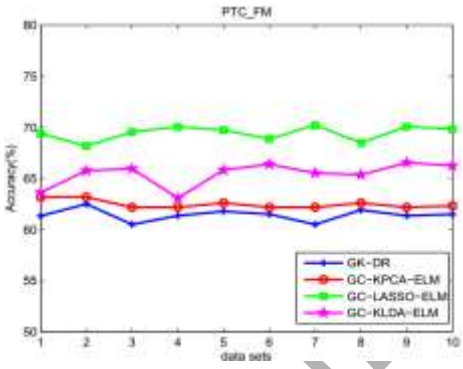

Figure 12: Average accuracy on *PTC FM*

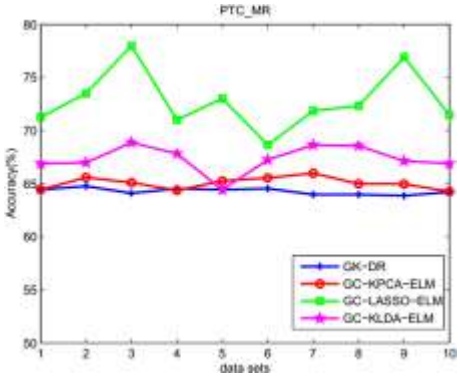

Figure 13: Average accuracy on *PTC MR*

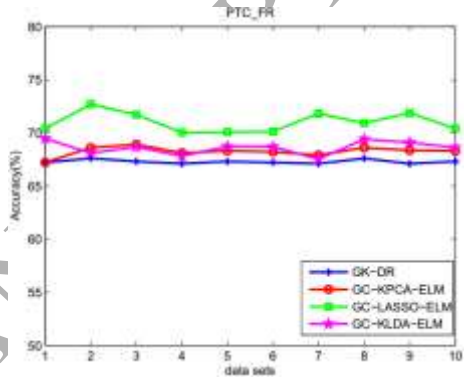

Figure 14: Averag accuracy on *PTC FR*

Second, we compare other four methods on MUTAG, PTC_MM, PTC_FM, PTC_MR and PTC_FR: GK-DR, GC-KLDA-SVM, GC-LASSO-SVM and GC-LASSO-ELM, SVM means using SVM to be a classifier after dimensionality reduction or feature selection. Table 3 lists the average classification accuracy and the standard deviation on each data set. GK-DR, GC-KLDA-SVM and GC-LASSO-SVM are using SVM to be a classifier, we can see that the GC-LASSO-SVM has better accuracy than GC-KLDA-SVM, and the GC-KLDA-SVM has better accuracy than GK-DR. Comparing GC-LASSOELM with GC-LASSO-SVM, we can see that using ELM to classify graph data has better performance than using SVM, this also indicates using ELM to classify graph data has better performance than using SVM.

Table 3: Average classification accuracy

	GK-DR(%)	GC-KLDA-SVM(%)	GC-LASSO-SVM(%)	GC-LASSO-ELM(%)
MUTAG	83.52±0.2917	83.96±0.0898	85.42±0.2393	**86.12**±0.5362
PTC_MM	62.56±0.1441	63.21±0.164	64.41±0.1403	**65.64**±0.2279
PTC_FM	60.98±0.4367	62.41±0.6443	67.56±0.6339	**68.31**±0.1613
PTC_MR	64.36±0.3601	66.04±0.1218	67.87±0.1676	**71.37**±0.1687
PTC_FR	67.40±0.0132	68.18±0.3449	70.37±0.8672	**71.17**±0.4920

Figure 15 shows the classification accuracies of GK-DR, GC-KLDA-SVM, GC-LASSO-SVM and GC-LASSO-ELM on MUTAG data set. Figure 16 to Figure

19 show the classification accuracies of GK-DR, GC-KLDA-SVM, GC-LASSO-SVM and GC-LASSO-ELM on PTC data sets. The parameters are the same with the first experiment.

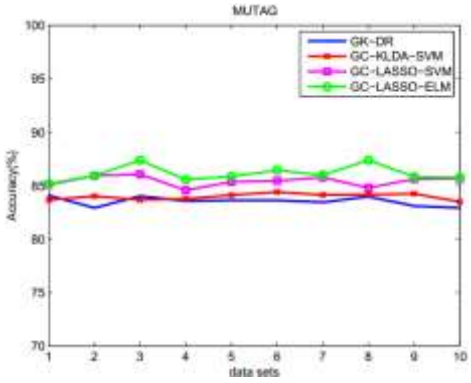

Figure 15: Average classification accuracy on MUTAG

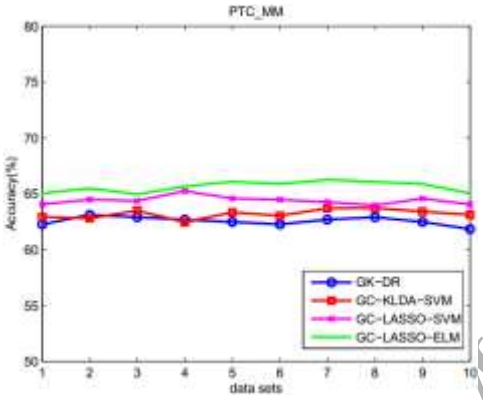

Figure 16: Average accuracy on *PTC_MM*

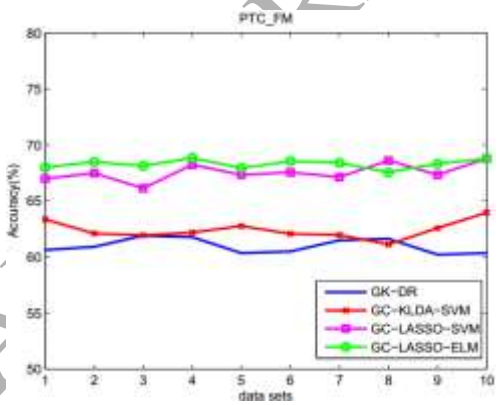

Figure 17: Average accuracy on *PTC_FM*

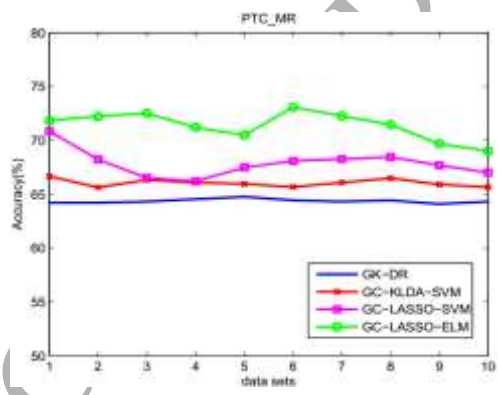

Figure 18: Average accuracy on *PTC MR*

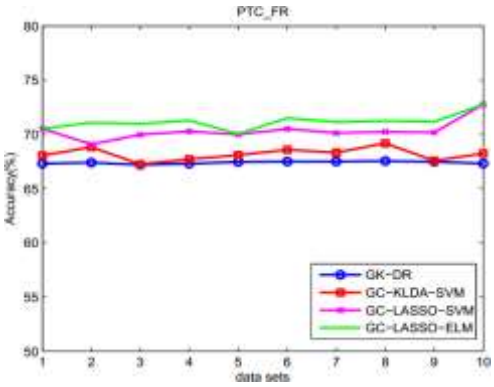

Figure 19: Averag accuracy on *PTC FR*

Figure 20 shows the average testing time of GK-DR and GC-LASSO-ELM testing on MUTAG, PTC_MM, PTC_FM, PTC_MR and PTC_FR, each experiment repeat 10 times. As we can see from this figure, the GC-LASSOELM spends less time than GK-DR on these five data sets. It indicates that our model has a better performance compared to GK-DR

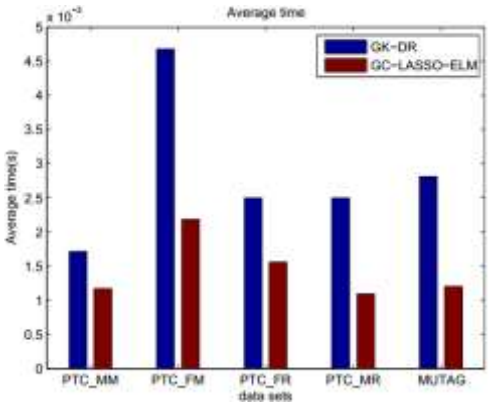

Figure 20: Average time

7. Conclusion

In this paper, an efficient graph classification method is proposed. We utilize the lasso for graph feature selection, and the ELM for classification. Experimental results on MUTAG and PTC data sets show that we have better performance when using lasso to select graph feature than using KPCA and KLDA to reduce dimension on classification accuracy, and it also shows that we have better accuracy and time performance when using GC-LASSOELM method for graph classification than using GK-DR. In the future work, we will use different kernel to study graph and compare their accuracy on ELM.

Acknowledgements

We want to express the sincere appreciation to the anonymous reviewers for their helpful comments and suggestions. Our work is supported by the National Technology Research and Development Program of China (863Program) with grant number 2012AA01A510, and the Natural Science Foundation of Jiangsu Province, China (Grant NO.BK20140073)

References

[1] S.V.N.Vishwanathan, Nicol N.Schraudilph, & Risi Kondor. 2010. Graph Kernels. Journal of Machine Learning Research 11 on (pp. 1201-1242).

[2] X.Gao, B.Xiao, D.Tao, & X.Li. 2009. Asurvey of graph edit distance. Pattern Anal. Appl.

[3] WU Xia, & ZHANG Daoqiang. 2010. Graph Kernel Based SemiSupervised Dimensionality Reduction Method. Journal of Frontiers of Computer Science and Technology.

[4] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, & Karstenn M.Borgwardt. 2011. Weisfeiler-lehman Graph Kernels. Journal of Machine Learning Research 12 on (pp. 2536-2561).

[5] Borgwardt K M & Kriegel H P. M. Shortest-path kernels on graphs. 2005. The 5th IEEE International Conference on Data Mining.

[6] Kashima H, Tsuda K & Inokuchi A. 2003. Marginalized kernels between labeled graphs. The 20th International Conference on Machine Learning(ICML).

[7] Gartner T, Flach P A & Wrobel S. 2003. On graph kernels: Hardness results and efficient alternatives. The 16th Annual Conference on Computational Learning Theory and 7th Kernel Workshop(COLT).
[8] Horvath T, Gartner T & Wrobel S. 2004. Cyclic pattern kernels for predictive graph mining. The International Conference On Knowledge Discovery and Data Mining.
[9] Mahe P & Vert J P. 2009.Graph kernels based on tree patterns for molecules. Machine Learning on (pp. 205-214).
[10] Shervashidze N & Borgwardt K. 2009. Fast subtree kernels on graphs. International Conference on Neural Information Processing Systems.
[11] Robert Tishirani. 1996. Regression Shrinkage and Selection via the Lasso. Journal of the Royal Statistical Society. Seris B, 267-288.
[12] Liu, J., Ji, S., & Ye, J. (2009). SLEP: Sparse learning with efficient projections. Arizona State University, 6.
[13] Huang, G. B., Chen, L., & Siew, C. K. (2006). Universal approximation using incremental constructive feedforward networks with random hidden nodes. Neural Networks, IEEE Transactions on, 17(4), 879-892.
[14] D. Serre, Matrices: Theory and Applications. Springer-Verlag New York, Inc, 2002
[15] Yibing Wang, Dong Li, Yi Du, & Zhisong Pan. (2015). Anomaly detection in traffic using L1-Norm Minimization Extreme Learning Machine. Neurocomputing, 149(PA):415-425.
[16] Bo Jia, Dong Li, Yi Du, Zhisong Pan & Guyu Hu. (2015). TwoDimensional Extreme Learning Machine. Mathematical Problems in Engineering, 2015:1-8.
[17] Ran Yangjun & Sun Xiaoguang. 2013 Lasso Extreme Learning Machine. Computer application and software, 30(2):6-9.
[18] Debnath A K,Compadre R D, Debnath G, et al. 1991. Structureactivity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with Molecular Orbital Energies and Hydrophobicity J Med Chem, 34: 786-797.
[19] Helma C, King R D, Kramer S, et al. 2001. The predictive toxicology challenge. Bioinformatics: 107-108.
[20] Sebastian Mika, Gunnar Ratsch, Jason Weston, et al. 1999. Fisher discriminant analysis with kernels. IEEE: 41-48.
[21] B.Scholkopf, A.Smola, K-R.Muller. 1999. Kernel principal component analysis. SV Learning, MIT Press, Cambridge, MA, pp.327-352.
[22] Huang, G. B., Zhu, Q. Y., & Siew, C. K. (2004, July). Extreme learning machine: a new learning scheme of feedforward neural networks. In Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on(Vol. 2, pp. 985-990). IEEE.
[23] Huang, G. B., & Siew, C. K. (2004, December). Extreme learning machine: RBF network case. In Control, Automation, Robotics and Vision Conference, 2004. ICARCV 2004 8th (Vol. 2, pp. 1029-1036). IEEE.
[24] Huang, G. B., & Siew, C. K. (2005). Extreme learning machine with randomly

assigned RBF kernels. International Journal of Information Technology,11(1), 16-24.
[25] Huang, G. B., Chen, L., & Siew, C. K. (2006). Universal approximation using incremental constructive feedforward networks with random hidden nodes. Neural Networks, IEEE Transactions on, 17(4), 879-892.
[26] Akusok A, Bjork K M, Miche Y, et al. High-Performance Extreme Learning Machines: A Complete Toolbox for Big Data Applications[J]. Access IEEE, 2015, PP:1-1.19
[27] Bohte S M, La Poutr J A, Kok J N. Error-Backpropagation in Temporally Encoded Networks of Spiking Neurons[J]. Neurocomputing, 2001,48(1-4):17-37.
[28] Huang G B, Ding X, Zhou H. Optimization method based extreme learning machine for classification [J]. Neurocomputing, 2010, 74(s 13):155-163.
[29] Rong H J, Ong Y S, Tan A H, et al. A fast pruned-extreme learning machine for classification problem[J]. Neurocomputing, 2008, 72(1-3):359-366

u Yajun(1990-),Male, Master Degree Candidate, Research direction: pattern recognition, machine learning and time seires prediction; E-maill:492675818@qq.com.

Pan Zhisong(1973-),Male,Professor,Doctoral tutor, Research direction: pattern recognition, machine learningandnetworksecurity, E-maill:hotpzs@hotmail.

Hu Guyu(1963 -), Male, Professor,Doctoral tutor, Research direction: intelligent information processing, cloud computing, computer network;

Ren Huifeng(1981-),Mail,Docter,post-doctoral researcher,Research direction:image processing, machine learning and pattern recognition.

E-mail: renhf316@163.com