Pattern Recognition 48 (2015) 3783–3796

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Finding the best not the most: regularized loss minimization subgraph selection for graph classification

Shirui Pan [a,*], Jia Wu [a], Xingquan Zhu [b], Guodong Long [a], Chengqi Zhang [a]

[a] Centre for Quantum Computation & Intelligent Systems, FEIT, University of Technology Sydney, Australia
[b] Department of Computer and Electrical Engineering & Computer Science, Florida Atlantic University, USA

ARTICLE INFO

Article history:
Received 18 August 2014
Received in revised form
15 May 2015
Accepted 18 May 2015
Available online 31 May 2015

Keywords:
Feature selection
Classification
Graph classification
Sparse learning

ABSTRACT

Classification on structure data, such as graphs, has drawn wide interest in recent years. Due to the lack of explicit features to represent graphs for training classification models, extensive studies have been focused on extracting the most discriminative subgraphs features from the training graph dataset to transfer graphs into vector data. However, such filter-based methods suffer from two major disadvantages: (1) the subgraph feature selection is separated from the model learning process, so the selected most discriminative subgraphs may not best fit the subsequent learning model, resulting in deteriorated classification results; (2) all these methods rely on users to specify the number of subgraph features K, and suboptimally specified K values often result in significantly reduced classification accuracy.

In this paper, we propose a new graph classification paradigm which overcomes the above disadvantages by formulating subgraph feature selection as learning a K-dimensional feature space from an *implicit* and *large* subgraph space, with the optimal K value being automatically determined. To achieve the goal, we propose a regularized loss minimization-driven (RLMD) feature selection method for graph classification. RLMD integrates subgraph selection and model learning into a unified framework to find discriminative subgraphs with guaranteed minimum loss *w.r.t.* the objective function. To automatically determine the optimal number of subgraphs K from the exponentially large subgraph space, an effective *elastic net* and a subgradient method are proposed to derive the stopping criterion, so that K can be automatically obtained once RLMD converges. The proposed RLMD method enjoys gratifying property including proved convergence and applicability to various loss functions. Experimental results on real-life graph datasets demonstrate significant performance gain.

1. Introduction

Recent years have witnessed an increasing number of applications involving objects with structural relationships, including chemical compounds in Bioinformatics [1], brain networks [2], image structures [3], and academic citation networks [4]. For these applications, graph is a natural and powerful tool for modeling and capturing dependency relationships between objects.

Unlike conventional data, where each instance is represented in a feature-value vector format, graphs exhibit node–edge structural relationships and have no natural vector representation[1]. As a result, a common practice is to transfer graphs into vectors [5–9] in structure space or in Euclidean space, so that traditional machine learning algorithms such as Support Vector Machines (SVM) and Decision Tree can be applied. In the structure space (also referred to as quotient space) [7,8], the distance relations and nature of the original data are preserved, and some geometrical and analytical concepts such as derivatives of functions on structures can be determined, so that it can be applied to solve problems in structural pattern recognition. In the Euclidean space, the structural relations may be lost, but it provides simpler and more powerful analytical techniques for data analysis. Therefore, numerous approaches [10,9,11–18] have been proposed to represent graphs in Euclidean space. The key idea of transferring graphs into vectors in Euclidean space is to extract a set of subgraphs as features and use the presence/absence of features to represent each graph. From a feature selection perspective [19], these subgraph-based algorithms follow a filter approach for graph classification, i.e., the subgraph feature selection and the subsequent model training are separated into two steps. In summary, existing filter-based graph classification methods roughly fall into the following two categories.

* Correspondence to: Centre for Quantum Computation & Intelligent Systems, FEIT, University of Technology, Sydney, Ultimo, NSW 2007, Australia. Tel.: +61 450768511, fax: +61 2 9514 4535.
E-mail addresses: shirui.pan@uts.edu.au (S. Pan), jia.wu@student.uts.edu.au (J. Wu), xzhu3@fau.edu (X. Zhu), guodong.long@uts.edu.au (G. Long), chengqi.zhang@uts.edu.au (C. Zhang).

[1] In this paper, we only consider graphs with labels but no other feature values on nodes and edges.

http://dx.doi.org/10.1016/j.patcog.2015.05.019
0031-3203/© 2015 Elsevier Ltd. All rights reserved.

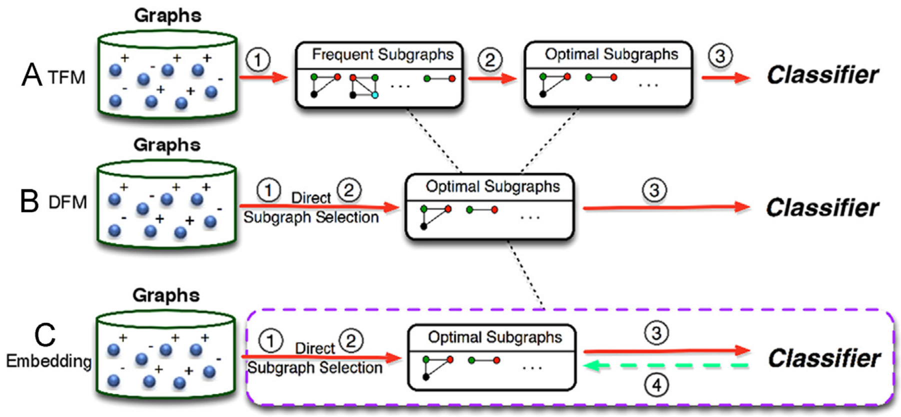


Fig. 1. Subgraph-based methods for graph classification from the feature selection perspective. TFM methods (A) sequentially perform frequent subgraph mining ①, optimal feature selection ②, and classifier learning process ③. DFM methods (B) integrate the feature selection ② into the frequent subgraph mining ① process. Our embedding method RLMD (C) unifies all steps (①②③) into a whole framework, and iterates until convergence ④.

Two-step Filter Methods (TFMs): This type of method first mines a set of frequent subgraphs as features and then applies a feature selection procedure to the discovered subgraphs, and uses the selected subgraph features to learn a classifier (*e.g.*, an SVM or Naive Bayes), as shown in Fig. 1(A). An early study [9] has shown that using frequent subgraphs as features can achieve reasonable good classification results. However, because TFMs separate subgraph feature discovery and feature evaluation into two steps, they may suffer from severe disadvantage in that the number of discovered subgraphs will grow exponentially when the minimum support value for subgraph mining is low. As a result, it will make the feature selection step heavily time-consuming. On the other hand, for relatively high minimum support values, many good subgraphs are pruned out because they do not meet the frequency requirement, so cannot be found to represent graphs.

Direct Filter Methods (DFMs): To improve the subgraph feature selection efficiency, numerous approaches [11,12,15–17] have been proposed to combine subgraph mining and feature selection into one step, representing a *direct discriminative feature selection* [18] scheme. So the feature selection is integrated into a subgraph mining process (Fig. 1 (B)), with pruning rules derived from the anti-monotone property of the significance (p-value) of each graph being used to reduce the search space. While DFMs substantially overcome the subgraph feature selection bottleneck, they also have a number of major disadvantages: (1) The subgraph selection is separated from the model learning process, so the selected subgraphs features may not best fit the underlying learning model, and (2) all these methods require users to specify the number of subgraph features K, whereas the optimal number of subgraphs K required for training a good classifier for graph classification is unknown and difficult to determine. Although subgraphs are selected using optimized measures, due to the redundancy inside the feature set, the accuracy of the classifiers, when varying the number of selected subgraph features K, is highly variable, as shown in Fig. 2. This is a common problem for all existing filter-based graph classification methods.

The above observations motivate the proposed research which *aims* to *integrate* subgraph mining, feature selection, and model training into one single framework (Fig. 1(C)) with the optimal number of subgraphs K being automatically determined for graph classification. To achieve this goal, we formulate subgraph feature selection as the problem of learning a K-dimensional feature space from a *huge subgraph space* in order to result in minimum regularized loss on the training data as follows:

$$\min_{\boldsymbol{w}} \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y_i, f(\boldsymbol{x}_i)) + \gamma R(\boldsymbol{w}) \tag{1}$$

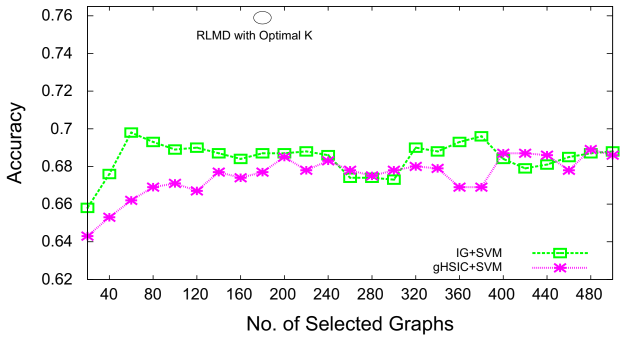


Fig. 2. Classification accuracy for filter subgraph-based methods *w.r.t.* different numbers of subgraphs on the NCI-1 chemical compound dataset. IG is a TFM method which uses information gain to select subgraphs, whereas gHSIC [12] is a DFM method. All methods use SVM as a base classifier. The optimal number of subgraph features K is crucial, but difficult to decide for filter methods. In comparison, the proposed method (RLMD) automatically finds 180 best subgraphs and achieves the highest accuracy, which is 6% more accurate than the second best method.

where $\{\boldsymbol{x}_i, \dots, \boldsymbol{x}_n\}$ are the vector representations of the training graphs, $\mathcal{L}$ is a loss function measuring the difference between the prediction $f(\boldsymbol{x}_i)$ and the true label y_i, and $R(\boldsymbol{w})$ is a regularization term on parameters $\boldsymbol{w}$ to avoid over-fitting.

Indeed, the optimization in (1) has been widely studied [20–22] in machine learning community, but mainly for data with vector format. Several significant challenges remain for graph data:

1. *Implicit Subgraph Features:* For graph classification, no subgraph features are readily available (i.e., $\boldsymbol{x}_i$ is unknown) for training the model in (1). Instead, the feature space used to represent graphs is implicit and needs to be discovered by subgraph mining procedure as needed.
2. *K-dimensional Features from Huge Subgraph Space:* The number of subgraph candidates representing graphs is exponentially large. Finding an optimal number of K subgraphs for different graph datasets (in order to result in best classifiers), is crucial but has not been addressed by existing research.

In this paper, we propose a *unified* regularized loss minimization-driven (RLMD) graph classification framework. Our theme is to progressively select the most discriminative subgraph features from the training data in order to achieve minimum regularized loss for a well defined objective function. To integrate subgraph selection into the model learning process (*Challenge 1*), we formulate an objective function and design a subgradient method to induce a measurement to assess the utility of each subgraph, so that the best subgraph features can be identified and incrementally included to optimize the objective function for maximum performance gain. To determine the optimal K value for each dataset (*Challenge 2*), we use an *elastic net* [21] and derive a stopping condition, so that the K value can be automatically obtained when the algorithm converges. By using the automatically determined optimal K value, as shown in Fig. 2, RLMD finds 180 best subgraphs and achieves the best performance, which is 6% more accurate than the second best method.

The main contributions of this paper are summarized as follows:

- We propose an *embedded* and *theoretically convergent* graph classification algorithm, which can automatically determine the optimal number of subgraphs K for graph classification. This is a *unified* approach in the sense that (1) it can employ any differentiable loss function (including least squares, exponential, and logistic loss functions) for graph classification; and (2) it integrates subgraph mining, feature selection, and model learning into one single framework.
- We generalize the *column generation technique* of gBoost [23] for graph classification, and demonstrate that gBoost [23] is a special case of our loss minimization algorithm.
- We propose the use of *elastic net* (which integrates two sparsity-inducing regularization norms, ℓ_1-norm and ℓ_2-norm) to produce a sparse and robust solution for discriminative subgraph selection.
- We derive a branch-and-bound rule according to the subgradient of our objective function to prune search space for optimal subgraph mining.
- Experimental results show that our algorithm RLMD outperforms two-step filter methods (TFMs), direct filter methods (DFMs), and gBoost algorithm with significant performance gain.

2. Related work

Our work is closely related to graph-based learning and graph classification.

2.1. Graph-based learning

Learning from data with dependency structures has been commonly addressed by existing research. Instead of considering samples as *I.I.D* observations, graph-based learning takes the relationships/correlations between samples to ensure effective learning. For example, graph-based approaches have been popularly used to propagate labels in semi-supervised learning [24–26], where training samples are connected through one or multiple graphs. A recent method [27] considers preserving global and local structures inside the training data for feature selection. For large scale networks, predicting linkage relationships between nodes (i.e. link prediction) can be used for friendship recommendation in social networks [28], or suggesting potential interactions between proteins in bioinformatics research. A recent work [29] proposed to use latent feature kernels to support link prediction on sparse graphs. All above methods consider a large scale network with thousands (or millions) of integer-connected nodes in the network. In contrast, we consider small graph classification problem, in which each graph has a label indicating the property of the graph, and the graph normally contains tens or several hundreds of nodes. The purpose is to predict the label of the graph by using node and structure information inside the graphs, for purposes such as chemical compound activity prediction [1] and gender classification using magnetic resonance connectome (i.e. brain-graph) [2].

2.2. Graph classification

Existing methods for graph classification [18,10,9,11–17,23,30,31] can be roughly categorized into two groups: similarity-based methods and vector representation-based methods.

2.2.1. Similarity-based methods

These approaches aim to directly learn global similarities between graphs by using graph kernels [9,32–34] or graph embedding [35]. Global similarities are then fed to similarity-based classifiers, such as KNN or SVM, for learning. One clear drawback of global similarity-based approaches is that the similarity is calculated based on global graph structures, such as random walks or embedding space. Therefore, it is not clear which substructures are more important for classifying graphs into different classes.

2.2.2. Vector representation-based methods

Another branch of methods transfer graphs into vector representations in structure space or in Euclidean space. In structure space [7,8], geometrical and analytical concepts such as the angle between structures and the derivatives of functions on structures can be obtained, so that the structural pattern recognition problems can be formulated as optimization problems with certain cost functions. In Euclidean space, the goal is to transfer graphs into vector representations in Euclidean space so existing analytical techniques can be applied for data analysis. Methods in this category are mainly filter-based approaches, including two-step filter methods (*TFMs*) or direct filter methods (*DFMs*).

TFMs are straightforward approaches for graph classification which simply decompose frequent subgraph generation and selection as two separated steps. An early work [9] has shown that learning an SVM classifier based on the discovered frequent subgraphs can achieve reasonably good accuracy for graph classification. On the other hand, research [16,15] also indicates that TFM methods may result in a bottleneck for the subsequent feature selection module. Specifically, the number of frequent subgraphs will grow exponentially if the minimum support threshold is low, which imposes a great challenge for the subsequent feature selection task. This challenge has motivated many direct filter methods (*DFMs*), which seek to integrate subgraph discovery and feature selection into one step.

For *DFMs* (a review on this category can be found in [18]), a key issue is to define a proper measurement to assess the utility of each subgraph. Yan et al. [17] proposed a LEAP algorithm to exploit the correlation between structural similarity and significance similarity, so that a branch-and-bound rule can be derived to prune out unpromising searching space efficiently. Ranu and Singh [16] proposed a scalable GraphSig algorithm, which is able to mine significant subgraphs with low frequencies. Thoma et al. [15] propose a CORK algorithm to find subgraph features. Recently, researchers have extended DFM to other graph applications, and have proposed effective algorithms such as gSemi [11] for the semi-supervised setting, gCGVFL [36] for multi-view learning, gHSIC [12] for multi-label classification, and our recent multi-graph classification for classifying graph bags, each containing multiple graphs [37,38].

Although filter methods for graph classification have been extensively studied, they all suffer from two major disadvantages: (1) the feature selection is not linked to the model learning process. As a result, the selected subgraph features may not best fit the underlying learning algorithms; and (2) the optimal number of subgraphs K for graph classification is difficult to decide and often varies from dataset to dataset, and inappropriately specified K value often results in significantly reduced classification accuracy. This is the common drawback for filter-based methods [19].

Embedded Methods: Our algorithm belongs to the embedded approach which integrates the subgraph selection into the model training process. In this subcategory, Saigo et al. [23] proposed a gBoost (its variants for imbalanced graph classification [39,40] and cost-sensitive learning [41] are proposed recently) algorithm which formulates the graph classification as a linear program. As will be elaborated in Section 4.6, our algorithm is more general in the sense that it can adopt any differentiable loss function and use more robust regularization to produce better performance. In fact, gBoost [23] can be considered as a special case of our loss minimization problem.

3. Problem Definition

Definition 1. Connected Graph: A graph is denoted by $G=(\mathcal{V},E,L=\{L_1,L_2\},\mathcal{A}=\{\mathcal{A}_1,\mathcal{A}_2\})$, where $\mathcal{V}$ is the vertex set, $E\subseteq\mathcal{V}\times\mathcal{V}$ is the edge set, $\mathcal{A}=\{\mathcal{A}_1,\mathcal{A}_2\}$ with $\mathcal{A}_1$ and $\mathcal{A}_2$ being the set of labels for vertices and edges, respectively; and $L=\{L_1,L_2\}$, $L_1:\mathcal{V}\rightarrow\mathcal{A}_1$, $L_2:E\rightarrow\mathcal{A}_2$ are labeling functions that assigns labels to a node or an edge, respectively. A connected graph is a graph such that there is a path between any pair of vertices.

In this paper, we focus on connected graphs and assume that each graph G has a class label y, $y\in\mathcal{Y}=\{-1,+1\}$. We only focus on binary-class classification tasks, but our methods can be easily extended to multi-class tasks.

Definition 2. Subgraph: Given two graphs $G=(\mathcal{V},E,L=\{L_1,L_2\},\mathcal{A}=\{\mathcal{A}_1,\mathcal{A}_2\})$ and $g_k=(\mathcal{V}',E',L'=\{L_1',L_2'\},\ \mathcal{A}'=\{\mathcal{A}_1',\mathcal{A}_2'\})$, g_k is a subgraph of G (i.e., $g_k\subseteq G$) if there is an injective function $\hat{f}$: $\mathcal{V}'\rightarrow\mathcal{V}$, such that $\forall(a,b)\in E'$, we have $(\hat{f}(a),\hat{f}(b))\in E$, $L_1'(a)=L_1(\hat{f}(a))$, $L_1'(b)=L_1(\hat{f}(b))$, $L_2'(a,b)=L_2(\hat{f}(a),\hat{f}(b))$. If g_k is a subgraph of G ($g_k\subseteq G$), G is a supergraph of g_k ($G\supseteq g_k$).

Subgraph-based Graph Classification: Given a set of labeled graphs $\mathcal{T}=\{(G_1,y_1),\ldots,(G_n,y_n)\}$, subgraph-based graph classification *aims* to select an optimal set of discriminative subgraphs $\mathcal{F}_1$ from $\mathcal{T}$, and learn a classification model from the reduced subgraph space $\mathcal{F}_1$ to predict previously unseen test graphs with a maximum accuracy. Set $\mathcal{F}_1$ is optimal if the classifier learned from $\mathcal{F}_1$ has the highest classification accuracy, compared to classifiers trained from any subset of $\mathcal{T}$. A major feature of our method is that it can automatically determine the best set of subgraph features to represent each graph datasets without requiring users to specify the number of subgraph features. This essentially advances the existing subgraph feature-based graph classification methods from finding the most discriminative subgraph features to finding the best subgraph set for maximum accuracy gain.

4. Regularized loss minimization for graph classification

To support graph classification, state-of-the-art algorithms [23,10] use a set of subgraphs discovered from the training graphs as features, where each subgraph g_k can be used to represent a graph G_i as follows:

$$\hbar_{g_k}(G_i)=2I(g_k\subseteq G_i)-1. \tag{2}$$

Here $I(a)=1$ if a holds, and 0 otherwise. This rule simply maps a graph G_i into $+1$ if $g_k\subseteq G_i$, or -1 otherwise.

Let $\mathcal{F}=\{g_1,\ldots,g_m\}$ be the full set of subgraphs for the training graphs. We can use $\mathcal{F}$ as features to represent each graph G_i into a vector space as $\boldsymbol{x}_i=\{\hbar_{g_1}(G_i),\ldots,\hbar_{g_m}(G_i)\}$, with $\boldsymbol{x}_i^k=\hbar_{g_k}(G_i)$. In the following subsection, G_i and $\boldsymbol{x}_i$ are used interchangeably as they both refer to the same graph. Given the full subgraph features $\mathcal{F}$, the prediction function for the graph $\boldsymbol{x}_i$ is a linear classifier:

$$f(\boldsymbol{x}_i)=\boldsymbol{x}_i\cdot w+b=\sum_{g_k\in\mathcal{F}}w_k\hbar_{g_k}(G_i)+b \tag{3}$$

where $\boldsymbol{w}=[w_1,\ldots,w_m]'$ is the weight vector for all features $\mathcal{F}$, and b is the bias of the model. The predicted class of $\boldsymbol{x}_i$ is $+1$ if $f(\boldsymbol{x}_i)>0$ or -1 otherwise. Note that in practice, subgraph space $\mathcal{F}$ is *implicit* and exponentially *large*, i.e., the number of subgraphs grows exponentially with respect to the number of nodes.

4.1. Regularized loss minimization formulation

In this paper, we propose to learn a K-dimensional feature space from the *implicit* and *large* subgraph space $\mathcal{F}$ to achieve the lowest regularized empirical risks for the graph dataset, with K being automatically determined. Eq. (1) can be reformulated as the following objective function:

$$\min\mathcal{J}(\boldsymbol{w},\boldsymbol{b})=\min_{\boldsymbol{w},\boldsymbol{b}}\underbrace{\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i,\boldsymbol{x}_i\cdot w+b)}_{\mathcal{C}}\ \underbrace{+\gamma_1\|\boldsymbol{w}\|_1+\gamma_2\|\boldsymbol{w}\|_2^2}_{R} \tag{4}$$

The first term $\mathcal{C}$ measures the loss on the training graphs, where $\mathcal{L}(y_i,f(\boldsymbol{x}_i))$ can be any loss function measuring the misclassification penalty of a graph G_i. The second part R consists of regularization terms to enforce sparse and robust solutions. Parameters γ_1 and γ_2 are used to trade-off these parts ($\gamma_1\geq0,\gamma_2\geq0$). For the regularization, our objective is to obtain a sparse and stable solution on $\boldsymbol{w}$, i.e., low dimensional subgraph features for final graph classification. Here, we combine both ℓ_1 and ℓ_2 norm, which is known as *elastic net* in machine learning [21]. The motivation of our regularization is as follows: The ℓ_1-norm regularizer $(\sum_k|w_k|)$ can produce solutions with many coefficients being 0, which is known as lasso [20] and has been widely applied for variable selections.

Although ℓ_1 regularization can produce a sparse solution, it suffers from two major disadvantages: (1) the number of selected variables is limited by the number of observations; and (2) the lasso penalized model can only select one variable from a group of correlated variables and does not care which one is selected [21]. In contrast, ℓ_2 regularization, which is widely used in SVM formulation $(\|\boldsymbol{w}\|_2^2=\sum_{k=1}^{m}|w_k|^2)$, can produce more stable and robust classification results. However, ℓ_2 formulation cannot produce a sparse solution. By combining ℓ_1 and ℓ_2 norm, known as elastic net [21], we can overcome these issues and enjoy the sparse and stable properties.

4.2. Sparse subgraph learning: challenges and solution overview

Challenges: For explicit vector data with moderate feature size, the problem defined in (4) can be effectively solved in traditional supervised learning. However, for graph data the challenges are evident: (1) the feature set $\mathcal{F}$ is unavailable (*implicit*) unless we enumerate all subgraphs from the training graphs, which is NP-complete; and (2) the whole subgraph set is exponentially *large*, and only a small subset of subgraphs are useful for classifiers to achieve maximum graph classification accuracy.

Solution Overview: To solve the aforementioned challenges, we propose a regularized loss minimization-driven (RLMD) subgraph selection method for graph classification. Driven by our formulation in (4), our principle is to iteratively mine the best subgraph feature to reduce the empirical loss on the training graphs. To this end, we resort to the subgradient method in the functional space to define the utility of each subgraph, and embed the feature selection/ranking into the subgraph mining/enumeration process. To handle the exponentially large subgraph space, we derive an effective branch-and-bound pruning scheme to reduce the search space. After a new subgraph is selected, we include and re-solve the new restricted objective function of Eq. (4) by using currently selected subgraphs. To find optimal K value, we derive a stopping criterion for our feature selection procedure based on the subgradient in the functional space, so that K can be automatically obtained once the algorithm converges.

Logistic Loss Function: A Running Example. Our method is based on the gradient/subgradient on the functional space of the objective function (4). In this paper, we use the logistic loss function as an example to illustrate how subgraph selection is performed by subgradient methods, and the logistic loss function is given as follows:

$$\mathcal{L}(y_i, f(x_i)) = \log\left(1+\exp\{-y_i f(\boldsymbol{x}_i)\}\right) \tag{5}$$

Note that our algorithm is a general method in the sense that any other differentiable loss function, such as least square loss $\mathcal{L}(y,f_t)=1/2(y-f_t)^2$ or exponential loss $\mathcal{L}(y,f_t)=\exp\{-yf_t\}$, can be directly used in our algorithm. As discussed latter in Section 4.6, our method is also applicable to convex locally Lipschitz but non-differentiable loss functions such as the hinge loss used by the (margin) perceptron and linear SVM.

The partial derivative of the loss term $\mathcal{C}$ in (4) on the subgraph feature g_k is defined as $\partial\mathcal{C}/\partial w_k$. For logistic loss function,

$$\begin{aligned}\frac{\partial\mathcal{C}}{\partial w_k} &= \frac{1}{n}\sum_{i=1}^{n}\frac{\partial\mathcal{L}(y_i,f(\boldsymbol{x}_i))}{\partial f(\boldsymbol{x}_i)}\frac{\partial f(\boldsymbol{x}_i)}{\partial w_k}\\ &= -\frac{1}{n}\sum_{i=1}^{n}\frac{y_i\boldsymbol{x}_i^k}{1+e^{y_i f(\boldsymbol{x}_i)}} = \sum_{i=1}^{n} y_i\alpha_i\boldsymbol{x}_i^k\end{aligned} \tag{6}$$

Here, $\alpha_i = -1/n(1+e^{y_i f(\boldsymbol{x}_i)})$ can be regarded as a weight associated with graph G_i for the subgraph mining process.

4.3. RLMD subgraph selection for graph classification

Because we aim to learn a sparse solution of subgraph features (K-dimensional feature space) from graph data, some subgraphs/features g_k with zero weights, i.e., $w_k=0$ will not be used for learning the classification model. Thus it makes sense to partition the subgraph features $\mathcal{F}$ into two disjoint subsets $\mathcal{F}_1$ and $\mathcal{F}_2$. $\mathcal{F}_1$ stores active features which are used to learn the classification model and this set is frequently updated as desired, and $\mathcal{F}_2$ includes unselected graphs with 0 weights (i.e., for $g_k\in\mathcal{F}_2, w_k=0$). Then we can iteratively select the best feature from $\mathcal{F}_2$ to $\mathcal{F}_1$, and solve the following restricted subproblem:

$$\begin{aligned}\min\mathcal{J}_t(\boldsymbol{w},b) &= \min_{\boldsymbol{w},b}\underbrace{\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}\left(y_i,\boldsymbol{x}_i(t)\cdot\boldsymbol{w}+b\right)}_{\mathcal{C}'}+\underbrace{\gamma_1\|\boldsymbol{w}\|_1+\gamma_2\|\boldsymbol{w}\|_2^2}_{\mathcal{R}'}\\ &= \mathcal{J}_t(\boldsymbol{w}(t),b(t))\end{aligned} \tag{7}$$

where $\boldsymbol{x}_i(t)$ is the feature representations for graph G_i based on the active set $\mathcal{F}_1$ in the tth iteration, and $(\boldsymbol{w}(t),b(t))$ is the solution of Eq. (7). Note that $\mathcal{J}_t(\boldsymbol{w},\boldsymbol{b})$ is used here to denote the restricted subproblem Eq. (7) while $\mathcal{J}(\boldsymbol{w},\boldsymbol{b})$ is referred to original problem in (4).

The optimal number of subgraphs K can be automatically determined by setting $K=|\mathcal{F}_1|$ once the algorithm converges. Note that given a solution $(\boldsymbol{w}(t),b(t))$, the loss term $\mathcal{C}'$ in (7) equals to $\mathcal{C}$ in (4) because the prediction of graph mainly depends on the active set $\mathcal{F}_1$, i.e., $\mathcal{C}'=\mathcal{C}$. In the following, we will derive the stopping condition of our algorithm, and prove its convergence.

Stopping Condition for Optimal K value: Our objective function (4) is convex and non-smooth, i.e., it may be non-differentiable at a point $\boldsymbol{w}$. When it is non-differentiable at $\boldsymbol{w}$, we can compute its generalized gradient (i.e., subgradient) instead. According to the optimization conditions, when reaching the optimum, we will have

$$0\in\frac{\partial\mathcal{C}}{\partial w_k}+\gamma_1 o_k+2\gamma_2 w_k; \tag{8}$$

where o_k is the subgradient with respect to w_k

$$o_k\in\begin{cases}\operatorname{sign}(w_k) & : w_k\neq 0\\ [-1,1] & : w_k=0\end{cases} \tag{9}$$

where $\operatorname{sign}(a)=1$ if $a>0$ otherwise -1. According to Eqs. (8) and (9), we can now state the optimal condition for our sparse subgraph learning problem.

Proposition 1. *Optimal Solution: Let* $\hat{\boldsymbol{w}}=[\hat{w}_1,\ldots,\hat{w}_m]$. *Suppose that* $(\hat{\boldsymbol{w}},\hat{b})$ *is the optimal solution of our objective function* (4), *then Eqs.* (10) and (11) *hold.*

$$\frac{\partial\mathcal{C}}{\partial\hat{w}_k}+\gamma_1\operatorname{sign}(\hat{w}_k)+2\gamma_2\hat{w}_k=0 \quad \text{if} \quad \hat{w}_k\neq 0 \tag{10}$$

$$\left|\frac{\partial\mathcal{C}}{\partial\hat{w}_k}\right|\leq\gamma_1 \quad \text{if} \quad \hat{w}_k=0 \tag{11}$$

Eq. (11) holds because for $\hat{w}_k=0$, the third term of Eq. (8) disappears. Combining (8) and (9) will result in (11).

To reduce the objective value $\mathcal{J}_t$ in (7), we propose to select a subgraph in $\mathcal{F}_2$ whose weight violates Eq. (11), and update the selected active set $\mathcal{F}_1$ with the newly selected feature and re-optimize the restricted subproblem Eq. (7) with current features. This process will repeat until no candidate violates Eq. (11). In other words, Eq. (11) is a stopping condition and determines the number of subgraphs being selected for RLMD's subgraph selection process.

Utility of Subgraphs: Eq. (11) can be used naturally to induce a criterion for quantifying the utility value of a subgraph. The larger $|\partial\mathcal{C}/\partial w_k|$ is, the more informative it will be for reducing the objective function. Accordingly, we formally define the informative score as follows:

Definition 3. Informative Score: *For a subgraph pattern* g_k, *its informative score for graph classification is defined as follows:*

$$\Theta(g_k)=\left|\frac{\partial\mathcal{C}}{\partial w_k}\right|=\left|\frac{\partial\mathcal{C}'}{\partial w_k}\right|=\left|\sum_{i=1}^{n}y_i\alpha_i\boldsymbol{x}_i^k\right| \tag{12}$$

where $\alpha_i=-\frac{1}{n(1+e^{y_i f(\boldsymbol{x}_i)})}$.

Note that the informative score directly depends on the weight of each graph α_i, which is calculated based on the active set $\mathcal{F}_1$. Intuitively, the best subgraph of $\mathcal{F}_2$ is the one with the maximum informative score, because it is more likely to violate the stopping condition (11).

RLMD Algorithm: Algorithm 1 illustrates the detailed steps of RLMD for graph classification. Initially, the weights for all training graphs are equally set as $1/n$, and the active set $\mathcal{F}_1$ is initialized to be empty.

In the next step, the algorithm mines an optimal subgraph $g^\star$ from $\mathcal{F}_2$ which has the highest informative scores defined by Eq. (12). This step involves a subgraph mining procedure, which will be addressed in the next subsection. On steps 4–5, if current optimal subgraph no longer violates the optimal condition (11), the algorithm terminates. Here, we have relaxed the convergence condition to ε tolerance; this is because in the last few iterations, the maximum score will only change subtly (we set $\epsilon=0.005$ in our experiments).

Algorithm 1. Regularized loss minimization-driven subgraph selection (RLMD) for graph classification

```
REQUIRE
  $\{(G_1,y_1),\dots,(G_n,y_n)\}$ : Training Graphs;
  $S_{max}$: Maximum number of iterations;
ENSURE
  $\mathbf{w},\mathbf{b}$: Parameters for classifier model
1:$\alpha_i=1/n$; $\mathcal{F}_1\leftarrow\varnothing$; $t\leftarrow 0$;
2:WHILE $t<S_{max}$ do
3: Mine an optimal subgraph features $g^\star$ with maximum
     informative score defined by (12) ;//Algorithm 2;
4:IF $\Theta(g^\star)\le\gamma_1+\varepsilon$ then
5:break;
6:end if
7:$\mathcal{F}_1\leftarrow\mathcal{F}_1\bigcup g^\star$;
8:Solve (7) based on $\mathcal{F}_1$ to get the new solution $(\mathbf{w}(t),b(t))$;
9: Update the graph weights on each training graph
     $\alpha_i=-\frac{1}{n(1+e^{y_i f(\mathbf{x}_i(t))})}$
10: $t\leftarrow t+1$;
11:end while
12: $K=|\mathcal{F}_1|$;
return
$\mathbf{w},\mathbf{b}$;
```

On step 7, we add the newly selected subgraph $g^\star$ to the existing subgraph set $\mathcal{F}_1$, and re-solve the following restricted subproblem Eq. (7). To solve this restricted objective function, we use the MALSAR toolbox[2] in our experiments. It is worth noting that because this step only involves a very small number of features, it is very efficient in practice.

Subsequently, the algorithm updates the weight α_i for each graph G_i. This will help compute the derivative of $\partial\mathcal{C}/\partial w_k$ for subgraph mining in next round. After the algorithm terminates, the optimal number of subgraphs K can be easily obtained as $K=|\mathcal{F}_1|=t$ on step 12.

Note that our algorithm 1 generalizes the column generation technique in gBoost [23] by iteratively selecting the most violated subgraph in each iteration until convergence. Our algorithm 1 relies on ε and γ_1, which serve as a stopping condition to determine K. In practice, ε is a subtle value insensitive to the algorithm performance. Meanwhile, γ_1 is much easier to set than asking users to specify K values because γ_1 is chosen in a much smaller range, as we will demonstrate in Section 5.2.4.

4.4. Theoretical study

Theorem 1. *(Convergence Property) Algorithm 1 guarantees that the restricted objective function (7) will monotonically decrease.*

Proof 1. Suppose in the tth iteration, the optimal objective value based on current t features (i.e., $|\mathcal{F}_1|=t$) is obtained with $(\mathbf{w}(t),b(t))$, i.e.,

$$\mathcal{J}_t(\mathbf{w}(t),b(t))=\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}\big(y_i,\mathbf{x}_i(t)\cdot\mathbf{w}(t)+b(t)\big)_{\mathcal{C}=\mathcal{C}'}+\underbrace{\gamma_1\|\mathbf{w}(t)\|_1+\gamma_2\|\mathbf{w}(t)\|_2^2}_{\mathcal{R}=\mathcal{R}'}=(\mathcal{C}+\mathcal{R})_{|(\mathbf{w}(t),b(t))}$$

then in the $t+1$th iterations, the optimal objective value for Eq. (7) is

$$\min\mathcal{J}_{t+1}(\mathbf{w},\mathbf{b})=\min(\mathcal{C}+\mathcal{R})\le(\mathcal{C}+\mathcal{R})_{|([\mathbf{w}(t),0,b(t))}=\mathcal{J}_t(\mathbf{w}(t),b(t))$$

Here $[\mathbf{w}(t),0]$ means that the weights for subgraphs selected in the tth iteration remain unchanged while the weight for newly selected subgraph in the $(t+1)$th iteration is 0.

Thus the objective value of the restricted problem Eq. (7) based on the currently selected features $\mathcal{F}_1$ always monotonously decreases in two successive iterations. Because the objective function value is non-negative (bounded), we can ensure that it will finally converge as iteration continues. The proof is complete.

Suppose the algorithm converges in the Kth iteration with a solution $(\mathbf{w}(K),b(K))$, and the objective value for Eq. (7) is $\mathcal{J}_K(\mathbf{w}(K),b(K))$. By adding $m-K$ zeros for subgraphs in $\mathcal{F}_2$ to $\mathbf{w}(K)$, *i,e.,* $\hat{\mathbf{w}}(K)=[\mathbf{w}(K),0\cdots]$, we obtain a solution $\big(\hat{\mathbf{w}}(K),b(K)\big)$ for Eq. (4).

Corollary 1. *Optimal Solution Guarantees: If Algorithm 1 converges with solution $(\mathbf{w}(K),b(K))$ for Eq. (7), then $\big(\hat{\mathbf{w}}(K),b(K)\big)$ is an optimal solution for Eq. (4).*

Proof 2. According to our Proposition 1, $\big(\hat{\mathbf{w}}(K),b(K)\big)$ is an optimal solution of Eq. (4), because $\forall w_k=0$ $(g_k\in\mathcal{F}_2)$, we have $\Theta(g_k)<\gamma_1$ based on our stopping condition. Thus we will have

$$\mathcal{J}_K(\mathbf{w}(K),b(K))=\mathcal{J}\big(\hat{\mathbf{w}}(K),b(K)\big)=\min\mathcal{J}(\mathbf{w},\mathbf{b})$$

where $J_K(\mathbf{w}(K),b(K))$ and $\mathcal{J}\big(\hat{\mathbf{w}}(K),b(K)\big)$ refer to the objective values of the restricted subproblem Eqs. (7) and (4), respectively.

We have proved that objective value for Eq. (7) is monotonously decreasing (Theorem 1) and its recovered solution $\big(\hat{\mathbf{w}}(K),b(K)\big)$ is an optimal solution to Eq. (4) (Corollary 1).

4.5. Optimal subgraph mining

In order to mine optimal subgraph $g^\star$ on step 3 of Algorithm 1, we need to perform the subgraph enumeration procedure. In RLMD, we employ the frequent subgraph mining-based algorithm gSpan [42]. The key idea of gSpan is that each subgraph has a unique DFS Code, which is defined by a lexicographic order of the discovery time during the search process. By employing a depth first search strategy on the DFS Code tree (where each node is a subgraph), gSpan can enumerate all frequent subgraphs efficiently.

During the subgraph mining process, the search space is exponentially large, which requires an effective pruning scheme to reduce the search space. In this subsection, we will derive the upper-bound of the informative score for each subgraph, which helps prune the search space and speed up the subgraph mining.

Theorem 2. *Upper-bound Score: Let g and g' be two subgraph patterns, and $g\subseteq g'$, for the subgraph g, we define*

$$A_1(g)=2\sum_{\{i|y_i=+1,g\in G_i\}}\alpha_i$$
$$A_2(g)=2\sum_{\{i|y_i=-1,g\in G_i\}}\alpha_i$$
$$A_3=\sum_{t=1}^{n}\alpha_i y_i$$
$$\hat{\Theta}(g)=\begin{cases}\max\{|A_1(g)-A_3|,|A_2(g)|\}: & A_3\ge 0\\ \max\{|A_2(g)+A_3|,|A_1(g)|\}: & A_3<0\end{cases}$$

then $\Theta(g')\le\hat{\Theta}(g)$, where $\Theta(g')$ is defined in (12).

Proof 3. We start with the definition of $\Theta(g')$:

$$\Theta(g')=|\sum_{i=1}^{n}y_i\alpha_i\mathbf{x}_i'|$$
$$=|\sum_{i=1}^{n}y_i\alpha_i\cdot[2I(g'\subseteq G_i)-1]|$$

[2] http://www.MALSAR.org

$$
\begin{aligned}
&= |2\sum_{g' \subseteq G_i} y_i\alpha_i - \sum_{i=1}^{n} \alpha_i y_i| \\
&= |A_1(g') - A_2(g') - A_3| \\
&\le \begin{cases} \max\{|A_1(g') - A_3|, |A_2(g')|\}: & A_3 \ge 0 \\ \max\{|A_2(g') + A_3|, |A_1(g')|\}: & A_3 < 0 \end{cases} \\
&\le \begin{cases} \max\{|A_1(g) - A_3|, |A_2(g)|\}: & A_3 \ge 0 \\ \max\{|A_2(g) + A_3|, |A_1(g)|\}: & A_3 < 0 \end{cases} \\
&= \hat{\Theta}(g)
\end{aligned}
$$

The first inequality holds because for $\alpha_i < 0$, $A_1(g') \le 0$ and $A_2(g') \le 0$, so the upper-bound depends on A_3. If $A_3 \ge 0$, $A_1(g')$ and A_3 will have different signs, then the upper-bound is the maximum between $\{|A_1(g') - A_3|, |A_2(g')|\}$. The case is similar for $A_3 < 0$. The second inequality holds because $|A_1(g')| \le |A_1(g)|$ and $|A_2(g')| \le |A_2(g)|$ for $g \subseteq g'$.

Theorem 2 states that for any super graph of a subgraph g, its informative score is upper-bounded by $\hat{\Theta}(g)$. This rule can prune unpromising candidates effectively.

Algorithm 2. Optimal subgraph mining.

```
REQUIRE:
  ${(G_1, y_1), ..., (G_n, y_n)}$ : Training graphs;
  $\alpha_i$ : Weight for each graph example;
  $\mathcal{F}_1$: Already selected subgraph set;
ENSURE:
  $g^\star$:The optimal subgraph;
1: $\eta = 0$;
2: WHILE Recursively visit the DFS Code Tree in gSpan do
3:    $g_p \leftarrow$ current visited subgraph in DFS Code Tree;
4:    IF$g_p$ has been examined then
5:    continue;
6:    end iF
7: Compute scores $\Theta(g_p)$ for subgraph $g_p$ according (12);
8: iF $g_p \notin \mathcal{F}_1 \& \Theta(g_p) > \eta$ then
9: $\eta = \Theta(g_p)$;
10: $g^\star \leftarrow g_p$;
11: end IF
12: IF $\hat{\Theta}(g_p) > \eta$ then
13:    Depth-first search the subtree rooted from node $g_p$;
14: end if
15: end WHILE
16: return $g^\star$;
```

Optimal Subgraph Exploration Algorithm: Our optimal subgraph mining algorithm is listed in Algorithm 2. The minimum value η in the optimal set is initialized in step 1. Duplicated subgraph features are pruned in steps 4–5, and the informative score $\Theta(g_p)$ for g_p is calculated in step 7. If g_p is not selected before ($g_p \notin \mathcal{F}_1$) and $\Theta(g_p)$ is larger than η, we replace the optimal subgraph $g^\star$ with the current g_p and update the optimal score η (steps 8–11).

A branch-and-bound pruning rule, according to Theorem 2, is subsequently used to prune the search space on step 12. Lastly, the optimal subgraph $g^\star$ is returned in step 16.

The above pruning process is a key feature of our algorithm, because we do not require a support threshold value for subgraph mining (whereas all filter subgraph mining methods will require users to predefine a threshold value in order to discover subgraphs).

4.6. Relation to gBoost

Our RLMD subgraph selection algorithm advances the existing *column generation* style techniques employed in gBoost [23] for graph classification. The learning objective function for gBoost is

$$
\begin{aligned}
&\max_{\rho, \mathbf{w}, \xi} \quad \rho - \frac{1}{\nu n}\sum_{i=1}^{n} \xi_i \\
&s.t.\, y_i \sum_{k=1}^{m} \hbar_{g_k}(G_i) w_k + \xi_i \ge \rho; \\
&\textstyle\sum_{k=1}^{m} w_k = 1; \\
&w_k \ge 0, \xi_i \ge 0;
\end{aligned} \tag{13}
$$

From [43], we know that this formula is equivalent to the following linear programming:

$$
\begin{aligned}
&\min_{\mathbf{w}, \xi} \quad \sum_{k=1}^{m} w_k + C\sum_{i=1}^{n} \xi_i \\
&s.t. \quad y_i \sum_{k=1}^{m} \hbar_{g_k}(G_i) w_k + \xi_i \ge 1; \\
&w_k \ge 0, \xi_i \ge 0;
\end{aligned} \tag{14}
$$

Eq. (14) is actually a ℓ_1 SVM formulation, and can also be formulated as the regularized loss minimization formulation problem:

$$
\min \|\mathbf{w}\|_1 + C\sum_{i=1}^{n} \mathcal{L}_h(y_i, f(\mathbf{x}_i)) \tag{15}
$$

Here, $\mathcal{L}_h(y_i, f(\mathbf{x}_i)) = \max(1 - y_i f(\mathbf{x}_i), 0)$, which is known as hinge loss in machine learning.

Compared to our objective function in (4), we can find that gBoost Eq. (15) is a special case of Eq. (4), with the ℓ_2 regularization term being 0. Although the hinge loss function is non-differentiable, our subgradient method still applies, as long as $\partial\mathcal{C}/\partial w_k$ in (6) is properly defined. This observation shows the following advantages of our algorithm: (1) gBoost employs a hinge loss function which is similar to SVM and requires the problem to be formulated as a linear programming. Our algorithm generalizes and advances gBoost by removing the linear programming constraint and can employ any differentiable loss function, in addition to the logistic loss function considered in our paper. This generalization has great attractiveness in many applications, especially when the probability estimation for classification is required (the logistic function can provide some probabilistic information compared to the hinge loss function); (2) while gBoost employs ℓ_1 norm regularization to obtain a sparse solution, our algorithm considers an additional norm ℓ_2. This combined norm (known as *elastic net*) enables a sparse and more stable solution.

5. Experiment

5.1. Experimental settings

Benchmark Data: We validate the performance of the proposed algorithm on two types of graph classification datasets.

Anti-cancer activity prediction (NCI): The NCI graph collection[3] is a benchmark for predicting the biological activity of small molecules for different types of cancers. Each NCI dataset belongs to a bioassay task for anticancer activity prediction, such as Breast cancer or Leukemia cancer. Each molecule is represented as a graph, with atoms representing nodes and bonds denoting edges. A molecule is positive if it is active against a certain type of cancer,

[3] http://pubchem.ncbi.nlm.nih.gov

Table 1
Datasets used in experiments

ID	#Pos	#Total	Learning tasks
1	1793	37,349	Non-small cell lung
33	1467	37,022	Melanoma
41	1350	25,336	Prostate
47	1735	37,298	Central nerv sys
81	2081	37,549	Colon
83	1959	25,550	Breast
109	1773	37,518	Ovarian
123	2715	36,903	Leukemia
145	1641	37,043	Renal

or negative otherwise. Table 1 summarizes nine NCI graph classification tasks used in our experiments, where columns 2–4 denote the number of positive molecules, the total number of graphs, and the type of cancer of each dataset. In our experiments, we randomly select 1000 graphs from each dataset with balanced class distributions for graph classification.

Predictive Toxicology Challenge Dataset (PTC): The PTC challenge includes a number of carcinogenicity classifications for the toxicology prediction of chemical compounds.[4] The dataset we selected contains 417 compounds with four types of test animals: MM (male mouse), FM (female mouse), MR (male rat), and FR (female rat). Each compound has labels selected from {CE, SE, P, E, EE, IS, NE, N}. Similar to [44], we set {CE, SE, P} as positive labels, and {NE,N} as negative labels.

Baseline Methods: In our experiments, we consider three types of baseline methods, namely the two-step filter methods (TFMs), direct filter methods (DFMs), and embedded methods, as follows:

- *IG+SVM* is a TFM method that simply mines a set of frequent subgraphs, and then performs feature selection by using Information Gain. A SVM classifier is trained by using selected subgraph features for graph classification.
- *TOP+SVM* is similar to IG+SVM except that it selects the top K subgraphs based on their frequency rather than their information gain values.
- *gSemi+SVM* [11] is a DFM method, which integrates the feature selection into the subgraph mining process. The measurement for feature selection mainly considers the *must-link* and *cannot link* constraints between graph samples within the same or between different classes.
- *gHSIC+SVM* [12] is another DFM method which exploits the correlations between features and labels.
- *gBoost* *[23]* is a state-of-the-art embedded method which formulates the feature selection as a linear problem and selects subgraph features which best fit the objective function.
- *RLMD* is our proposed method which employ a logistic loss function together with an *elastic net* for regularization, and automatically determines optimal number of subgraphs K.

We conduct 10-fold cross-validation on all graph datasets and report the average results and *standard errors* of 10 folds in the final result. The parameters for γ_1 are selected from {0.005,0.01,0.03,0.05}, and γ_2 is selected from {0.01,0.03,0.05}. We will further analyze the impact of γ_1 and γ_2 in wider ranges in Section 5.2.4. For the filter methods (IG+SVM, TOP+SVM, gSemi+SVM, and gHSIC+SVM), the minimum support for frequent subgraph mining is set to 10% on NCI graph datasets and 1% on PTC classification tasks, and an SVM classifier is trained with C parameter from the range {0.1,1,10,100, 1000,10000}. For the gBoost algorithm, the parameter ν is selected from {0.1, 0.2, 0.3, 0.4}. Following [23], we select the best average results of 10-fold cross-validation for each baseline algorithm by varying these parameters, which represents the best performance each baseline can achieve.

For fairness of comparison, we increase the number of features to be selected for the filter methods (IG+SVM, TOP+SVM, gSemi+SVM, and gHSIC+SVM), and increase the iterations for the embedded methods (gBoost and RLMD), then collect and compare the performance of all algorithms under the same number of features. We set $S_{\max}=200$, which defines the maximum number of features used to learn the classifier models. Note that for RLMD, the algorithm may stop before reaching the maximum iterations/subgraphs we set, i.e., the optimal K is obtained. When RLMD stops, the optimal number of subgraph features has been discovered and RMLD will not add additional subgraphs to the feature set. We also compare RLMD under the optimal subgraph value to other baselines with the same number of K features (the purpose is to show that the optimal subgraph features discovered by RLMD are indeed optimal for graph classification).

5.2. Experimental results

5.2.1. Results on NCI graph dataset

For the NCI graph datasets, we vary the number of selected subgraph features from 20 to 200 for filter methods, and the number of iterations for gBoost and RLMD from 1 to 200. The accuracies and AUC values are shown in Fig. 3.

Comparison with Filter methods: The results in Fig. 3 show that with the increase in the number of features/iterations, the filter methods (TOP+SVM, IG+SVM, gSemi+SVM, and gHSIC+SVM) are inferior to RLMD. This is because filter methods separate the feature selection module from the model learning process. The subgraph features selected from filter methods may not fit the underlying learning model very well (we use SVM in our experiments). This is actually an observed common drawback of filter methods [19]. The performance among these filter-based methods varies from one graph dataset to another, and none of them significantly outperforms others. For instance, gSemi+SVM outperforms TOP+SVM, IG+SVM, and gHSIC+SVM on NCI-1 (Fig. 3(A)) when the number of selected graphs is considerably large (≥ 160), but is worse than gHSIC on NCI-109 (Fig. 3(G)). This may be attributed to the inherent differences underneath the graph datasets.

How many subgraphs to select: Another drawback of filter methods, shown in our experiments, is that the performance of filter methods varies significantly *w.r.t.* different numbers of selected features (K). Indeed, all these filter methods only select subgraph features with maximum discriminative score regardless of the redundancy among the features. Adding redundant features may decrease the performance of an algorithm. Further analysis of subgraph features is presented with a case study in the next subsection.

In contrast, for embedded methods, the above drawbacks can be handled effectively. Our algorithm RLMD unifies the feature selection and model learning into a whole framework, so that the feature selection process is driven by the well-defined objective function, and the selected features can further enhance the learning models. At the same time, RLMD is guaranteed to be convergent given an appropriate γ_1 value, which means that we do not need to specify the total number of selected graphs K. For instance, in Fig. 3(A), RLMD reaches convergence with 180 features.

RLMD vs. gBoost: It is evident that RLMD outperforms gBoost for most datasets. This is mainly because gBoost only uses ℓ_1-norm

[4] http://www.predictive-toxicology.org/ptc/

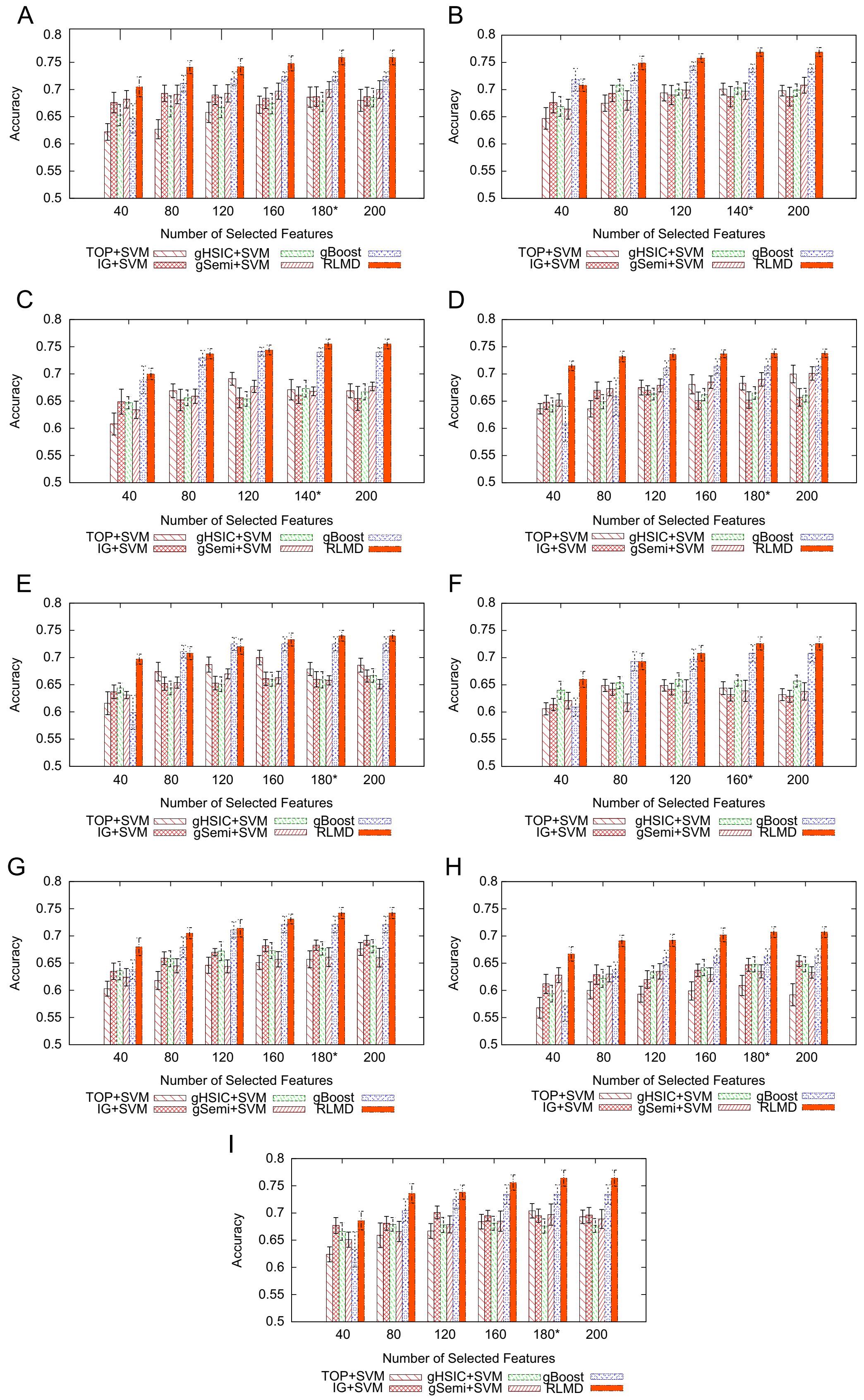


Fig. 3. The classification accuracy and standard error on NCI dataset *w.r.t.* the number of selected graphs. The optimal number of subgraphs selected by RLMD (once it converges for all 10 folds experiments) is marked by a star $*$ at x-axis. Sub figures A-I: NCI 1, 33, 41, 47, 81, 83, 109, 123, and 145.

Table 2
Averaged accuracies and standard errors on NCI graphs with optimal *K*.

ID	RLMD	TOP+SVM	IG+SVM	gHSIC+SVM	gSemi+SVM	gBoost
1	**0.759** $\pm$ 0.014	0.686 $\pm$ 0.019	0.687 $\pm$ 0.018	0.677 $\pm$ 0.018	0.700 $\pm$ 0.015	0.724 $\pm$ 0.009
33	**0.769** $\pm$ 0.008	0.701 $\pm$ 0.011	0.687 $\pm$ 0.019	0.703 $\pm$ 0.012	0.697 $\pm$ 0.015	0.739 $\pm$ 0.008
41	**0.755** $\pm$ 0.009	0.677 $\pm$ 0.017	0.649 $\pm$ 0.015	0.672 $\pm$ 0.015	0.670 $\pm$ 0.009	0.740 $\pm$ 0.008
47	**0.738** $\pm$ 0.009	0.683 $\pm$ 0.013	0.652 $\pm$ 0.015	0.665 $\pm$ 0.012	0.690 $\pm$ 0.012	0.714 $\pm$ 0.014
81	**0.740** $\pm$ 0.010	0.679 $\pm$ 0.012	0.660 $\pm$ 0.014	0.659 $\pm$ 0.015	0.658 $\pm$ 0.009	0.725 $\pm$ 0.013
83	**0.726** $\pm$ 0.012	0.644 $\pm$ 0.012	0.632 $\pm$ 0.012	0.658 $\pm$ 0.011	0.639 $\pm$ 0.020	0.708 $\pm$ 0.016
109	**0.742** $\pm$ 0.010	0.657 $\pm$ 0.016	0.683 $\pm$ 0.009	0.677 $\pm$ 0.012	0.661 $\pm$ 0.017	0.721 $\pm$ 0.015
123	**0.707** $\pm$ 0.010	0.609 $\pm$ 0.019	0.647 $\pm$ 0.012	0.648 $\pm$ 0.014	0.635 $\pm$ 0.012	0.663 $\pm$ 0.013
145	**0.764** $\pm$ 0.015	0.704 $\pm$ 0.013	0.695 $\pm$ 0.012	0.676 $\pm$ 0.013	0.697 $\pm$ 0.020	0.734 $\pm$ 0.018

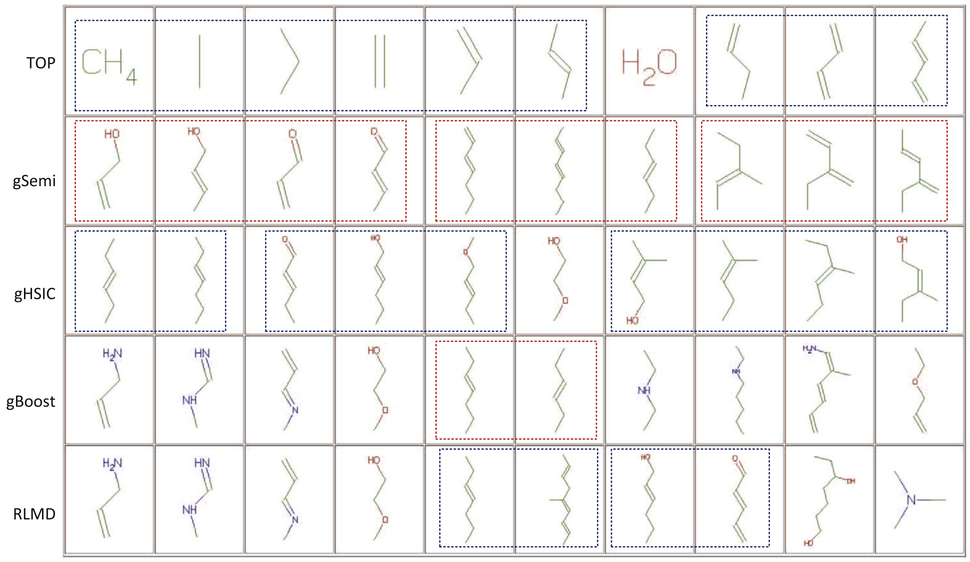


Fig. 4. Case study: comparison of discriminative subgraph features discovered by different algorithms. Subgraph features with high similarities are grouped and highlighted in the dashed rectangles. Subgraphs mined by filter methods are similar to each other and share high redundancy.

Table 3
Accuracies and standard errors on PTC graphs with optimal *K* for 10-fold cross-validation.

ID	RLMD	TOP+SVM	IG+SVM	gHSIC+SVM	gSemi+SVM	gBoost
MR	**0.655** $\pm$ 0.013	0.606 $\pm$ 0.024	0.607 $\pm$ 0.025	0.596 $\pm$ 0.019	0.601 $\pm$ 0.020	0.608 $\pm$ 0.027
MM	**0.664** $\pm$ 0.019	0.060 $\pm$ 0.025	0.599 $\pm$ 0.023	0.613 $\pm$ 0.021	0.603 $\pm$ 0.019	0.622 $\pm$ 0.018
FR	**0.704** $\pm$ 0.018	0.607 $\pm$ 0.029	0.584 $\pm$ 0.029	0.635 $\pm$ 0.026	0.619 $\pm$ 0.023	0.678 $\pm$ 0.008
FM	**0.615** $\pm$ 0.018	0.592 $\pm$ 0.031	0.594 $\pm$ 0.026	0.581 $\pm$ 0.021	0.575 $\pm$ 0.018	0.603 $\pm$ 0.017

Table 4
AUC values and standard errors on PTC graphs with optimal *K* for 10-fold cross-validation.

ID	RLMD	TOP+SVM	IG+SVM	gHSIC+SVM	gSemi+SVM	gBoost
MR	**0.681** $\pm$ 0.021	0.597 $\pm$ 0.025	0.596 $\pm$ 0.025	0.560 $\pm$ 0.021	0.580 $\pm$ 0.021	0.649 $\pm$ 0.033
MM	**0.680** $\pm$ 0.020	0.593 $\pm$ 0.025	0.590 $\pm$ 0.025	0.583 $\pm$ 0.024	0.600 $\pm$ 0.022	0.600 $\pm$ 0.034
FR	**0.673** $\pm$ 0.022	0.600 $\pm$ 0.029	0.575 $\pm$ 0.029	0.618 $\pm$ 0.021	0.623 $\pm$ 0.021	0.640 $\pm$ 0.026
FM	**0.614** $\pm$ 0.013	0.585 $\pm$ 0.032	0.587 $\pm$ 0.024	0.582 $\pm$ 0.021	0.580 $\pm$ 0.017	0.583 $\pm$ 0.017

regularization to produce a sparse solution. As pointed out in [21], the lasso (ℓ_1-norm) has several drawbacks: (1) when the number of features (m, which is exponentially huge) is much bigger than the number of observations (n), the ℓ_1 norm selects at most n variables before it saturates; and (2) when the pairwise correlations in a group of variables are very high, lasso tends to select only one variable from the group and does not discriminate which one it selects. In contrast, RLMD uses an elastic net (combination of ℓ_1 and ℓ_2 norm), which encourages a grouping effect, where strongly correlated features will be included/excluded. As a result, RLMD results in a similar sparsity of representation to gBoost, but often outperforms gBoost.

Overall Performance with Optimal K: In Table 2, we summarize the performance of our algorithm under optimal K value with other methods, where filter methods use the same number of subgraphs (K) for graph classification, and gBoost runs until convergence. The result in Table 2 clearly demonstrates that RLMD outperforms two-step filter methods (TOP+SVM and IG+SVM), direct filter methods (gHSIC+SVM and gSemi+SVM), and gBoost algorithm in NCI datasets.

5.2.2. Case study: subgraph feature comparison

In this subsection, we use NCI-1 dataset as a case study to investigate subgraphs discovered by different algorithms. In our experiments, the top-10 subgraph features are discovered and illustrated in Fig. 4.

It is evident that the features for all filter methods (TOP, gSemi, and gHSIC) share high correlations. For the gSemi algorithm, for instance, the top-10 subgraphs form 3 groups. In each group, the subgraph features are very similar to each other. This is because the subgraph mining algorithm follows the depth-first-search (DFS) scheme, and subgraphs from the same sub-tree are very close to each other in terms of their geometrical structure. Because these methods consider each subgraph independently, the selected subgraphs may have high redundancy, which imposes a great challenge in determining the optimal K subgraphs for graph classification and also causes fluctuating results when the K values are varied.

In contrast, the subgraph correlations for gBoost and RLMD are much smaller. The subgraphs discovered by gBoost and RLMD are highly overlapping (the first 5 subgraphs are identical). As pointed out by [21], ℓ_1 regularization tends to select only one subgraph from a group of features and is not selective about which one is included, thus the redundancy among the features in gBoost is minimal. By using elastic norm, RLMD retains several group effects (some discriminative features may be included and excluded simultaneously), and usually achieves better results. This result is consistent with observations reported in [21] for vector data.

5.2.3. Results on PTC tasks

We also conducted extensive experiments on the PTC datasets. The accuracies and AUC values (i.e., the area under ROC curves) are reported in Tables 3 and 4, where the results are obtained after RLMD converges, and K=200 for all filter methods.

The results in Tables 3 and 4 show that RLMD achieves considerable performance gains over all filter methods (TFM and DFM) and gBoost algorithm for all PTC datasets. Note that for PTC classifications, AUC values are more important because they are all imbalanced classification tasks.

Table 5
Impact of different γ_1 values on NCI-1 dataset with $\gamma_2 = 0.03, S_{max} = 200$.

γ_1	0	0.01	0.03	0.05	0.15
#Selected subgraphs	200	180	130	65	0
Accuracy	0.775	0.759	0.711	0.679	0.5
AUC	0.831	0.811	0.795	0.713	0.5

5.2.4. Parameter analysis

In this subsection, we study the impact of parameters γ_1 and γ_2 on algorithm performance. Both γ_1 and γ_2 values are selected from {0, 0.01, 0.03, 0.05, 0.07, 0.09, 0.11, 0.15, 0.2}, and the results under 10-fold cross-validation on NCI-1 and NCI-33 are shown in Fig. 5.

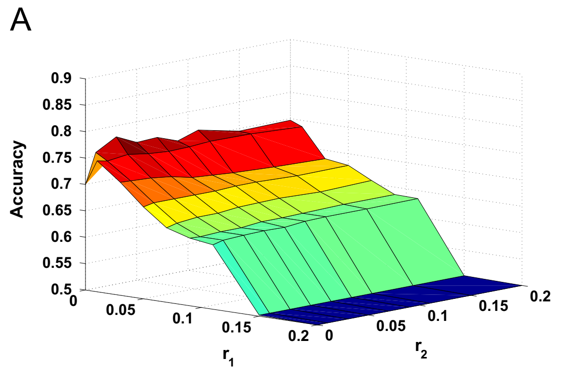
Impact of γ_1 values: The experimental results in Fig. 5 show that γ_1 plays a more important role for the final classification model. With the increase of γ_1 from 0 to 0.2, the classification performance drops rapidly in terms of accuracy.

To better understand the impact of γ_1, we also summarize the number of subgraphs selected with different γ_1 values in Table 5. The results show that increasing γ_1 values will result in fewer subgraphs being selected for the final classifier model, because a larger ℓ_1 norm regularizes more elements as 0. For $\gamma_1 = 0$, there is no sparse solution. In other words, every subgraph should be used for graph classification. In this case, RLMD will only terminate when all subgraphs are incorporated for learning the model, or the maximum number of iterations S_{max} is reached. As the subgraph space is exponentially large, it is impractical to use all subgraph features to learn the model. The algorithm relies on $S_{\max}$ to terminate (200 is set in our experiment). The result shows that γ_1=0 even achieves better classification result, which is attributed to the fact that although ℓ_1 regularization introduces a sparse solution, it may be biased in some applications [45], so the accuracy may drop. For other cases with γ_1 being considerably large ($\gamma_1 = 0.5$), the regularization term dominates the objective function (4) with no subgraph being used for classification, which results in poor classification accuracy.

Note that the convergence property of our algorithm is dependent on γ_1. In our experiments, we notice that γ_1 is very easy to set (in a small range [0.01,0.03]) for obtaining satisfactory results. This is much easier than requiring users to specify the number of subgraph features K needed for each graph dataset, because users may not have any prior knowledge about the selection of K values for different datasets, and different K values often result in significant changes in the algorithm performance.

Interplay between γ_1 and subgraph selection: We further compare the common subgraphs selected by different γ_1 values, and report the results in Fig. 6. The results show that the subgraphs selected by using a smaller γ_1 values contain many subgraphs which are selected by using a larger γ_1 value. This observation is further evident in Fig. 6(E) and (F). The reason is that a smaller γ_1 value will result in more subgraph features to be selected, which increases the possibility of covering a small subgraph set selected by using a larger γ_1 value. In other words, a slightly smaller γ_1 value will result in more subgraph feature candidates to be explored and be beneficial for the classification task.

Impact of γ_2 values: We also vary γ_2 from 0 to 0.2, and report the results in Table 6. The results show that a small regularization value $\gamma_2 = 0.03$ outperforms the case of $\gamma_2 = 0$, where the ℓ_2



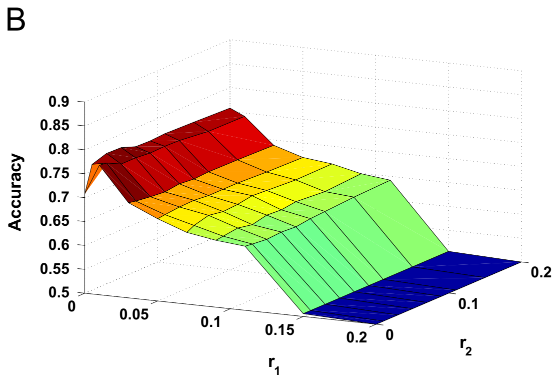


Fig. 5. The accuracies with different γ_1 and γ_2 values (A) NCI 1, and (B) NCI 33.

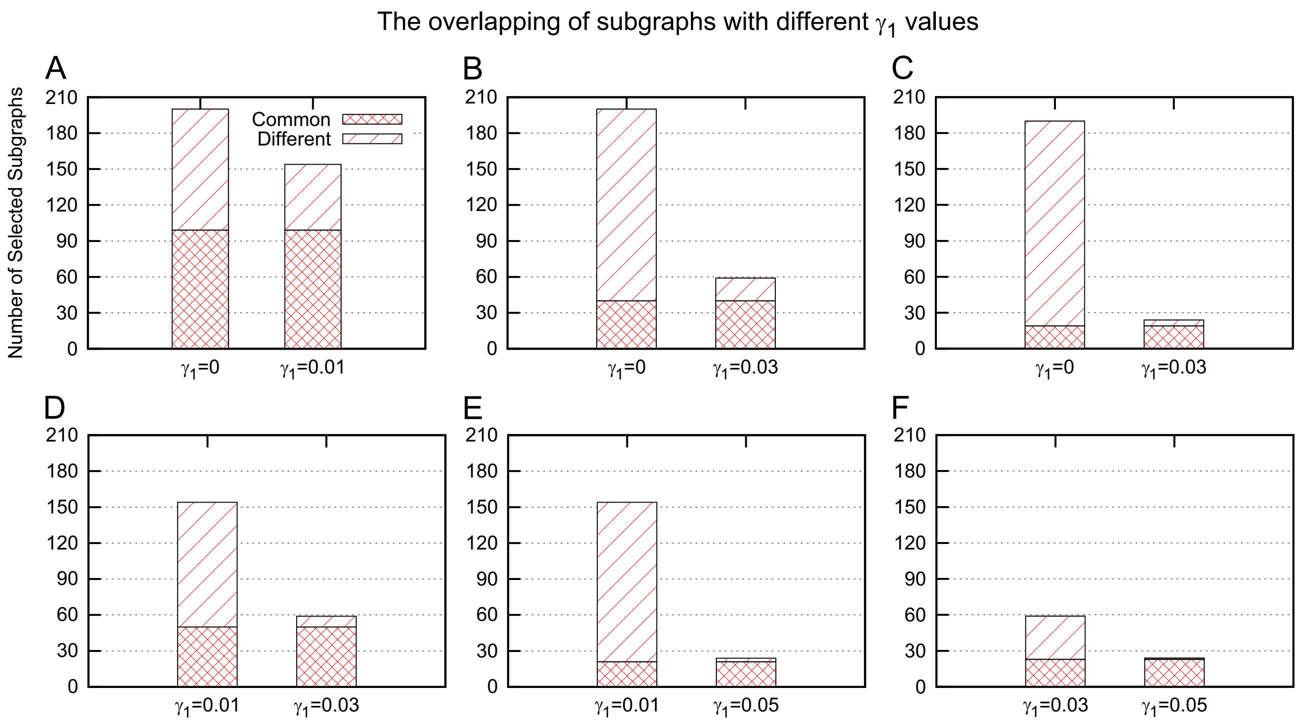


Fig. 6. The overlapping of subgraphs (common subgraphs *vs.* different subgraphs) for different γ_1 values on NCI-1 dataset with a 70–30% splitting on the NCI-1 dataset, i.e., 70% graphs are randomly selected as training graphs, and 30% are used as test graphs. $\gamma_2 = 0.03$ and $S_{max} = 200$. (A) γ1=0 v.s. γ1=0.01, (B) γ1=0 v.s. γ1=0.03, (C) γ1=0 v.s. γ1=0.05, (D) γ1=0.01 v.s. γ1=0.03, (E) γ1=0.01 v.s. γ1=0.05, and (F) γ1=0.03 v.s. γ1=0.05.

Table 6
Impact of different γ_2 values on NCI-1 dataset with $\gamma_1 = 0.01$.

γ_2	0	0.03	0.07	0.15	0.2
Accuracy	0.748	0.759	0.74	0.741	0.72
AUC	0.810	0.811	0.806	0.811	0.78

Table 7
Impacts of different ϵ values on NCI-1 dataset.

ϵ	0.01	0.005	0.001	0.0001
Accuracy	0.748	0.759	0.761	0.760
AUC	0.803	0.811	0.814	0.812

regularization effect disappears (only ℓ_1 is used). This result is consistent with observations from a previous study [21]. This may be because ℓ_1 ignores the correlated subgraphs in a group of features. When γ_2 keeps increasing, the classification performance drops because the larger ℓ_2 regularization dominates the objective function and the loss minimization term has less effect.

Impact of ϵ values: We vary the ϵ values from 0.01 to 0.0001 to study the final classification performance of our algorithm, and report the final classification results in Table 7. The results show that as long as ϵ is subtle (from 0.001 to 0.0001), our classification can achieve similar classification results, i.e., an ϵ-tolerance accuracy result to the optimal solution.

6. Conclusion

In this paper, we proposed a regularized loss minimization-driven (RLMD) graph classification method. We argued that existing filter-based subgraph selection methods simply focus on finding most discriminative subgraph features, and suffer severe disadvantages in determining the optimal number of subgraphs for graph classification and separating feature selection from the model learning phase. As a result, they might be able to find most discriminative subgraph features, but cannot form high accuracy classifiers because they cannot determine how many discriminative features are needed to train classifiers with the best performance gain. By integrating subgraph mining, discriminative subgraph selection, and model learning into one unified framework, RLMD is able to automatically determine the optimal number of discriminative subgraphs for best graph classification results. Our algorithm generalizes the state-of-the-art gBoost algorithm in the sense that it can employ any differentiable loss function and achieve better classification accuracy by using an elastic net regularization. Experimental results on real-world graph datasets show a clear performance gain over existing two-step filter methods (TFMs), direct filter methods (DFMs), and embedding methods.

Conflict of interest

None declared.

Acknowledgments

We thank anonymous reviewers for their constructive comments. This work is partially supported by Australia ARC Discovery Project (DP140102206), and National Scholarship for Building High Level Universities, China Scholarship Council (No. 2011630030).

References

[1] M. Deshpande, M. Kuramochi, N. Wale, G. Karypis, Frequent substructure-based approaches for classifying chemical compounds, IEEE Trans. Knowl. Data Eng. (2005) 1036–1050.

[2] J. Vogelstein, W. Roncal, R. Vogelstein, C. Priebe, Graph classification using signal-subgraphs: applications in statistical connectomics, IEEE Trans. Pattern Anal. Mach. Intell. 35 (7) (2013) 1539–1551.
[3] A. Morales-Gonzlez, N. Acosta-Mendoza, A. Gago-Alonso, E.B. Garca-Reyes J.E. Medina-Pagola, A new proposal for graph-based image classification using frequent approximate subgraphs, Pattern Recognit. 47 (1) (2014) 169–177.
[4] S. Pan, X. Zhu, C. Zhang, P.S. Yu, Graph stream classification using labeled and unlabeled graphs, in: 2013 IEEE 29th International Conference on Data Engineering (ICDE), IEEE, 2013, pp. 398–409.
[5] F. Wysotzki, W. Kolbe, J. Selbig, Concept learning by structured examples—an algebraic approach, Int. Joint Conf. Artif. Intell. (1981) 153–158.
[6] P. Geibel, F. Wysotzki, Learning relational concepts with decision trees, in: International Conference on Machine Learning, vol. 96, 1996, pp. 166–174.
[7] B.J. Jain, K. Obermayer, Structure spaces, J. Mach. Learn. Res. 10 (2009) 2667–2714.
[8] B.J. Jain, K. Obermayer, Learning in Riemannian orbifolds. arXiv:1204.4294 [physics.ins-det] (2012).
[9] H. Kashima, K. Tsuda, A. Inokuchi, Kernels for graphs, in: B. Schlkopf, K. Tsuda, J.P. Vert (Eds.), Kernel Methods in Computational Biology, MIT Press, Cambridge, MA, 2004, pp. 101–113.
[10] H. Fei, J. Huan, Boosting with structure information in the functional space: an application to graph classification, in: ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), Washington DC, USA, 2010.
[11] X. Kong, P. Yu, Semi-supervised feature selection for graph classification, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2010, pp. 793–802.
[12] X. Kong, P.S. Yu, Multi-label feature selection for graph classification, in: IEEE International Conference on Data Mining, IEEE, 2010, pp. 274–283.
[13] Y. Zhu, J. Yu, H. Cheng, L. Qin, Graph classification: a diversified discriminative feature selection approach, in: Conference on Information and Knowledge Management (CIKM), ACM, 2012, pp. 205–214.
[14] N. Jin, C. Young, W. Wang, Graph classification based on pattern co-occurrence, in: Conference on Information and Knowledge Management (CIKM), 2009, pp. 573–582.
[15] M. Thoma, H. Cheng, A. Gretton, J. Han, H. Kriegel, A. Smola, L. Song, P. Yu, X. Yan, K. Borgwardt, Near-optimal supervised feature selection among frequent subgraphs, in: SIAM International Conference on Data Mining, USA, 2009.
[16] S. Ranu, A.K. Singh, GraphSig: a scalable approach to mining significant subgraphs in large graph databases, in: International Conference on Data Engineering (ICDE), IEEE, 2009, pp. 844–855.
[17] X. Yan, H. Cheng, J. Han, P. S. Yu, Mining significant graph patterns by leap search, in: ACM SIGMOD Conference, ACM, 2008, pp. 433–444.
[18] H. Cheng, X. Yan, J. Han, Discriminative frequent pattern-based graph classification, Link Min. Models Algorithms Appl. (2010) 237–262.
[19] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, J. Mach. Learn. Res. 3 (2003) 1157–1182.
[20] R. Tibshirani, Regression shrinkage and selection via the lasso, J. R. Stat. Soc. Ser. B (Methodol.) (1996) 267–288.
[21] H. Zou, T. Hastie, Regularization and variable selection via the elastic net, J. R. Stat. Soc.: Ser. B (Stat. Methodol.) 67 (2) (2005) 301–320.
[22] T. Zhou, D. Tao, X. Wu, Manifold elastic net: a unified framework for sparse dimension reduction, Data Min. Knowl. Discov. 22 (3) (2011) 340–371.
[23] H. Saigo, S. Nowozin, T. Kadowaki, T. Kudo, K. Tsuda, gBoost: a mathematical programming approach to graph classification and regression, Mach. Learn. 75 (2009) 69–89.
[24] M. Culp, G. Michailidis, Graph-based semisupervised learning, IEEE Trans. Pattern Anal. Mach. Intell. 30 (1) (2008) 174–179.
[25] A. Mantrach, N. van Zeebroeck, P. Francq, M. Shimbo, H. Bersini, M. Saerens, Semi-supervised classification and betweenness computation on large, sparse, directed graphs, Pattern Recognit. 44 (6) (2011) 1212–1224.
[26] M. Karasuyama, H. Mamitsuka, Multiple graph label propagation by sparse integration, IEEE Trans. Neural Netw. Learn. Syst. 24 (12) (2013) 1999–2012.
[27] X. Liu, L. Wang, J. Zhang, J. Yin, H. Liu, Global and local structure preservation for feature selection, IEEE Trans. Neural Netw. Learn. Syst. 25 (6) (2014) 1083–1095.
[28] D. Liben-Nowell, J. Kleinberg, The link-prediction problem for social networks, J. Am. Soc. Inf. Sci. Technol. 58 (7) (2007) 1019–1031.
[29] C. Nguyen, H. Mamitsuka, Latent feature kernels for link prediction on sparse graphs, IEEE Trans. Neural Netw. Learn. Syst. 23 (11) (2012) 1793–1804.
[30] N. Jin, C. Young, W. Wang, GAIAgraph classification using evolutionary computation, in: ACM SIGMOD Conference, ACM, 2010, pp. 879–890.
[31] J. Wu, X. Zhu, C. Zhang, P. Yu, Bag constrained structure pattern mining for multi-graph classification, IEEE Trans. Knowl. Data Eng. 26 (10) (2014) 2382–2396.
[32] L. Bai, L. Rossi, A. Torsello, E.R. Hancock, A quantum Jensen–Shannon graph kernel for unattributed graphs, Pattern Recognit. 48 (2) (2015) 344–355.
[33] N. Shervashidze, P. Schweitzer, E.J. van Leeuwen, K. Mehlhorn K.M. Borgwardt, Weisfeiler–Lehman graph kernels, J. Mach. Learn. Res. 12 (2011) 2539–2561.
[34] M. Neumann, N. Patricia, R. Garnett, K. Kersting, Efficient graph kernels by randomization, in: Machine Learning and Knowledge Discovery in Databases, Springer, 2012, pp. 378–393.
[35] K. Riesen, H. Bunke, Graph classification by means of Lipschitz embedding, IEEE Trans. SMC Part B: Cybern. 39 (2009) 1472–1483.
[36] J. Wu, Z. Hong, S. Pan, X. Zhu, C. Zhang, Multi-graph-view learning for graph classification, in: IEEE International Conference on Data Mining, 2014, pp. 590–599.
[37] J. Wu, S. Pan, X. Zhu, Z. Cai, Boosting for multi-graph classification, IEEE Trans. Cybern. 45 (3) (2015) 430–443.
[38] J. Wu, Z. Hong, S. Pan, X. Zhu, C. Zhang, Z. Cai, Multi-graph learning with positive and unlabeled bags, in: SIAM International Conference on Data Mining, SIAM, 2014, pp. 217–225.
[39] S. Pan, X. Zhu, Graph classification with imbalanced class distributions and noise, Int. Joint Conf. Artif. Intell. (2013) 1586–1592.
[40] S. Pan, J. Wu, X. Zhu, C. Zhang, Graph ensemble boosting for imbalanced noisy graph stream classification, IEEE Trans. Cybern. 45 (5) (2015) 940–954.
[41] S. Pan, J. Wu, X. Zhu, Cogboost: Boosting for fast cost-sensitive graph classification, IEEE Trans. Knowl. Data Eng. 1 (2015), http://dx.doi.org/10.1109/TKDE.2015.2391115, in press.
[42] X. Yan, J. Han, gSpan: graph-based substructure pattern mining, in: IEEE International Conference on Data Mining, 2002, pp. 721–724.
[43] A. Demiriz, K. Bennett, J. Shawe-Taylor, Linear programming boosting via column generation, Mach. Learn. (2002) 225–254.
[44] T. Kudo, E. Maeda, Y. Matsumoto, An application of boosting to graph classification, Neural Inf. Process. Syst. (2004) 729–736.
[45] M. Tan, I.W. Tsang, L. Wang, Towards ultrahigh dimensional feature selection for big data, J.Mach. Learn. Res. 15 (2014) 1371–1429.

Shirui Pan received his master degree in computer science from Northwest A&F University, Yangling, Shaanxi, China, in 2011. Since September 2011, he has been working toward the Ph.D. degree in the Centre for Quantum Computation and Intelligent Systems (QCIS), Faculty of Engineering and Information Technology, University of Technology, Sydney (UTS). His research focuses on machine learning and data mining.

Jia Wu received his bachelor degree in computer science from China University of Geosciences (CUG), Wuhan, China, in 2009. Since September 2009, he has been working toward the Ph.D. degree under the Master–Doctor combined program in computer science from CUG. Besides, he is also pursuing a Ph.D. degree in QCIS Centre, Faculty of Engineering and Information Technology, University of Technology, Sydney (UTS), Australia. His research focuses on data mining and machine learning.

Xingquan Zhu received the Ph.D. degree in computer science from Fudan University, Shanghai, China.

He is an associate professor in the Department of Computer & Electrical Engineering and Computer Science, Florida Atlantic University. Prior to that, he was with the Centre for Quantum Computation & Intelligent Systems, University of Technology, Sydney, Australia. His research interests mainly include data mining, machine learning, and multimedia systems. Since 2000, he has published more than 170 refereed journals and conference papers in these areas, including two Best Paper Awards and one Best Student Paper Award.

Dr. Zhu was an associate editor of the IEEE Transactions on Knowledge and Data Engineering (2014-date), and is currently serving on the Editor Board of International Journal of Social Network Analysis and Mining SNAM (2010-date) and Network Modeling Analysis in Health Informatics and Bioinformatics Journal (2014-date).

Guodong Long received his Ph.D. degree in computer science from University of Technology, Sydney (UTS), Australia, in 2014. He is a researcher associate and core member in the Centre for Quantum Computation and Intelligent Systems (QCIS), Faculty of Engineering and Information Technology, University of Technology, Sydney (UTS). His research focuses on machine learning, data mining and cloud computing.

Chengqi Zhang received the Ph.D. degree from the University of Queensland, Brisbane, Australia, in 1991 and the D.Sc. degree (higher doctorate) from Deakin University, Geelong, Australia, in 2002.

Since December 2001, he has been a Professor of Information Technology with the University of Technology, Sydney (UTS), Sydney, Australia, where he has been the Director of the UTS Priority Investment Research Centre for Quantum Computation and Intelligent Systems since April 2008. Since November 2005, he has been the Chairman of the Australian Computer Society National Committee for Artificial Intelligence. He has published more than 200 research papers, including several in first-class international journals, such as the Artificial Intelligence, IEEE, and ACM Transactions. He has published six monographs and edited 16 books, and has attracted 11 Australian Research Council grants. His research interests mainly focus on data mining and its applications.

He has been serving as an Associate Editor for three international journals, including IEEE Transactions on Knowledge and Data Engineering (2005–2008); and he served as General Chair, PC Chair, or Organising Chair for five international Conferences including ICDM 2010 and WI/IAT 2008. He is General Co-Chair of KDD 2015 in Sydney and the Local Arrangements Chair of IJCAI-2017 in Melbourne, and a Fellow of the Australian Computer Society and a Senior Member of the IEEE.