Bag Constrained Structure Pattern Mining for Multi-Graph Classification

Jia Wu, *Student Member, IEEE*, Xingquan Zhu, *Senior Member, IEEE*, Chengqi Zhang, *Senior Member, IEEE*, and Philip S. Yu, *Fellow, IEEE*

Abstract—This paper formulates a multi-graph learning task. In our problem setting, a bag contains a number of graphs and a class label. A bag is labeled positive if at least one graph in the bag is positive, and negative otherwise. In addition, the genuine label of each graph in a positive bag is unknown, and all graphs in a negative bag are negative. The aim of multi-graph learning is to build a learning model from a number of labeled training bags to predict previously unseen test bags with maximum accuracy. This problem setting is essentially different from existing multi-instance learning (MIL), where instances in MIL share well-defined feature values, but no features are available to represent graphs in a multi-graph bag. To solve the problem, we propose a Multi-Graph Feature based Learning (*gMGFL*) algorithm that explores and selects a set of discriminative subgraphs as features to transfer each bag into a single instance, with the bag label being propagated to the transferred instance. As a result, the multi-graph bags form a labeled training instance set, so generic learning algorithms, such as decision trees, can be used to derive learning models for multi-graph classification. Experiments and comparisons on real-world multi-graph tasks demonstrate the algorithm performance.

Index Terms—Graph classification, multi-instance learning, multi-graph, subgraph features

1 Introduction

Graphs and dependency structures commonly exist in real-world applications [36]. For example, an online webpage may consist of text descriptions, images, and videos, where text can be represented as graphs to preserve the content and contextual information (this representation has demonstrated a better performance than simple bag of words representation) [1]. In addition, each image can also be transferred into graphs to represent image regions and their structure dependencies (this representation is also better than simply representing the whole image using visual features such as color histograms or textures) [13]. As a result, a webpage can be regarded as a *bag* that contains a number of graphs, each representing a portion of the webpage content. For each viewer, a webpage is interesting if one, or multiple parts of the content (text or image) are interesting to him/her (i.e., A bag is positive if one, or multiple graphs inside the bag are positive). The webpage is not interesting (negative) to the viewer if none of the content falls within the viewer's interests (i.e., A bag is negative if all graphs inside the bag are negative). A conceptual view of multi-graph representation for a webpage is shown in Fig. 1.

The above multi-graph representation can also be generalized to many real-world domains, such as

- *Bio-pharmaceutical activity test.* Labeling individual molecules (which are commonly represented as graphs) is expensive and time-consuming. To reduce labeling costs, molecular group activity prediction can be used to investigate the activity of a group (bag) of molecules. Detailed investigations, on individual graph, are carried out only for each active group (i.e., a positive bag).
- *Online product recommendation based on review.* Each online product may receive many customer reviews. For each review composed of detailed text descriptions, we can use a graph to represent the review text. As a result, a product can be represented as a bag of graphs. Assume customers mainly concern about several key properties, such as "affordability" and "durability", of the product. A product (i.e., a bag) can be labeled as positive if it receives very positive review in any of these properties, and negative otherwise.
- *Scientific publication categorization.* A scientific publication can be represented as a graph by using correlations of keywords in the paper. Meanwhile, each paper also has a number of references, each of which can be represented as a graph. So each paper and all references cited in the paper form a bag of graphs. A bag can be labeled as positive if the paper or any of its references is relevant to a specific topic.

- *J. Wu is with the Centre for Quantum Computation and Intelligent Systems, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, New South Wales, Australia, and the Department of Computer Science, China University of Geosciences, Wuhan, P.R. China. E-mail: jia.wu@student.uts.edu.au.*
- *X. Zhu is with the Department of Computer & Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, FL. E-mail: xzhu3@fau.edu.*
- *C. Zhang is with the Centre for Quantum Computation and Intelligent Systems, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, New South Wales, Australia. Email: chengqi.zhang@uts.edu.au.*
- *P.S. Yu is with the Department of Computer Science, University of Illinois at Chicago, IL 60607. E-mail: psyu@cs.uic.edu.*

Manuscript received 16 Aug. 2013; revised 2 Dec. 2013; accepted 11 Dec. 2013. Date of publication 8 Jan. 2014; date of current version 29 Aug. 2014. Recommended for acceptance by A. Zhang.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2013.2297923

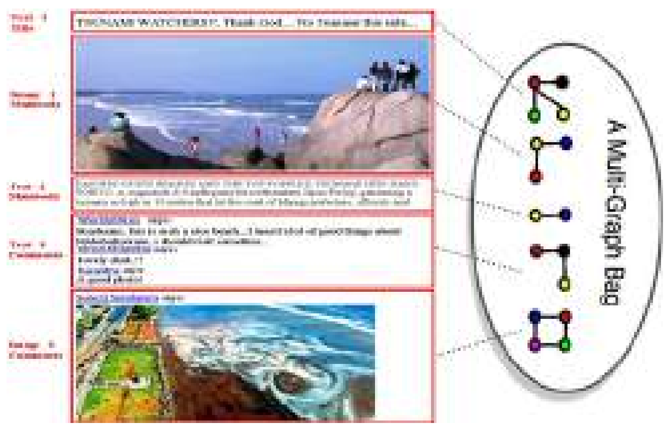


Fig. 1. An example of multi-graph representation for a webpage in Flickr online photo sharing system (http://www.flickr.com/). Each rectangular box represents one type of information source in the current page. Although the current page is marked as "Tsunami watchers," Image 1 and Text 3 are not relevant to the Tsunami, but Text 1, 2 and Image 2 are relevant to it. By converting content in each box into a graph, the whole webpage can be represented as a bag of graphs (i.e., a multi-graph representation).

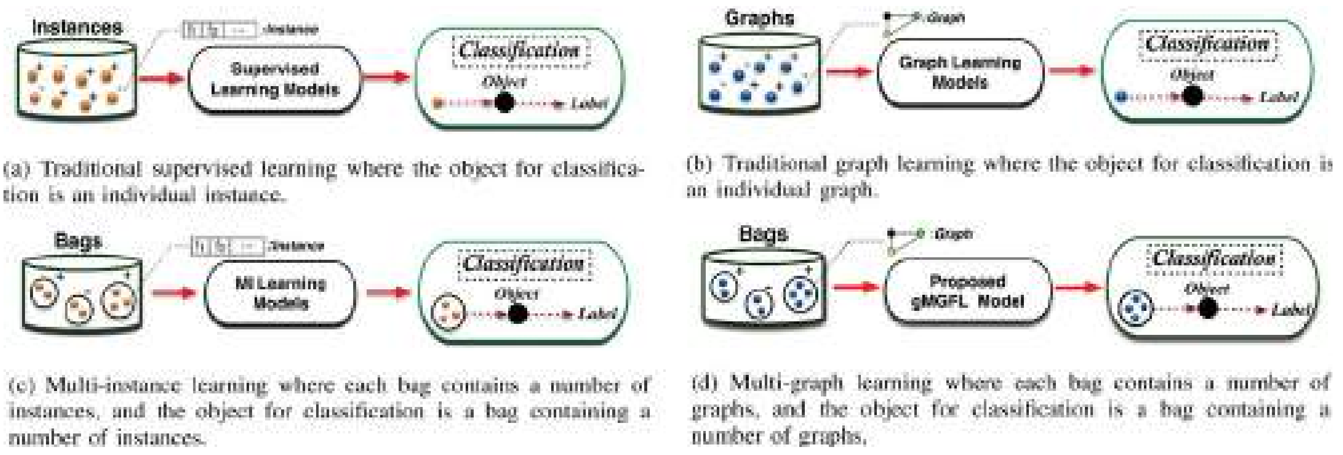
For the above three real-world applications, a bag-of-graph representation can be used to represent the object (e.g., molecular group, online product or scientific publication) as a bag containing many graphs (i.e., molecules, online product reviews or scientific publications). The details of these multi-graph examples are explained in Figs. 5 and 7.

The inherent advantage of the bag-of-graph representation is twofold: 1) *Content and structure description.* Graph representation ensures that the content of each observation (such as a portion of the text or a region of the image) and the structure dependency between observations (such as the contextual information between words or spatial relationships between image regions) are both properly described. This is more accurate than existing approaches that normally focus on the content, such as bag of words or visual features, but ignore the structure correlations between objects. 2) *Label ambiguity.* Real-world applications often contain inconsistent information, e.g., a webpage contains information not directly related to the subject or a drug contains properties not in the interests of the test [9]. Bag-of-graph representation allows this type of ambiguity/inconsistency in each observed bag, and then tries to find the genuine model behind the observed inconsistences. This is more realistic for modeling many real-world learning tasks.

In this paper, we generalize the aforementioned tasks as multi-graph learning (MGL) as shown in Fig. 2d, where: (1) underlying data observations are represented as graphs, and (2) the label information is only available for a bag of graphs (i.e., label ambiguity). Compared to traditional learning tasks, multi-graph learning is more general in representing real-world complex objects, and is effective for accommodating the label ambiguity.

When label information is only available for a bag of instances, traditional multi-instance (MI) learning [47] has a set of solutions which either (1) customize an existing learning algorithm to tackle the label ambiguity problem [6], [7], [28], [40], or (2) develop a learning paradigm specifically for multi-instance learning (MIL) [23], [44], [45]. However, all existing multi-instance learning algorithms can only handle tabular instances, as shown in Fig. 2c, where instances in each bag are characterized by a set of pre-defined features. For example, to represent a document by using a multi-instance model, a table can be used to represent a bag with each column (feature) denoting a keyword and each row (instance) denoting a portion of the document (such as a paragraph of the document). This instance-feature representation is commonly used in traditional supervised learning as shown in Fig. 2a. However, for graph data, such an instance-feature representation is unavailable, and there is no solution for multi-graph learning.

To classify graph structured data, existing algorithms can be roughly categorized into two groups: (1) global distance based approaches (including graph kernel [22], [34], graph embedding [30], and transformation [31]); and (2) local subgraph feature based methods [8]. The former considers a similarity function between two graphs for classification, and the latter uses a set of subgraphs to transfer graph into vector (i.e., feature) space so that generic learning algorithms can be applied. Empirical studies [17] have shown that subgraph feature based methods are generally superior to distance based algorithms, mainly because most distance approaches rely on the comparisons of



(a) Traditional supervised learning where the object for classification is an individual instance.

(b) Traditional graph learning where the object for classification is an individual graph.

(c) Multi-instance learning where each bag contains a number of instances, and the object for classification is a bag containing a number of instances.

(d) Multi-graph learning where each bag contains a number of graphs, and the object for classification is a bag containing a number of graphs.

Fig. 2. The conceptual view of different learning paradigms: (a) Supervised learning; (b) Graph learning; (c) Multi-instance learning; (d) Multi-graph learning.

global graph structures, such as common paths or walks between graphs, to calculate graph distances. In reality, even a simple graph can produce a large number of paths, which makes the calculated distance unreliable for revealing the genuine distance between graphs. In addition, the subgraph features can be served as tokens to provide a transparent view in explaining why two graphs are similar to each other. Many methods exist to extract subgraph features (such as AGM [15], Gaston [26], gSpan [43] and gHSIC [18], [46]) for classification, including a graph boosting approach [33].

For graph classification, all existing methods require each single graph to be explicitly labeled, as shown in Fig. 2b. In a multi-graph setting, label information is only available for a bag of graphs, so traditional graph classification approaches cannot be directly applied to solve our problem.

The above observations motivate the proposed research, namely Multi-Graph Feature based Learning (gMGFL), which intends to explore a set of most effective subgraphs, from multi-graph bags, to transfer each bag into an instance-feature representation. So generic learning algorithms can be applied for multi-graph classification. Due to the label ambiguity and the bag constraint, the main technical challenge of gMGFL is two-fold.

- *Subgraph feature exploration.* Finding informative subgraphs to represent each bag is crucial for multi-graph learning. Because labels are unavailable for individual graphs, the subgraph selection should take bag labels and sample distributions inside each bag into consideration to find the most informative subgraphs for bag classification.
- *Multi-graph bag representation.* Transferring multi-graph bags into instance-feature representations offers opportunities for a rich set of learning algorithms to be applied for multi-graph classification. Such a representation should preserve graph structures inside each bag and offer sufficient discriminative power for classification.

To solve the above challenges, we introduce an optimization framework that combines bag and graph level constraints to assess the informativeness score for each subgraph. We use the informativeness score as a pruning criterion, and combine subgraph mining and informative subgraph exploration to dynamically assess each subgraph and prune out uninformative subgraphs on the fly. By using selected subgraphs to represent multi-graph bags, gMGFL demonstrates good performance in solving three real-world multi-graph classification tasks.

The remainder of this paper is structured as follows. Problem definition is discussed in Section 2, followed by the overall framework in Section 3. Technical details of proposed multi-graph feature based classification algorithm gMGFL are outlined in Section 4, followed by experiments in Section 5. Section 6 reviews related work, and we conclude the paper in Section 7.

2 Problem Definition

In this section, we define important notations used in the paper.

Definition 1 (Connected Graph). *A graph is represented as $G = (\mathcal{V}, E, \mathcal{L}, l)$ where $\mathcal{V}$ is a set of vertices $\mathcal{V} = \{v_1, \dots, v_{n_v}\}$, $E \subseteq \mathcal{V} \times \mathcal{V}$ is a set of edges, and $\mathcal{L}$ is the set of labels for the vertices and edges. $l : \mathcal{V} \cup E \rightarrow \mathcal{L}$ is the function assigning labels to the vertices and edges. A connected graph is a graph that has a path between any pair of vertices. Besides, all graphs discussed in our paper are connected graphs.*

Definition 2 (Graph Bag). *A graph bag $B_i = \{G_1^i, \dots, G_j^i, \dots, G_{n_i}^i\}$ contains a number of graphs, where G_j^i and n_i denotes the jth graph and the total number of graphs in the bag, respectively. For ease of representation, we also use G_j to denote the jth graph in a given bag. A graph bag B_i's label is denoted by $y_i \in \mathcal{Y}$, with $\mathcal{Y} = \{-1, +1\}$. So a bag is either positive (B_i^+) or negative (B_i^-).*

In this paper, we use $\mathcal{B} = \{B_1, \dots, B_i, \dots, B_{N_b}\}$ to denote a set of bags, where N_b denotes the number of bags in $\mathcal{B}$. We can also aggregate all graphs in $\mathcal{B}$ as $\mathcal{G} = \{G_1, \dots, G_i, \dots, G_{N_g}\}$, where N_g denotes the number of graphs in $\mathcal{G}$. Similarly, the set of all positive bags can be denoted by $\mathcal{B}^+$, and $\mathcal{B}^-$ denotes the set of negative bags.

Definition 3 (Subgraph). *Let $G = (\mathcal{V}, E, \mathcal{L}, l)$ and $g_k = (\mathcal{V}', E', \mathcal{L}', l')$ each denote a connected graph. g_k is a subgraph of G, i.e., $g_k \subseteq G$, iff there exists an injective function $\varphi : \mathcal{V}' \rightarrow \mathcal{V}$ s. t. (1)$\forall v \in \mathcal{V}', l'(v) = l(\varphi(v))$; (2) $\forall (u, v) \in E'$, $(\varphi(u), \varphi(v)) \in E$ and $l'(u, v) = l(\varphi(u), \varphi(v))$. If g_k is a subgraph of G, then G is a supergraph of g_k.*

Definition 4 (Subgraph Feature Representation for Graph). *Let $\mathcal{S}_g = \{g_1, \dots, g_k, \dots, g_s\}$ denote a set of subgraphs discovered from a given graph set. For each graph G_i, we use a subgraph feature vector $\mathbf{x}_i^G = [(x_i^{g_1})^G, \dots, (x_i^{g_k})^G, \dots, (x_i^{g_s})^G]^\top$ to represent G_i in the feature space, where $(x_i^{g_k})^G = 1$ iff g_k is a subgraph of G_i (i.e., $g_k \subseteq G_i$) and $(x_i^{g_k})^G = 0$ otherwise.*

Definition 5 (Subgraph Feature Representation for Bag). *Given a set of subgraphs $\mathcal{S}_g$, a graph bag B_i can be represented by a feature vector $\mathbf{x}_i^B = [(x_i^{g_1})^B, \dots, (x_i^{g_k})^B, \dots, (x_i^{g_s})^B]^\top$, where $(x_i^{g_k})^B = 1$ iff g_k is a subgraph of any graph G_j in bag B_i (i.e., $\exists G_j \in B_i \wedge G_j \supseteq g_k$) and $(x_i^{g_k})^B = 0$ otherwise.*

Given a labeled multi-graph set $\mathcal{B}$, the *aim* of multi-graph learning (gMGFL) is to build a prediction model from the labeled training multi-graph set $\mathcal{B}$ to predict previously unseen multi-graph bags with maximum accuracy.

3 Overall Framework of gMGFL

Fig. 3 lists the overall framework of the proposed multi-graph feature based learning algorithm, which includes the following major steps:

- *Subgraph candidate generation.* Generating subgraph candidates is a key step towards finding informative subgraph features to represent multi-graph bags. In order to find subgraph candidates with diverse structures, we aggregate graphs in multi-graph bags into three graph sets: (1) graphs in all bags, (2) graphs in all positive bags, and (3) graphs in all negative bags. An improved gSpan [43] based subgraph mining procedure (detailed in Section 4.2) is triggered for each graph set, through which a set of

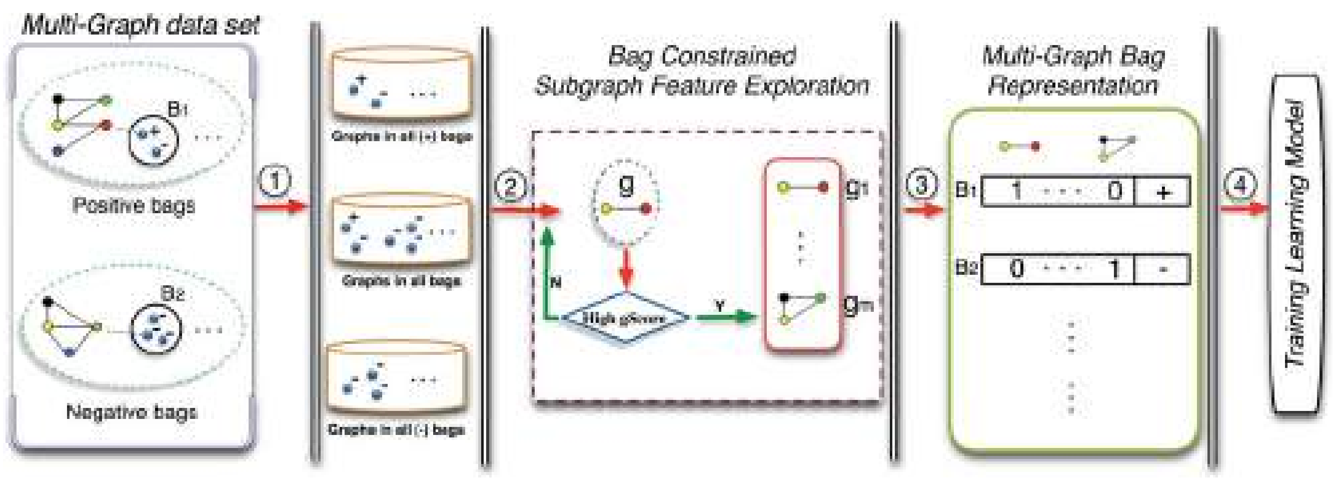


Fig. 3. The proposed multi-graph learning (gMGFL) framework: The objective is to find a set of discriminative subgraphs to convert each multi-graph bag into an instance in the new feature space for learning. gMGFL starts from subgraph candidate generation ①, and then uses the proposed bag constrained subgraph evaluation criteria to assign a score (gScore) for the discovered subgraph ②. After choosing the m subgraphs with the highest gScore in Definition 6, multi-graph bag representation ③ is used to help transfer each bag into a binary feature instance with its subgraph feature value being set to 1 if the bag (e.g., B_1) contains the subgraph feature (e.g., g_1). At last, generic learning algorithms can be applied to train learning models for multi-graph classification ④.

diverse subgraph candidate patterns are discovered for validation.

- *Bag constrained subgraph feature exploration.* A set of informative subgraph features are selected to represent each bag B_i in bag set $\mathcal{B}$. In our algorithm, we derive bag level and graph level constraints to find informative subgraph features.
- *Multi-graph bag representation.* In order to represent each bag B_i, a feature vector $\mathbf{x}_i^B = [(x_i^{g_1})^B, \ldots, (x_i^{g_k})^B, \ldots, (x_i^{g_s})^B]^\top$ is used, where $(x_i^{g_k})^B = 1$ iff any graph in bag B_i contains the given subgraph g_k and $(x_i^{g_k})^B = 0$ otherwise. An example of "Multi-graph Bag Representation" is shown in Fig. 3.
- *Classification process.* After transferring multi-graph bags into feature vectors, generic learning algorithms can be applied to train learning models for multi-graph classification.

In the following sections, we first propose our subgraph feature exploration module for gMGFL, and then discuss detailed algorithm.

4 Multi-Graph Learning Algorithm

4.1 Bag Constrained Subgraph Exploration

Subgraph feature exploration for multi-graph learning intends to assess subgraph candidates and find a set of most informative subgraphs to represent multi-graph bags. This process has two main challenges:

- How to utilize multi-graph bag labels to find informative subgraphs?
- How to tackle label ambiguity in positive bags, where genuine positive graph(s) is unknown, to find informative subgraphs?

Assume a set of graphs are collected from the bag set $\mathcal{B}$, let $\mathcal{S}_g$ denote the complete set of subgraphs discovered from $\mathcal{B}$, and $\mathbf{g} = \{g_1, \ldots, g_m\}$ is a set of subgraphs selected from $\mathcal{S}_g$. Our bag constrained subgraph feature exploration *aims* to find a set of most informative subgraph features $\mathbf{g}$ ($\mathbf{g} \subseteq \mathcal{S}_g$). To this end, we define $\mathcal{Z}(\mathbf{g})$ as an evaluation function to measure the informativeness of $\mathbf{g}$. So the objective of the subgraph feature exploration is defined in Eq. (1), where $|\cdot|$ represents the cardinality of the subgraph set, and m is the number of subgraphs to be selected from $\mathcal{S}_g$.

$$\mathbf{g}^\star = \arg\max_{\mathbf{g} \subseteq \mathcal{S}_g} (\mathcal{Z}(\mathbf{g})) \quad s.t. \ |\mathbf{g}| = m. \tag{1}$$

The objective function in Eq. (1) indicates that the bag constrained subgraph features $\mathbf{g}^\star$ should have maximum discriminative power, i.e., $\max(\mathcal{Z}(\mathbf{g}))$, among all alternative subgraph sets with the same number of subgraph features.

4.1.1 Subgraph Evaluation Criteria

In order to calculate the informativeness score of a feature set g, i.e., $\mathcal{Z}(\mathbf{g})$, we impose constraints to the bag and graph levels as follows: for any two bags B_i and B_j, if they have the same labels, we form a pairwise must-link constraint between B_i and B_j. If B_i and B_j have different labels, we form a cannot-link constraint between them. To further take the data distributions inside each bag into consideration, we also add graph level constraints to ensure that subgraph features can make graphs in each negative bag close to each other, and graphs in each positive bag are maximally separated by the subgraphs as explained below.

Accordingly, a set of good subgraph features should satisfy following constraints: (a) *Bag level must-link.* Because each bag B_i is associated with a known class label (positive or negative), the subgraph features should ensure that bags with the same label are similar to each other. (b) *Bag level cannot-link.* For bags with different class labels, subgraph features should represent the disparity between them. (c) *Graph level must-link.* In multi-graph scenarios, only graphs in negative bags are genuinely negative. The selected subgraph features should ensure that graphs in each negative bag are similar to each other so they can share commonness of being negative graphs; (d) *Graph level separability.* The ground truth labels of graphs in positive bags are unknown, although at least one graph must be positive. In this case, we use Principle Component Analysis (PCA) principle [11] and seek to find subgraph features that preserve the diverse information in the positive bags (i.e., graphs in positive bags are maximally separable).

Based on the above constraints, we derive a criterion to measure the informativeness score $\mathcal{Z}(\mathbf{g})$ as follows:

$$\begin{aligned}\mathcal{Z}(\mathbf{g}) =& \frac{1}{2A}\sum_{k=1}^{m}\sum_{y_iy_j=-1}\left(\left(x_i^{g_k}\right)^B - \left(x_j^{g_k}\right)^B\right)^2\\ &-\frac{1}{2B}\sum_{k=1}^{m}\sum_{y_iy_j=1}\left(\left(x_i^{g_k}\right)^B - \left(x_j^{g_k}\right)^B\right)^2\\ &-\frac{1}{2C}\sum_{k=1}^{m}\sum_{\forall G_i,G_j\in\mathcal{B}^-}\left(\left(x_i^{g_k}\right)^G - \left(x_j^{g_k}\right)^G\right)^2\\ &+\frac{1}{2D}\sum_{k=1}^{m}\sum_{\forall G_i,G_j\in\mathcal{B}^+}\left(\left(x_i^{g_k}\right)^G - \left(x_j^{g_k}\right)^G\right)^2,\end{aligned}\tag{2}$$

where $A=\sum_{y_iy_j=-1}1$, $B=\sum_{y_iy_j=1}1$, $C=\sum_{G_i,G_j\in\mathcal{B}^-}1$ and $D=\sum_{G_i,G_j\in\mathcal{B}^+}1$. A, B, C and D assess the total pairwise constraints in the bag cannot-link, bag must-link, graph must-link and graph separability.

By using a bag level matrix $W_B=[W_{ij}^B]^{N_b\times N_b}$, and a graph level matrix, $W_G=[W_{ij}^G]^{N_g\times N_g}$ defined in Eqs. (3) and (4), respectively.

$$W_{ij}^B=\begin{cases}1/A & y_iy_j=-1,\\ -1/B & y_iy_j=1.\end{cases}\tag{3}$$

$$W_{ij}^G=\begin{cases}-1/C & \forall G_i,G_j\in\mathcal{B}^-,\\ 1/D & \forall G_i,G_j\in\mathcal{B}^+,\\ 0 & otherwise.\end{cases}\tag{4}$$

Eq. (2) can be rewritten as follows:

$$\begin{aligned}\mathcal{Z}(\mathbf{g}) &= \mathcal{Z}(\mathbf{g})^B + \mathcal{Z}(\mathbf{g})^G\\ &=\frac{1}{2}\sum_{k=1}^{m}\sum_{y_iy_j}\left(\left(x_i^{g_k}\right)^B - \left(x_j^{g_k}\right)^B\right)^2W_{i,j}^B\\ &+\frac{1}{2}\sum_{k=1}^{m}\sum_{G_iG_j}\left(\left(x_i^{g_k}\right)^G - \left(x_j^{g_k}\right)^G\right)^2W_{i,j}^G.\end{aligned}$$

For bag level evaluation $\mathcal{Z}(\mathbf{g})^B$, we have

$$\begin{aligned}\mathcal{Z}(\mathbf{g})^B &=\frac{1}{2}\sum_{k=1}^{m}\sum_{y_iy_j}\left(\left(x_i^{g_k}\right)^B - \left(x_j^{g_k}\right)^B\right)^2W_{i,j}^B\\ &=\sum_{k=1}^{m}\sum_{y_iy_j}\left(\left(\left(x_i^{g_k}\right)^B\right)^2W_{i,j}^B - \left(x_i^{g_k}\right)^B\left(x_j^{g_k}\right)^BW_{i,j}^B\right)\\ &=\sum_{k=1}^{m}\left(\left(\boldsymbol{f}_{g_k}^B\right)^\top D_B\boldsymbol{f}_{g_k}^B - \left(\boldsymbol{f}_{g_k}^B\right)^\top W_B\boldsymbol{f}_{g_k}^B\right)\\ &=\sum_{k=1}^{m}\left(\boldsymbol{f}_{g_k}^B\right)^\top L_B\boldsymbol{f}_{g_k}^B.\end{aligned}$$

where $L_B=D_B-W_B$ is a Laplacian matrix, where $D_B=diag(d_i^B)$ is a diagonal matrix with $d_i^B=\sum_j W_{ij}^B$. $\boldsymbol{f}_{g_k}^B$ is an indicator vector of subgraph g_k with respect to all bags B_i in bag set $\mathcal{B}$, i.e., $\boldsymbol{f}_{g_k}^B=[f_{g_k}^{B_1},f_{g_k}^{B_2},\ldots,f_{g_k}^{B_{N_b}}]^\top$, where $f_{g_k}^{B_i}=1$ iff $\exists G\in B_i\wedge G\supseteq g_k$ and $f_{g_k}^{B_i}=0$ otherwise.

Similarly, the graph level evaluation $\mathcal{Z}(\mathbf{g})^G$ can be rewritten in a matrix form. By combining graph level score $\mathcal{Z}(\mathbf{g})^G$, which can be derived by using the same derivation as $\mathcal{Z}(\mathbf{g})^B$, Eq. (5) can be rewritten as follows:

$$\begin{aligned}\mathcal{Z}(\mathbf{g}) &= \mathcal{Z}(\mathbf{g})^B + \mathcal{Z}(\mathbf{g})^G\\ &=\sum_{k=1}^{m}\left(\left(\boldsymbol{f}_{g_k}^B\right)^\top L_B\boldsymbol{f}_{g_k}^B + \left(\boldsymbol{f}_{g_k}^G\right)^\top L_G\boldsymbol{f}_{g_k}^G\right)\\ &=\sum_{k=1}^{m}\boldsymbol{f}_{g_k}^\top L\boldsymbol{f}_{g_k}.\end{aligned}\tag{7}$$

In Eq. (7), $L_G=D_G-W_G$ is known as a Laplacian matrix, where $D_G=diag(d_i^G)$ is a diagonal matrix with $d_i^G=\sum_j W_{ij}^G$. $\boldsymbol{f}_{g_k}^G$ is an indicator vector of subgraph g_k with respect to all graphs G_i in $\mathcal{G}$, i.e., $\boldsymbol{f}_{g_k}^G=[f_{g_k}^{G_1},f_{g_k}^{G_2},\ldots,f_{g_k}^{G_{N_g}}]^\top$, where $f_{g_k}^{G_i}=1$ iff $g_k\subseteq G_i$ and $f_{g_k}^{G_i}=0$ otherwise. According to Eq. (7), it is

$$\boldsymbol{f}_{g_k}=\begin{bmatrix}\boldsymbol{f}_{g_k}^B\\ \boldsymbol{f}_{g_k}^G\end{bmatrix}, L=\begin{bmatrix}L_B & 0\\ 0 & L_G\end{bmatrix}.\tag{8}$$

where $\boldsymbol{f}_{g_k}$ is an indicator vector of subgraph g_k with respect to the data combined with bag and graph level. By denoting the function as $h(g_k,L)=\boldsymbol{f}_{g_k}^\top L\boldsymbol{f}_{g_k}$, the problem of maximizing $\mathcal{Z}(g_k)$ in Eq. (1) is equivalent to finding a subgraph that can maximize the $h(g_k,L)$, which can be represented as

$$\mathbf{g}^\star=\max_{\mathbf{g}}\sum\nolimits_{g_k\in\mathbf{g}}h(g_k,L)\ \ s.t.\ \ \mathbf{g}\subseteq\mathcal{S}_g.\tag{9}$$

Definition 6 (gScore). *Suppose W_B and W_G are two matrices defined as Eqs. (3) and (4), respectively. L_B is a Laplacian matrix defined as $L_B=D_B-W_B$, where D_B is a diagonal matrix with $D_{ii}^B=\sum_j W_{ij}^B$, similarly with L_G. L is also a Laplacian matrix composed of L_B and L_G, as defined in Eq. (8). The informativeness score of a subgraph g_k is defined in Eq. (10)*

$$q(g_k)=h(g_k,L)=\boldsymbol{f}_{g_k}^\top L\boldsymbol{f}_{g_k}.\tag{10}$$

In order to find the subgraph set g that maximizes the informativeness $\mathcal{Z}(\mathbf{g})$ defined in Eq. (1), we can calculate the gScore value of each individual subgraph in $\mathcal{S}_g$ and sort them, according to their gScore, in a descending order, *i.e.*, $q(g_1)\geq q(g_2)\cdots\geq q(g_s)$. Then by using the top-$m$ features $\mathbf{g}=\{g_1,g_2,\ldots,g_m\}$, we can maximize $\mathcal{Z}(\mathbf{g})$.

4.2 Bag Constrained Subgraph Search

To discover subgraphs for validation, one of the most straightforward solutions for finding a discriminative bag constrained subgraph set is exhaustive enumeration, i.e., enumerate all subgraphs in a multi-graph data set, with their gScore values being calculated for ranking. However, the number of subgraphs grows exponentially with respect to the size of graphs in bags, which makes the exhaustive enumeration approach impractical for real-world data. Alternatively, we employ a Depth-First-Search (DFS) based

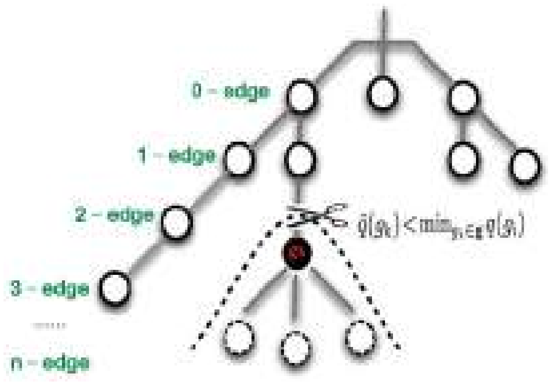


Fig. 4. A Search Space: DFS Code Tree with Branch-and-Bound Pruning.

algorithm gSpan [43] to iteratively enumerate subgraphs. The key idea of gSpan is to first assign a unique minimum DFS code to each graph, and then discover all frequent subgraphs by a pre-order traversal of the tree. Some recent graph classification approaches [19], [38], [42] incorporate constraints to prune the search space of gSpan. Specifically, gSpan labels a subgraph with a DFS code and then produces child DFS codes from the right-most path of the DFS Code Tree. If the child DFS code is a minimum DFS code, which is defined by a lexicographic order of the discovery time during the search process, the corresponding graph is processed (i.e., the DFS Code Tree, where each node is a subgraph, is obtained). By employing a depth first search strategy on the tree, gSpan can effectively enumerate all frequent subgraphs efficiently. In this paper we derive a bag constraint upper bound for the gScore to prune the search space in the DFS Code Tree, which is defined as follows:

Algorithm 1 BSE: Bag Constrained Subgraph Exploration

```
Input:
    $\mathcal{G}$: A graph data set;
    $min\_sup$: The threshold of the frequent subgraph;
    $m$: The number of subgraph features to be selected;
Output:
    $\mathbf{g} = \{g_1, g_2, \cdots, g_m\}$: A set of subgraph features;
 1: $\mathbf{g} = \emptyset$, $\tau = 0$;
 2: while Recursively visit the DFS Code Tree in gSpan do
 3:     $g_k \leftarrow$ current visited subgraph in DFS Code Tree;
 4:     if $freq(g_k) < min\_sup$, then
 5:         return;
 6:     Compute the gScore $q(g_k)$ for subgraph $g_k$;
 7:     if $|\mathbf{g}| < m$ or $q(g_k) > \tau$ , then
 8:         $\mathbf{g} \leftarrow \mathbf{g} \bigcup g_k$;
 9:     if $|\mathbf{g}| > m$, then
10:         $\mathbf{g} \leftarrow \mathbf{g} / \arg\min_{g_i \in \mathbf{g}} q(g_i)$;
11:     $\tau = \min_{g_i \in \mathbf{g}} q(g_i)$;
12:     if $\hat{q}(g_k) \geq \tau$, then
13:         Depth-first search the subtree rooted from $g_k$;
14: end while
15: return  g;
```

Theorem 1 (Upper bound of gScore). *Given two subgraphs $g_k, g_k{}' \in S_g$, g'_k is a supergraph of g_k (i.e., $g'_k \supseteq g_k$). The gScore value g'_k ($q(g'_k)$) is bounded by $\hat{q}(g_k)$, i.e., $q(g'_k) \leq \hat{q}(g_k)$, where $\hat{q}(g_k)$ is defined as follows:*

$$\hat{q}(g_k) \triangleq \boldsymbol{f}_{g_k}^\top \hat{L} \boldsymbol{f}_{g_k}, \tag{11}$$

where $\hat{L} = [\begin{smallmatrix}\hat{L}_B & 0\\ 0 & \hat{L}_G\end{smallmatrix}]$, in which the matrices $\hat{L}_B$ and $\hat{L}_G$ are defined as $\hat{L}^B_{ij} \triangleq max(0, L^B_{ij})$ and $\hat{L}^G_{ij} \triangleq max(0, L^G_{ij})$.

Proof.

$$\begin{aligned}
q(g'_k) &= \boldsymbol{f}_{g'_k}^\top L \mathbf{f}_{\mathbf{g'_k}}\\
&= \left[\left(\boldsymbol{f}^B_{g'_k}\right)^\top \left(\boldsymbol{f}^G_{g'_k}\right)^\top\right] \times \begin{bmatrix} L_B & 0\\ 0 & L_G \end{bmatrix} \times \begin{bmatrix} \boldsymbol{f}^B_{g'_k}\\ \boldsymbol{f}^G_{g'_k}\end{bmatrix}\\
&= \left(f^B_{g'_k}\right)^\top L_B f^B_{g'_k} + \left(f^G_{g'_k}\right)^\top L_G f^G_{g'_k}\\
&= \sum_{i,j:B_i,B_j \in \mathcal{B}(g'_k)} L^B_{ij} + \sum_{i,j:G_i,G_j \in \mathcal{G}(g'_k)} L^G_{ij}
\end{aligned}$$

where $\mathcal{B}(g'_k) \triangleq \{B_i | g'_k \subseteq G_j \in B_i, 1 \leq i \leq N_b, 1 \leq j \leq N_g\}$ and $\mathcal{G}(g'_k) \triangleq \{G_i | g'_k \subseteq G_j, 1 \leq j \leq N_g\}$. Since g'_k is the supergraph of g_k (i.e., $g'_k \supseteq g_k$), according to the anti-monotonic property, we have $\mathcal{B}(g'_k) \subseteq \mathcal{B}(g_k)$ and $\mathcal{G}(g'_k) \subseteq \mathcal{G}(g_k)$. Besides, $\hat{L}^B_{ij} \triangleq max(0, L^B_{ij})$ and $\hat{L}^G_{ij} \triangleq max(0, L^G_{ij})$, so $\hat{L}^B_{ij} \geq L^B_{ij}$ and $\hat{L}^G_{ij} \geq L^G_{ij}$. Both $\hat{L}^B_{ij}$ and $\hat{L}^G_{ij}$ are great than or equal to zero. Thus, Eq. (12) can be rewritten as

$$\begin{aligned}
q(g'_k) &= \sum_{i,j:B_i,B_j \in \mathcal{B}(g'_k)} L^B_{ij} + \sum_{i,j:G_i,G_j \in \mathcal{G}(g'_k)} L^G_{ij}\\
&\leq \sum_{i,j:B_i,B_j \in \mathcal{B}(g'_k)} \hat{L}^B_{ij} + \sum_{i,j:G_i,G_j \in \mathcal{G}(g'_k)} \hat{L}^G_{ij}\\
&\leq \sum_{i,j:B_i,B_j \in \mathcal{B}(g_k)} \hat{L}^B_{ij} + \sum_{i,j:G_i,G_j \in \mathcal{G}(g_k)} \hat{L}^G_{ij}\\
&= \boldsymbol{f}_{g_k}^\top \hat{L} \boldsymbol{f}_{g_k} = \hat{q}(g_k).
\end{aligned}$$

Thus, for any $g'_k \supseteq g_k$, $q(g'_k) \leq \hat{q}(g_k)$. □

This upper bound is utilized to prune DFS-code tree in gSpan by using branch-and-bound pruning as shown in Fig. 4. Algorithm 1 lists the proposed bag constrained subgraph feature exploration method, which starts with an empty feature set g and a minimum gScore $\tau = 0$. The algorithm continuously enumerates subgraphs by recursively visiting the DFS Code Tree in the gSpan algorithm. If a subgraph g_k is not a frequent subgraph, both g_k and its subtree will be pruned (line 4-5). Otherwise, we calculate g_k's gScore value $q(g_k)$. If $q(g_k)$ is larger than τ which is the minimum gScore of the current set g, or g has less than m subgraphs (i.e., **g** is not full), g_k is added to the subgraph set g (lines 7-8). If the size of g exceeds the predefined value m, we need to remove one subgraph with the least discriminative power (lines 9-10). After that, the upper bound pruning module will check if $\hat{q}(g_k)$ is less than the threshold τ. If so, it means that the gScore value of any supergraph g'_k of g_k (i.e., $g'_k \supseteq g_k$) will not be greater than τ. Therefore, we can safely prune subtrees rooted from g_k in the search space. If $\hat{q}(g_k)$ is indeed greater than the threshold τ, the depth-first search will continue by following the children of g_k (line 12-13), until the frequent subgraph mining process is completed.

4.3 gMGFL

In the above section, we have addressed the problem of finding the informative bag constrained subgraph features by avoiding exhaustive enumeration. Based on the subgraph extraction approach, the proposed gMGFL framework, as shown in Algorithm 2, uses three graph sets $\mathcal{G}$ (graphs collected from all bags), $\mathcal{G}^+$ (graphs from all positive bags), and $\mathcal{G}^-$ (graphs from all negative bags) to discover subgraph candidates. The "for" loop represents a subgraph mining processing in each graph set, which repeats as long as the DFS tree growing process continues (lines 4-6). The final subgraph set g contains subgraphs with the highest gScore with respect to the subgraphs discovered from each individual graph set (line 8). By using subgraph features in g, the original graph bags are presented as bag constrained subgraph features (line 9) to help train a classifier $\mathcal{H}$ (line 10). At the test phase, a test bag B_t is transferred into a feature vector by using g, and then predicted by the classifier $\mathcal{H}$ to obtain its class label y_t (lines 11-12).

Algorithm 2 gMGFL: Multi-graph feature based learning

```
Input:
    $\mathcal{B} = \{B_1, B_2, \cdots\}$: Multi-graph bag set;
    $m$: The number of subgraphs to be selected;
    $min\_sup$: The threshold of the frequent subgraph;
Output:
    The target class label $y_t$ of a test multi-graph bag $B_t$ based on a set of selected subgraph features $\mathbf{g} = \{g_1, g_2, \cdots, g_m\}$;
    // Training Phase:
1:  $\{\mathcal{G}, \mathcal{G}^+, \mathcal{G}^-\} \leftarrow$ all graphs in $\mathcal{B}$, all graphs in positive bags ($\mathcal{B}^+$), and all graphs in negative bags ($\mathcal{B}^-$), respectively;
2:  $\mathbf{g} = \emptyset$;
3:  for each graph set in $\{\mathcal{G}, \mathcal{G}^+, \mathcal{G}^-\}$ do
4:      $\mathbf{g}^{\mathcal{G}} \leftarrow BSE(\mathcal{G}, min\_sup, m)$; //Algorithm 1
5:      $\mathbf{g}^{\mathcal{G}^+} \leftarrow BSE(\mathcal{G}^+, min\_sup, m)$; //Algorithm 1
6:      $\mathbf{g}^{\mathcal{G}^-} \leftarrow BSE(\mathcal{G}^-, min\_sup, m)$; //Algorithm 1
7:  end for
8:  $\mathbf{g} \leftarrow$ Subgraphs with highest gScores ($\mathbf{g}^{\mathcal{G}}$, $\mathbf{g}^{\mathcal{G}^+}$, $\mathbf{g}^{\mathcal{G}^-}$);
9:  $(\mathcal{X}^*)^B \leftarrow$ bag constrained feature set from $\mathcal{B}$ and $\mathbf{g}$;
10: $\mathcal{H} \leftarrow$ classifier built from $(\mathcal{X}^*)^B$;
    // Test Phase:
11: $\mathbf{x}_t^B \leftarrow$ bag constrained feature representation for test bag $B_t$;
12: $y_t \leftarrow h(\mathbf{x}_t^B|\mathcal{H})$;
13: return $y_t$;
```

The benefit of gMGFL is twofold: (1) Separating positive and negative bags increases the diversity of the candidate subgraphs, so the subgraph feature space becomes more dense, through which a good set of subgraphs can be discovered; (2) the bag and graph level constraints fully utilize the multi-graph features to find a set of most informative subgraph features to represent multi-graph bags.

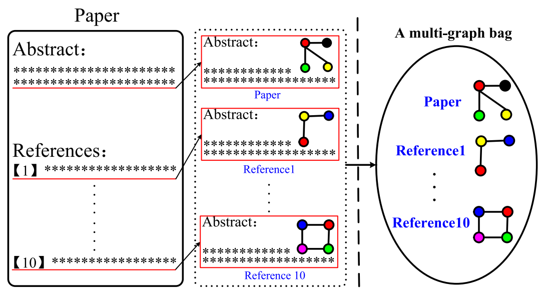


Fig. 5. An example of using a multi-graph bag to represent a research paper. Each paper is converted into an undirected graph by using the correlation of keywords in the abstract with edges denoting keyword correlations. Each reference cited in the paper also forms a graph. A bag is formed by using graphs built from the paper and references cited in the paper.

5 Experiments

5.1 Data Sets

5.1.1 DBLP Multi-Graph Data Set

The Digital Bibliography & Library Project (DBLP) data set[1] consists of bibliography data in Computer Science, with each record containing information such as abstract, authors, year, title, and references [37]. To build a multi-graph bag, we select papers published in two main fields: Artificial Intelligence (AI: IJCAI, AAAI, NIPS, UAI, COLT, ACL, KR, ICML, ECML and IJCNN) and Computer Vision (CV: ICCV, CVPR, ECCV, ICPR, ICIP, ACM Multimedia and ICME) to form a multi-graph learning task. A conceptual view of building a multi-graph bag is shown in Fig. 5. The objective is to predict whether a paper belongs to the AI or CV field by using the abstract of each paper, and the abstracts of its references. For each abstract, a fuzzy cognitive map (E-FCM) [27] based approach is used to extract a number of keywords and correlations between keywords. In our experiments, we use keywords as nodes and correlations between two keywords as edge weight values to build a graph. A threshold was used to remove edges whose correlation values are less than the given threshold. At the last step, the graph is converted into an unweighted graph by setting the weight values of all remaining edges as "1." Fig. 6 illustrates an example of using E-FCM to build a graph for a research paper entitled "Static analysis in datalog extensions" by using paper abstract. The same graph representation was also used in previous works [14], [21].

Notice that AI and CV overlap in many aspects, such as machine learning, optimization, and visual information retrieval, which makes a challenging multi-graph learning task. The original DBLP data set contains a significant number of papers without any reference. We choose 200 papers to form positive (AI) bags (100 bags with 567 graphs) and negative (CV) bags (100 bags with 569 graphs). Among the CV bags we selected, none of the references belongs to AI.

5.1.2 NCI Multi-Graph Data Set

The second task is to classify anti-cancer activities of chemical compounds on "Non-Small Cell Lung Cancer". National

1. http://dblp.uni-trier.de/xml/.

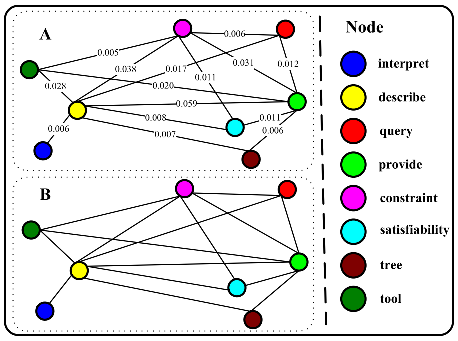


Fig. 6. An example of graph representation of a research paper entitled “Static analysis in datalog extensions” by using E-FCM [27]. The nodes are color coded with each color denoting one keyword. The edges represent correlations between keywords. For example, the correlation value between “describe” and “tree” is 0.007. In our experiments, all edges whose correlation values are less than threshold 0.005 are discarded. After that, we convert the weight graph in (a) into an unweighted graph showing in (b) for learning.

Cancer Institute (NCI) cancer screening data sets are commonly used benchmark for graph classification.[2] Each NCI data set belongs to a bio-assay task for anti-cancer activity prediction, where each chemical compound is a graph, with atoms representing nodes and bonds as edges. We build a multi-graph data set by using a graph data set (with ID 1). A chemical compound is positive if it is active against the corresponding cancer, or negative, otherwise. To build multi-graph bags, we randomly select one to four positive graphs and several negative graphs to form a positive bag. A negative bag is formed by randomly selecting a number of negative graphs to form a bag. The number of graphs in each bag varies from 1 to 10. In total, we built 100 positive and 100 negative bags, and the total number of positive and negative graphs are 208 and 901, respectively.

5.1.3 *Online Product Review Data Set*

The third learning task is to recommend high quality products to users based on the review reports. The benchmark data set is downloaded from Stanford Network Data set Collection.[3] The beer review data set *BeerAdvocate* contains review reports of different brands of beers. Each review is associated with some attributes such as product ID, reviewer ID, review score (customer rating score of the product varying from 1 to 5), and detailed review text descriptions [24], with respect to a number of key features such as “taste,” “appearance,” and “palate.” In our experiments, we consider that a product is of genuinely high quality if the average score of the product over all reviews is greater or equal to 4. On the other hand, the customer may be not satisfied in certain aspect of this product, such as taste or appearance, if all review scores of the product are less than 4. For each review report, we use fuzzy cognitive map [21] to form a graph representation with each node in the graph denoting one keyword and edges representing correlations between keywords. The detailed graph representation model is similar to the DBLP data set. In our experiments, all edges whose correlation values less than a certain threshold (0.006) are discarded. We choose 200 beer products (i.e., bags) to form 100 positive (average score ≥ 4) bags with 576 graphs (i.e., reviews) and 100 negative (each score ≤ 4) bags with 575 graphs.

2. http://pubchem.ncbi.nlm.nih.gov.
3. http://snap.stanford.edu/data/.

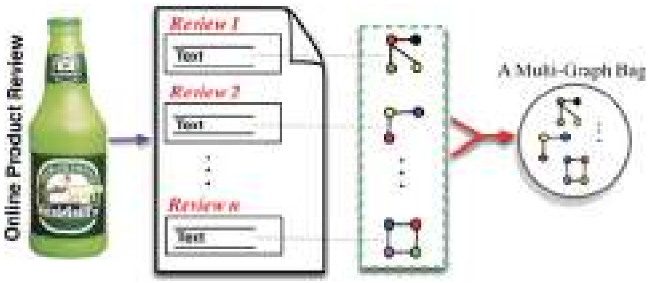


Fig. 7. An example of multi-graph representation for online product review. Each product receives a number of customer reviews, in which each review contains text descriptions of the product with respect to a number of key features such as “taste,” “appearance,” and “palate.” The main body of each review can form a graph representation by using keywords as nodes and key-word correlations as edges. The multiple reviews of each product therefore form a bag of graphs.

5.2 Baseline Methods

Because there is no existing method available for multi-graph learning, we implement the following baseline approaches for comparisons.

- *Information gain (IG) based methods (IG+MG and IG+MI).* In these approaches, a set of frequent subgraphs are mined from all bags. An Information Gain based subgraph selection criterion [18], [19], [46] is used to select m subgraphs with the highest IG scores. After obtaining the m subgraphs, IG based multi-graph method (IG+MG) uses m subgraphs to transfer each bag into one feature vector (the same as our multi-graph feature representation in Fig. 3). Meanwhile, the IG based multi-instance method (IG +MI) uses m subgraphs to transfer each graph into one instance, so a bag of graphs is converted into a bag of instances, through which the existing multi-instance learning methods can be applied for multi-graph classification.
- *Frequency based methods (Topk+MG and Topk+MI).* We also use frequency, which is an effective and classical subgraph selection method, as a measure to select top-k subgraphs with the highest frequency as m subgraph features. The top-k based multi-graph method (Topk+MG) then uses m subgraphs to convert each bag as one instance, whereas the top-k based multi-instance approach (Topk+MI) converts each bag of graphs as a bag of instances for learning.
- *Subgraph dependence evaluation based methods (gHSIC+MG and gHSIC+MI).* Recently, a dependence evaluation criterion named Hilbert-Schmidt Independence Criterion (HSIC) [12], has been successfully applied to select the discriminative subgraphs for graph related classification task (e.g., positive and unlabeled learning for graph classification [46] and multi-label graph learning [18]). In particular, gHSIC was designed to measure

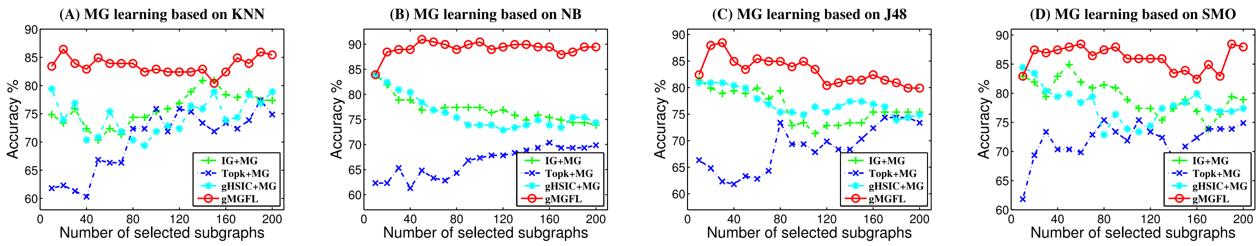


Fig. 8. Accuracy comparisons on DBLP data set by using different multi-graph learning algorithms with varied number of subgraphs: (a) KNN; (b) NB; (c) J48; and (d) SMO.

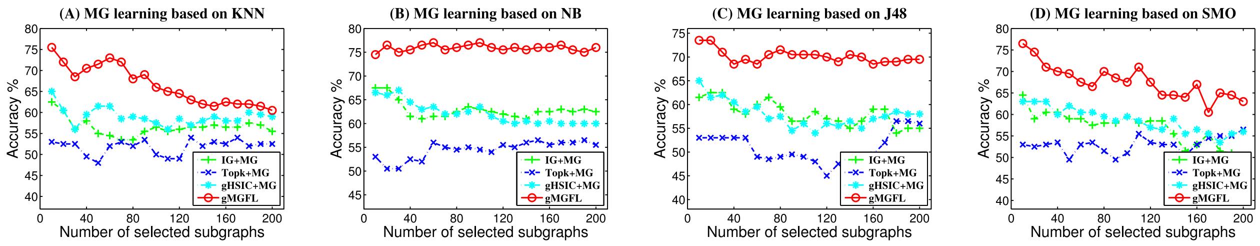


Fig. 9. Accuracy comparisons on NCI data set by using different multi-graph learning algorithms with varied number of subgraphs: (a) KNN; (b) NB; (c) J48; and (d) SMO.

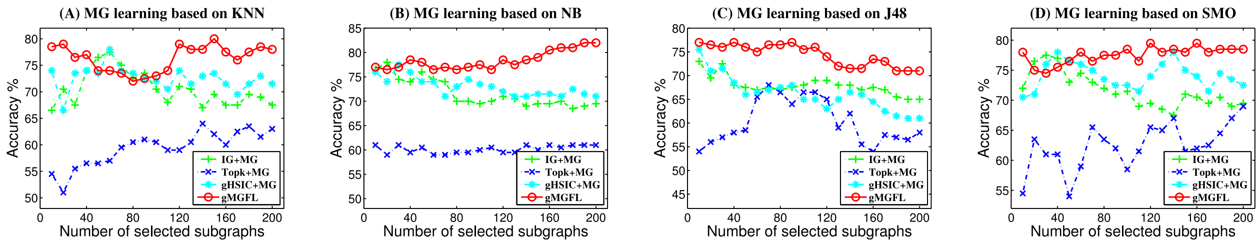


Fig. 10. Accuracy comparisons on Online Product Review data set by using different multi-graph learning algorithms with varied number of subgraphs: (a) KNN; (b) NB; (c) J48; and (d) SMO.

statistical dependency for subgraph based on the covariance operators in kernel space, which is more general than measuring dependence in the original space. As the state-of-the-art subgraph exploration method, we use gHSIC as a baseline to build multi-graph learning methods. The gHSIC based multi-graph method (gHSIC+MG) first adopts the gHSIC dependence evaluation criterion to mining m subgraphs and then uses the multi-graph representation for learning. The gHSIC based multi-instance method (gHSIC+MI) employs multi-instance setting based on the m subgraphs.

5.3 Experimental Settings

To demonstrate that our method is effective for different learning algorithms, four representative classifiers, Naive Bayes (NB), k nearest neighbors (KNN), decision trees (J48), and support vector machines (SMO) are used in our experiments. Meanwhile, in order to compare proposed gMGFL's performance with multi-instance classifiers (CitationKNN, MISMO, MIEMDD, MIOptimalBall), we also report gMGFL's mean accuracy and variance based on the four multi-graph classifiers. All experiments are based on 10 times 10-fold cross validation.

In all experiments, the default parameter settings are as follows. Minimum support threshold min_sup is set to 4 (for DBLP), 15 (for NCI) and 10 percent (for Online Product Review). Besides, all classifiers use default parameter settings in WEKA [41], such as $k = 1$ in KNN, $k = c = 1$ in CitationKNN. All experiments are collected from a Linux cluster computing node with an Interl(R) Xeon(R) @3.33 GHZ CPU and 3 GB fixed memory.

5.4 Experimental Results

5.4.1 Accuracy Comparisons with MG Setting

Figs. 8, 9 and 10 report the performance of the proposed gMGFL by using different types of multi-graph learning algorithms with respect to different number of subgraphs (varying from 10 to 200). The results show that gMGFL can achieve a high level accuracy on three data sets. The performance gain can be observed for different types of learning algorithms using multi-graph representation, such as IG+MG and Topk+MG, across different numbers of selected subgraphs. This is mainly attributed to gMGFL's two key components, including bag constrained subgraph exploration and the subsequent multi-graph bag representation.

Each sub-figure in Figs. 8, 9 and 10 (e.g., Fig. 8a) reports the performance of gHSIC+MG, IG+MG and Topk+MG, which employ HSIC dependence evaluation criterion, information gain and the frequency of the subgraphs, respectively, to represent multi-graph bags for classification. The

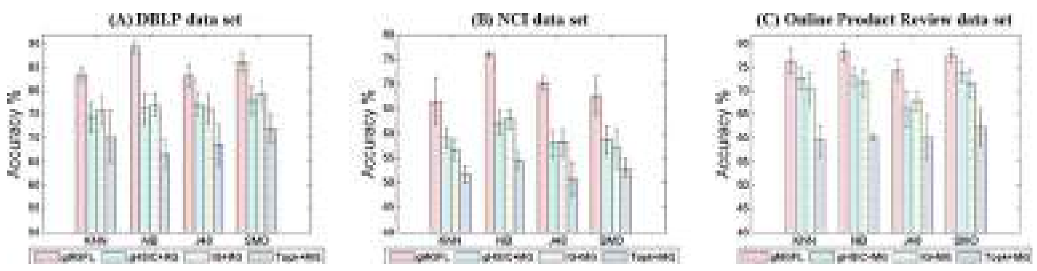


Fig. 11. Average classification accuracy (and standard deviation) comparisons between gMGFL and three multi-graph learning baseline models, gHSIC+MG, IG+MG and Topk+MG, on DBLP (a), NCI (b), and Online Product Review (c) data sets, with different number of subgraphs (varying from 10 to 200).

average accuracy with respect to different number of subgraphs on the DBLP, NCI, and Online Product Review data sets, are reported in Figs. 11a, 11b and 11c, respectively. Overall, the results show that frequency is not a good measure for identifying subgraphs to represent multi-graph bags. Although gHSIC and IG based measures are both superior to the Top-k, their subgraphs are still inefficient to represent multi-graph bags for classification, where gMGFL's accuracy is normally 10 percent more accurate on DBLP and NCI data sets and 5 percent on Online Product Review data set, as shown in Fig. 11. By combining bag and graph level constraints, gMGFL is able to find the most effective subgraphs to represent multi-graph bags for classification.

In some cases, gMGFL is less accurate than gHSIC+MG and IG+MG, especially on Online Product Review data set. For example, the results with respect to the number of subgraphs varying from 50 to 90 in Fig. 10a, 20 in Fig. 10b and 20 to 40 in Fig. 10d. To demonstrate that gMGFL is indeed statistically superior to the baselines, we report the pairwise t-test (with confident level $\alpha = 0.05$) to validate the statistical significance between four methods in Table 1, where each entry (value) denotes the p-value for a t-test between two algorithms, and a p-value less than $\alpha = 0.05$ indicates that the difference is statistically significant. The results in Table 1 confirms that gMGFL statistically outperforms gHSIC, IG+MG and Topk+MG in all cases.

5.4.2 Accuracy Comparisons with MI Setting

In Figs. 13, 14 and 15, we report the performance of gMGFL compared with four multi-instance learning algorithms (including CitationKNN, MISMO, MIEMDD, and MIOptimalBall). For these baseline methods we use subgraph features selected by using gHSIC, IG and frequency to convert each graph into one instance, so a graph bag becomes an instance bag. After that, we train four multi-instance classifiers to classify test bags. By contrast, the proposed gMGFL algorithm converts each bag as one instance and we report its mean accuracy and variance with respect to four learning algorithms (KNN, NB, J48, and SMO) in the figures. The average accuracy over subgraphs varying from 10 to 200 on the DBLP, NCI, and Online Product Review data sets are shown in Figs. 16a, 16b and 16c, respectively. Overall, the results show that gMGFL clearly outperforms existing multi-instance learning methods. This is mainly attributed to the effectiveness of the proposed subgraph feature exploration modules and the effectiveness of the multi-graph representations, which converts one bag into one instance with dense feature values for learning.

Among the baselines, MISMO based approaches achieve the best accuracy among all multi-instance learning algorithms, as shown in Fig. 16. Furthermore, the gHSIC+MI, IG+MI and Topk+MI using MISMO as the multi-instance learning approach, are all comparable with gMFGL in some cases (e.g., # of subgraphs is greater than 140 for TopK+MI in Fig. 13b). In Table 2, we report the pairwise t-test with confident level $\alpha = 0.05$ to demonstrate the statistical performance of the proposed gMFGL. The p-values (less than 0.05) in each entry assert that gMGFL statistically and significantly outperforms MI based learning methods gHSIC+MI, IG+MI and Topk+MI.

5.4.3 Subgraph Selection Quality Analysis

Noticeably, the results in Figs. 8, 9, 10 show that gMGFL's accuracies show different patterns with respect to the increasing number of subgraphs for different classifiers. More specifically, when applying gMGFL to NCI data set, the accuracy of KNN and SMO classifiers show significant drop whereas Bayes models (NB) and decision trees (J48) remain stable on the same data set. The similar trend does not exist in DBLP and Online Product Review data sets. This observation raises an important question on what is the essential connection between subgraphs selected from

TABLE 1
Pairwise t-Test Results of gMGFL versus Different Multi-Graph Learning Algorithms

DBLP				NCI				Online Product Review			
	A-B	A-C	A-D		A-B	A-C	A-D		A-B	A-C	A-D
KNN	2.7E-10	2.4E-08	4.6E-09	KNN	6.5E-08	1.4E-08	1.3E-10	KNN	2.2E-04	6.7E-05	2.1E-13
NB	2.7E-11	3.5E-12	1.6E-18	NB	7.4E-16	7.7E-17	1.2E-21	NB	6.7E-07	1.1E-06	5.9E-22
J48	1.4E-10	4.1E-10	2.8E-09	J48	5.4E-15	1.3E-15	1.4E-15	J48	1.3E-11	1.2E-11	1.6E-11
SMO	1.5E-08	4.4E-10	2.0E-14	SMO	5.0E-13	4.9E-13	2.3E-11	SMO	2.1E-05	2.8E-06	2.2E-13

A, B, C, and D denote proposed gMGFL, gHSIC+MG, IG+MG, and Topk+MG, respectively.

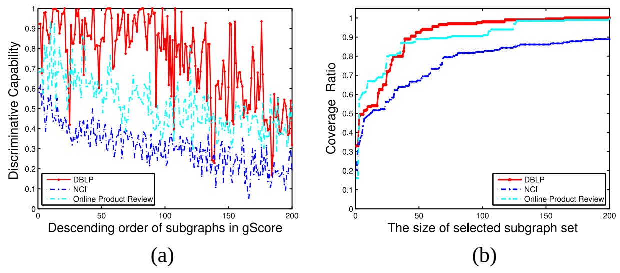


Fig. 12. (a) Discriminative capability of selected subgraphs, and (b) bag coverage rate for the subgraph set on three real-world data sets, respectively.

gMGFL and the classifiers used for learning, which results in such a distinct accuracy drop.

In order to address the above concerns, we study the discriminative capability of each single subgraph and the coverage of a subgraph set as follows. In Fig. 12a, we report the discriminative capability of each single subgraph. In the figure, the x-axis denotes the order of each subgraph as selected by gMGFL. The y-axis denotes the discriminative capability of each individual subgraph g_k, which is defined as one minus the ratio between the number of bags containing subgraph g_k in the minority classes divided by the number of bags containing subgraph g_k in the majority classes. A good subgraph will cover more bags in one class (majority class) than the other class (minority class), so its discriminative capability will approach to 1. A subgraph without discriminative power will equally cover positive and negative bags (i.e., its discriminative value will approach to 0). In Fig. 12b, we further report the coverage of the selected subgraph set with respect to the size from 1 to 200 (x-axis). The y-axis denotes the ratio between the number of bags covered by any of the subgraph and the total number of bags in the whole data set (i.e., the coverage of the subgraph set). Clearly, as the size of subgraph set increases, the coverage will continuously increase. The maximum coverage value is 1, which indicates that for each bag in the data set, at least one subgraph in the subgraph set will appear in the bag.

Fig. 12a shows that, on average, subgraphs in NCI have much lower discriminative capability than subgraphs in DBLP and Online Product Review data sets. This suggests that subgraph features in NCI are less effective to differentiate positive vs. negative bags, and are therefore less informative for classification. In addition, Fig. 12b shows that for the same number of subgraphs, the coverage of the subgraph set selected for NCI is much smaller than other two data sets. The slope of the coverage with respect to the increasing subgraph set size is also smaller than other two data sets, which indicates that adding one additional subgraph to the set does not improve the coverage of the subgraph set of NCI as much as other two data sets. One possible reason is that the new subgraph is redundant and covers similar set of bags as the existing subgraphs. From the results in Figs. 12a and 12b, we can conclude that the significant performance drop of gMGFL on NCI data set for KNN and SMO classifiers is mainly attributed to the low discriminative capability of each single subgraph and low coverage of the subgraph set. Indeed, both KNN and SMO are instance-based learning algorithms, whose classification models are based on all features (e.g., KNN uses all features to find distance between instances). Assume a redundant (or random) feature is introduced to the data set, KNN and SMO will suffer severer performance drop, compared to NB and Decision trees, which will differentiate the importance of features to build classification models.

5.4.4 Efficiency of gMGFL

In this section, we evaluate the efficiency of gMGFL's pruning module in Section 4.2. For comparison purposes, we implement an unbounded gMGFL (UgMGFL) and compare its runtime and accuracy with gMGFL, through which we

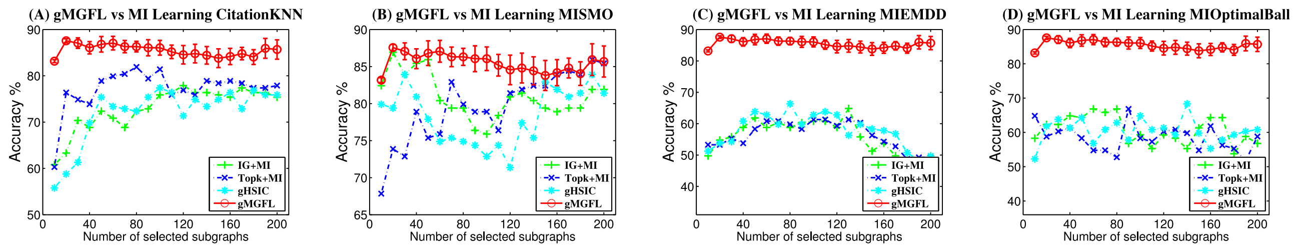


Fig. 13. Accuracy comparisons on DBLP data set by using proposed gMGFL algorithm and generic multi-instance (MI) learning methods with different number of subgraphs: (a) CitationKNN; (b) MISMO; (c) MIEMDD; and (d) MIOptimalBall.

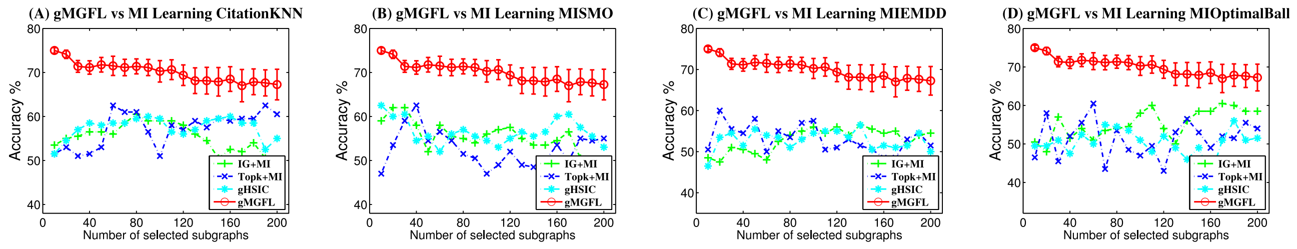


Fig. 14. Accuracy comparisons on NCI data set by using proposed gMGFL algorithm and generic multi-instance (MI) learning methods with different number of subgraphs: (a) CitationKNN; (b) MISMO; (c) MIEMDD; and (d) MIOptimalBall.

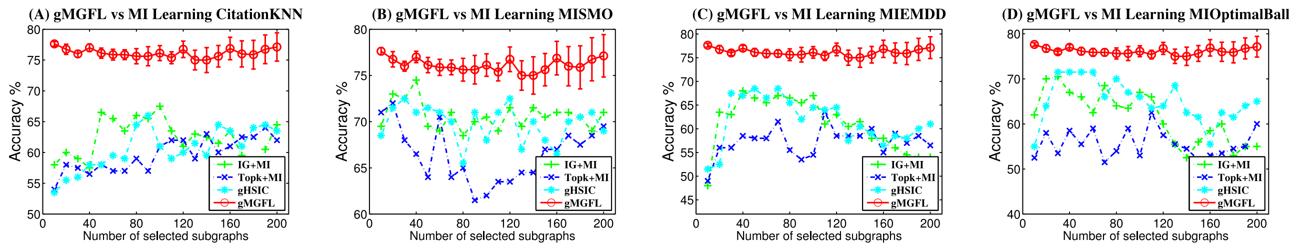


Fig. 15. Accuracy comparisons on Online Product Review data set by using proposed gMGFL algorithm and generic multi-instance (MI) learning methods with different number of subgraphs: (a) CitationKNN; (b) MISMO; (c) MIEMDD; and (d) MIOptimalBall.

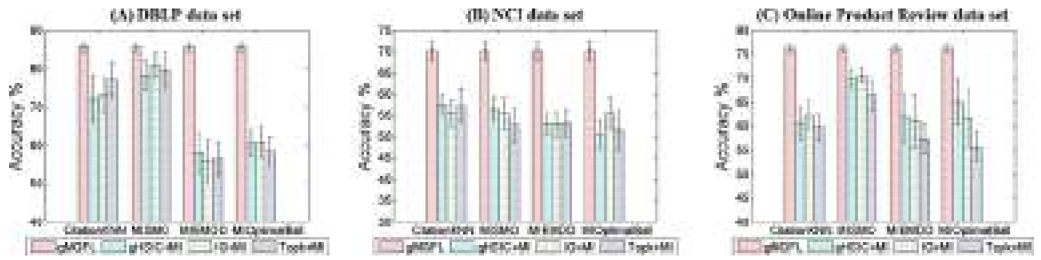


Fig. 16. Average classification accuracy (and standard deviation) comparison between gMGFL and three multi-instance learning baseline models, gHSIC+MI, IG+MI and Topk+MI, over subgraphs varying from 10 to 200 on DBLP (a), NCI (b), and Online Product Review (c) data sets, respectively.

TABLE 2
Pairwise t-Test Result of gMGFL versus Different Multi-Instance Learning Algorithms

DBLP				NCI				Online Product Review			
	A-B	A-C	A-D		A-B	A-C	A-D		A-B	A-C	A-D
CitationKNN	1.5E-08	1.5E-09	2.6E-08	CitationKNN	4.5E-12	3.4E-15	3.1E-09	CitationKNN	1.0E-13	2.3E-13	5.3E-16
MISMO	3.1E-07	4.7E-07	7.7E-05	MISMO	6.5E-14	2.1E-15	3.9E-13	MISMO	1.8E-11	5.8E-13	1.7E-12
MIEMDD	1.0E-15	2.0E-16	1.6E-17	MIEMDD	4.3E-15	1.4E-12	5.1E-16	MIEMDD	1.8E-10	5.3E-10	8.8E-16
MIOptimalBall	1.6E-18	1.8E-17	4.7E-17	MIOptimalBall	2.5E-15	3.1E-10	4.2E-12	MIOptimalBall	9.5E-09	6.5E-10	2.4E-17

A, B, C, and D denote proposed gMGFL, gHSIC+MI, IG+MI, and Topk+MI, respectively.

can demonstrate the efficiency of the pruning module. In our implementation, UgMGFL was implemented without using pruning strategy. It first uses gSpan to find a set of frequent subgraphs, and then selects the informative subgraphs by using the same criteria as gMGFL. Accordingly, gMGFL and UgMGFL share the same accuracy but different runtime performance.

In Figs. 17a, 17b and 17c, we report the average CPU runtime performance and the accuracy with respect to different minimum support min_sup values (the number of selected subgraphs is fixed to 100) on DBLP, NCI, and Online Product Review data sets, respectively. The results show that, as the min_sup values increase, the runtime and accuracy of both gMGFL and UgMGFL decrease. This is because a larger min_sup value will reduce the number of candidates for validation, which may rule out some informative subgraphs. By incorporating the proposed upper-bound pruning, gMGFL demonstrates much better runtime performance than its unbounded version. For example, when $min\_sup = 6$ percent in Fig. 17b, gMGFL's runtime is about 150 seconds, whereas UgMGFL actually doubles the running time (about 300 seconds), which implies a 50 percent efficiency gain for gMGFL.

In Figs. 18a, 18b and 18c, we also report the average CPU runtime and accuracy by varying the number of selected subgraphs from 20 to 200, on DBLP, NCI, and Online Product Review data sets, respectively (the minimum support threshold $min\_sup = 4$ for DBLP, 15 for NCI and 10 percent for Online Product Review data set). The results show that gMGFL's runtime and accuracy are relatively stable with respect to the number of selected subgraphs. For the unbounded version, increasing number of subgraph patterns will reduce its runtime efficiency. This is mainly attributed to the pruning module in gMGFL, which uses threshold min_sup and upper bound $\tau = \min_{g_i \in \mathrm{g}} q(g_i)$ (as shown in Algorithm 1) to dynamically prune the candidate set for better runtime efficiency.

6 Related Work

Multi-instance learning was first proposed by Dietterich et al. [9] for drug activity prediction. Since then, numerous approaches [47] have been proposed for different applications, such as stock market prediction, landmark matching, and computer security. In summary, existing MIL algorithms can be categorized into the following two major groups: (1) *Upgrading single-instance learners.* Updating existing single-instance based learning algorithms to support multi-instance is a common approach. Lazy learning Citation-KNN and Bayesian-KNN [40] extend k nearest-neighbor algorithm for multi-instances. Other approaches include tree based multi-instance

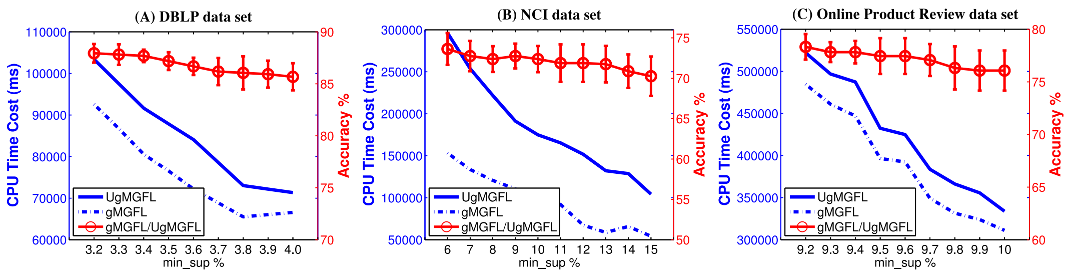


Fig. 17. Average CPU runtime and classification accuracy comparison between gMGFL *v.s.* unbounded gMGFL (UgMGFL) with respect to different min_sup and a fixed number of subgraphs $m = 100$.

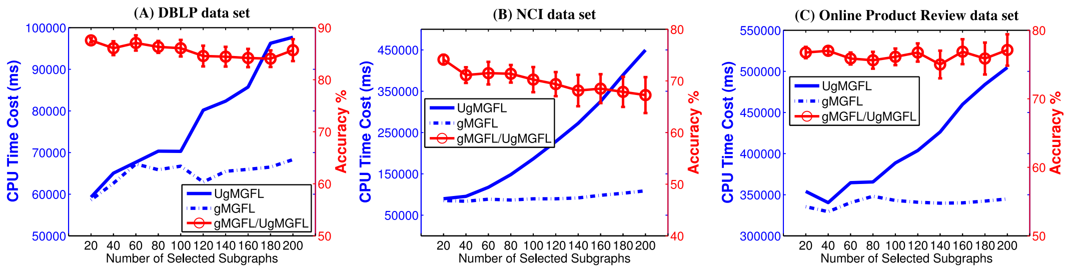


Fig. 18. Average CPU runtime and classification accuracy comparison between gMGFL *v.s.* unbounded gMGFL (UgMGFL) with respect to different number of selected subgraphs (m values) and a fixed min_sup (4 for DBLP, 15 for NCI and 10 percent for Online Product Review data set).

learning [3], multi-instance decision rules learning based on RIPPER algorithm [6], multi-instance kernel machines MISMO [28], and multi-instance logistic regression learning MILR [29]; (2)*Specifically designed MIL algorithms*. Specifically designed multi-instance learning utilizes bag constraints to reorganize instances inside each bag into specific formats for learning. Diverse Density (DD) [23] intends to search for a point in the feature space by maximizing the diverse density function that measures the co-occurrence of similar instances in different positive bags. MIEMDD [23] combines expectation-maximization (EM) with DD to search for the most likely concept, and MIOptimalBall [2] finds a suitable ball in the multiple-instance space, with a certain data point in the instance space as a ball center so that all instances of negative bags are outside the ball and at least one instance of each positive bag is inside the ball.

All existing MIL algorithms require instances in each bag to be represented as a tabular instance-feature format, so they are incapable of handling graph data.

Existing graph classification methods can be roughly classified into two categories (1) distance based approaches (including graph kernel [22], [39], graph embedding [30], and transformation [31]) and (2) subgraph feature based approaches. The former considers a similarity function between two graphs for learning whereas the latter tries to find some subgraphs as features to represent each graph in vector space for learning.

One drawback of the distance based approach is that the graph similarity is assessed based on the global graph structures, such as paths or walks, between graphs. So it is not clear which substructures (or which parts of the graphs) contribute the most to the similarity assessment. Subgraph feature based methods have been commonly used for graph learning and classification, where the main challenge is to find important subgraphs from the graph data set. The most common criterion in selecting subgraph features is frequency. For example, gSpan [43] uses a lexicographic order based coding to discover frequently connected subgraphs that can be used as features. Other frequent subgraph enumeration approaches include AGM [15], FSG [20], and Gaston [26].

In addition to the above unsupervised subgraph mining approaches, some supervised methods take class labels of individual graphs into consideration to find high quality discriminative subgraphs for learning. Examples include LEAP [42], gPLS [32], and an evolution computation based approach [16]. Moreover, some discriminative frequent pattern mining based exploration approaches in [4], [5], [10], and statistical metric pruning based methods in [25], [35] could also be applied for selecting effective subgraphs for classification. Recently, a dependence evaluation criterion named Hilbert-Schmidt Independence Criterion [12], has been successfully applied for searching high quality discriminative subgraphs for graph classification. For example, Zhao et al. [46] adopt HSIC based subgraph mining approach to deal with the positive and unlabeled learning for graph classification task. Kong and Yu [18] also use the gHSIC method to find the discriminative subgraph set for multi-label graph learning. For evaluation criterion, gHSIC assumes that the subgraph features should follow the property of dependence maximization, i.e., subgraph features should maximize the dependence between the subgraph features of graph objects and their labels.

7 Conclusion

This paper investigated a multi-graph learning task, where a number of graphs form a bag, with each bag being labeled as either positive or negative. To build a learning model for multi-graph classification, we proposed an optimization function that combines bag and graph level constraints to find a set of informative subgraph features to represent multi-graph bags as instances. So generic learning algorithms can be applied for multi-graph learning. We applied the proposed design to three real-world multi-graph tasks (DBLP citation network, NCI chemical compound classification, and Online Product Recommendation). Experiments demonstrate that our method is effective in finding informative subgraph features to represent multi-graph bags, and its classification accuracy is normally 5-10 percent higher than the baselines.

ACKNOWLEDGMENTS

This work was supported in part by the US National Science Foundation (NSF) through Grants DBI-0960443, CNS-1115234, and OISE-1129076, and the US Department of Army through grant W911NF-12-1-0066.

REFERENCES

[1] R. Angelova and G. Weikum, "Graph-Based Text Classification: Learn from Your Neighbors," *Proc. 29th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR)*, 2006.

[2] P. Auer and R. Ortner, "A Boosting Approach to Multiple Instance Learning," *Proc. 15th European Conf. Machine Learning (ECML)*, pp. 63-74, 2004.

[3] H. Blockeel and A. Srinivasan, "Multi-Instance Tree Learning," *Proc. 22nd Int'l Conf. Machine learning (ICML)*, pp. 57-64, 2005.

[4] B. Bringmann, A. Zimmermann, L. De Raedt, and S. Nijssen, "Don't Be Afraid of Simpler Patterns," *Proc. 10th European Conf. Principle and Practice of Knowledge Discovery in Databases (PKDD)*, pp. 55-66, 2006.

[5] H. Cheng, X. Yan, J. Han, and C.wei Hsu, "Discriminative Frequent Pattern Analysis for Effective Classification," *Proc. 23rd Int'l Conf. Data Eng. (ICDE)*, pp. 716-725, 2007.

[6] Y. Chevaleyre and J. Zucker, "A Framework for Learning Rules from Multiple Instance Data," *Proc. 17th European Conf. Machine Learning (ECML)*, pp. 49-60, 2001.

[7] Y. Chevaleyre and J. Zucker, "Solving Multiple-Instance and Multiple-Part Learning Problems with Decision Trees and Rule Sets. Application to the Mutagenesis Problem," *Proc. 14th Biennial Conf. Canadian Soc. Computational Studies of Intelligence: Advances in Artificial Intelligence (AI)*, pp. 204-214, 2001.

[8] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis, "Frequent Substructure-Based Approaches for Classifying Chemical Compounds," *IEEE Trans. Knowledge and Data Eng.*, vol. 17, no. 8, pp. 1036-1050, Aug. 2005.

[9] T. Dietterich, R. Lathrop, and T. Lozano-Pérez, "Solving the Multiple Instance Problem with Axis-Parallel Rectangles," *Artificial Intelligence*, vol. 89, pp. 31-71, 1997.

[10] W. Fan, K. Zhang, H. Cheng, J. Gao, X. Yan, J. Han, P. Yu, and O. Verscheure, "Direct Mining of Discriminative and Essential Frequent Patterns via Model-Based Search Tree," *Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD)*, pp. 230-238, 2008.

[11] M. Grbovic, C. Dance, and S. Vucetic, "Sparse Principal Component Analysis with Constraints," *Proc. 26th AAAI Conf. Artificial Intelligence (AAAI)*, pp. 935-941, July 2012.

[12] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, "Measuring Statistical Dependence with Hilbert-Schmidt Norms," *Proc. 16th Int'l Conf. Algorithmic Learning Theory (ALT)*, pp. 63-77, 2005.

[13] Z. Harchaoui and F. Bach, "Image Classification with Segmentation Graph Kernels," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 1-8, June 2007.

[14] X.L.Q. Hu, W. Xu, and Z. Yu, "Discovery of Textual Knowledge Flow Based on the Management of Knowledge Maps," *Concurrency and Computation: Practice and Experience*, vol. 20, no. 15, pp. 1791-1806, 2008.

[15] A. Inokuchi, T. Washio, and H. Motoda, "An Apriori-Based Algorithm for Mining Frequent Substructures from Graph Data," *Proc. Fourth European Conf. Principles of Data Mining and Knowledge Discovery (PKDD)*, pp. 13-23, 2000.

[16] N. Jin, C. Young, and W. Wang, "GAIA: Graph Classification Using Evolutionary Computation," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD)*, pp. 879-890, 2010.

[17] N. Ketkar, L. Holder, and D. Cook, "Empirical Comparison of Graph Classification Algorithms," *Proc. IEEE Symp. Computational Intelligence and Data Mining (CIDM)*, pp. 259-266, 2009.

[18] X. Kong and P. Yu, "Multi-Label Feature Selection for Graph Classification," *Proc. 10th Int'l Conf. Data Mining (ICDM)*, pp. 274-283, 2010.

[19] X. Kong and P. Yu, "Semi-Supervised Feature Selection for Graph Classification," *Proc. 16th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD)*, pp. 793-802, 2010.

[20] M. Kuramochi and G. Karypis, "Frequent Subgraph Discovery," *Proc. IEEE Int'l Conf. Data Mining (ICDM)*, pp. 313-320, 2001.

[21] X. Luo, Z. Xu, J. Yu, and X. Chen, "Building Association Link Network for Semantic Link on Web Resources," *IEEE Trans. Automation Science and Eng.*, vol. 8, no. 3, pp. 482-494, July 2011.

[22] P. Mahe, N. Ueda, T. Akutsu, J. Pettet, and J. Vert, "Extensions of Marginalized Graph Kernels," *Proc. Int'l Conf. Machine Learning (ICML)*, pp. 552-559, 2004.

[23] O. Maron and T. Lozano-Prez, "A Framework for Multiple-Instance Learning," *Proc. Conf. Advances in Neural Information Processing Systems (NIPS)*, pp. 570-576, 1998.

[24] J.J. McAuley and J. Leskovec, "From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise Through Online Reviews," *Proc. 22nd Int'l Conf. World Wide Web (WWW)*, pp. 897-908, 2013.

[25] S. Morishita and J. Sese, "Transversing Itemset Lattices with Statistical Metric Pruning," *Proc. 19th ACM SIGMOD-SIGACT-SIGART Symp. Principles of Database Systems (PODS)*, pp. 226-236, 2000.

[26] S. Nijssen and J. Kok, "A Quickstart in Frequent Structure Mining can Make a Difference," *Proc. 10th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD)*, pp. 647-652, 2004.

[27] K. Perusich and M. McNeese, "Using Fuzzy Cognitive Maps for Knowledge Management in a Conflict Environment," *IEEE Trans. Systems, Man, and Cybernetics Part C: Applications and Rev*, vol. 36, no. 6, pp. 810-821, 2006.

[28] X. Qi and Y. Han, "Incorporating Multiple SVMs for Automatic Image Annotation," *Pattern Recognition*, vol. 40, pp. 728-741, 2007.

[29] S. Ray and M. Craven, "Supervised versus Multiple Instance Learning: An Empirical Comparison," *Proc. 22nd Int'l Conf. Machine learning (ICML)*, pp. 697-704, 2005.

[30] K. Riesen and H. Bunke, "Graph Classification by Means of Lipschitz Embedding," *IEEE Trans. Systems, Man, and Cybernetics Part B: Cybernetics*, vol. 39, pp. 1472-1483, 2009.

[31] K. Riesen and H. Bunke, *Graph Classification and Clustering Based on Vector Space Embedding*. World Scientific Publishing, 2010.

[32] H. Saigo, N. Krämer, and K. Tsuda, "Partial Least Squares Regression for Graph Mining," *Proc. 14th ACM SIGKDD Int'al Conf. Knowledge Discovery and Data Mining (KDD)*, pp. 578-586, 2008.

[33] H. Saigo, S. Nowozin, T. Kadowaki, T. Kudo, and K. Tsuda, "gBOOST: A Math. Programming Approach to Graph Classification And Regression," *Machine Learning*, vol. 75, pp. 69-89, 2009.

[34] M. Seeland and A.K.S. Kramer, "A Structural Cluster Kernel for Learning on Graphs," *Proc. 18th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD)*, pp. 516-524, 2012.

[35] J. Sese and S. Morishita, "Itemset Classified Clustering," *Proc. 8th European Conf. Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pp. 398-409, 2004.

[36] J. Tang, T. Wang, Q. Lu, J. Wang, and W. Li, "A Wikipedia Based Semantic Graph Model for Topic Tracking in Blogosphere," *Proc. Int'l Joint Conf. Artificial Intelligence (IJCAI)*, pp. 2337-2342, July 2011.

[37] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: Extraction and Mining of Academic Social Networks," *Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD)*, pp. 990-998, 2008.

[38] M. Thoma, H. Cheng, A. Gretton, J. Han, H. Kriegel, A. Smola, L. Song, P. Yu, X. Yan, and K. Borgwardt, "Near-Optimal Supervised Feature Selection among Frequent Subgraphs," *Proc. SIAM Conf. Data Mining (SDM)*, pp. 1075-1086, 2009.

[39] S.V.N. Vishwanathan, K.M. Borgwardt, R.I. Kondor, and N.N. Schraudolph, "Graph Kernels," *CoRR*, abs/0807.0093, 2008.

[40] J. Wang, "Solving the Multiple-Instance Problem: A Lazy Learning Approach," *Proc. 17th Int'l Conf. Machine Learning (ICML)*, pp. 1119-1125, 2000.

[41] I. Witten. and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, The Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann 2005.

[42] X. Yan, H. Cheng, J. Han, and P.S. Yu, "Mining Significant Graph Patterns by Leap Search," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD)*, pp. 433-444, 2008.

[43] X. Yan and J. Han, "gSpan: Graph-Based Substructure Pattern Mining," *Proc. IEEE Int'l Conf. Data Mining (ICDM)*, pp. 721-724, 2002.

[44] H. Yuan, M. Fang, and X. Zhu, "Hierarchical Sampling for Multi-Instance Ensemble Learning," *IEEE Trans. Knowledge and Data Eng.*, vol. 25, no. 12, pp. 2900-2905, Dec. 2012.

[45] Q. Zhang and S. Goldman, "EM-DD: An Improved Multiple-Instance Learning Technique," *Proc. Advances in Neural Information Processing Systems (NIPS)*, pp. 1073-1080, 2001.

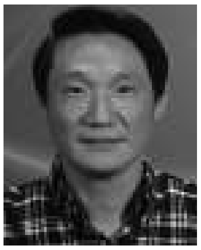
[46] Y. Zhao, X. Kong, and P.S. Yu, "Positive and Unlabeled Learning for Graph Classification," *Proc. IEEE 11th Int'l Conf. Data Mining (ICDM)*, pp. 962-971, 2011.

[47] Z. Zhou, M. Zhang, S. Huang, and Y. Li, "Multi-Instance Multi-Label Learning," *Artificial Intelligence*, vol. 176, pp. 2291-2320, 2012.

Jia Wu received the bachelor's degree in computer science from the China University of Geosciences (CUG), Wuhan, in 2009. Since September 2009, he has been working toward the PhD degree under the master's-doctor's combined program in computer science from CUG. Besides, he is also working toward the PhD degree in the Centre for Quantum Computation and Intelligent Systems, Faculty of Engineering and Information Technology, University of Technology, Sydney (UTS), Australia. His research interests include data mining and machine learning. He is a student member of the IEEE.

Xingquan Zhu received the PhD 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 journal 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* (2008-2012), and is currently serving on the Editorial Board of the *International Journal of Social Network Analysis and Mining SNAM* (2010-date) and *Network Modeling Analysis in Health Informatics and Bioinformatics Journal* (2014-date). He served or is serving as a program committee co-chair for the 14th IEEE International Conference on Bioinformatics and BioEngineering (BIBE-2014), IEEE International Conference on Granular Computing (GRC-2013), 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI-2011), and the 9th International Conference on Machine Learning and Applications (ICMLA-2010). He also served as a conference co-chair for ICMLA-2012. He is a senior member of the IEEE.

Chengqi Zhang received the PhD degree from the University of Queensland, Brisbane, Australia, in 1991 and the DSc 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*, and 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 the general chair, PC chair, or organising chair for five international Conferences including ICDM 2010 and WI/IAT 2008. He is also the general cochair of KDD 2015 in Sydney and the local arrangements chair of IJCAI-2017 in Melbourne. He is a senior member of the IEEE and a fellow of the Australian Computer Society.



Philip S. Yu received the BS degree in electrical engineering from the National Taiwan University, the MS and PhD degrees in electrical engineering from Stanford University, and the MBA degree from New York University. He is a distinguished professor in computer science at the University of Illinois at Chicago and also holds the Wexler chair in information technology. He spent most of his career at IBM, where he was manager of the Software Tools and Techniques Group at the Watson Research Center. His research interests include big data, including data mining, data stream, database, and privacy. He has published more than 780 papers in refereed journals and conferences. He holds or has applied for more than 250 US patents. Dr. Yu is a Fellow of the ACM and the IEEE. He is the editor-in-chief of the *ACM Transactions on Knowledge Discovery from Data*. He is on the steering committee of the IEEE Conference on Data Mining and ACM Conference on Information and Knowledge Management and was a member of the IEEE Data Engineering steering committee. He was the editor-in-chief of the *IEEE Transactions on Knowledge and Data Engineering* (2001-2004). He received the IEEE Computer Society 2013 Technical Achievement Award for *pioneering and fundamentally innovative contributions to the scalable indexing, querying, searching, mining and anonymization of big data,* the EDBT Test of Time Award (2014), and the Research Contributions Award from IEEE International Conference on Data Mining (2003). He had received several IBM honors including two IBM Outstanding Innovation Awards, an Outstanding Technical Achievement Award, two Research Division Awards and the 94th Plateau of Invention Achievement Awards. He was an IBM Master Inventor.