Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

Multi-Graph-View Learning for Complicated Object Classification

Jia Wu[†,‡], Shirui Pan[†], Xingquan Zhu[⋆], Zhihua Cai[‡], Chengqi Zhang[†]
[†] Quantum Computation & Intelligent Systems Centre, University of Technology, Sydney, Australia
[⋆] Dept. of Computer & Electrical Engineering and Computer Science, Florida Atlantic University, USA
[‡] Dept. of Computer Science, China University of Geosciences, Wuhan 430074, China
{jia.wu, shirui.pan}@student.uts.edu.au;
xzhu3@fau.edu; zhcai@cug.edu.cn; chengqi.zhang@uts.edu.au

Abstract

In this paper, we propose to represent and classify complicated objects. In order to represent the objects, we propose a multi-graph-view model which uses graphs constructed from multiple graph-views to represent an object. In addition, a bag based multi-graph model is further used to relax labeling by only requiring one label for a bag of graphs, which represent one object. In order to learn classification models, we propose a multi-graph-view bag learning algorithm (MGVBL), which aims to explore subgraph features from multiple graph-views for learning. By enabling a joint regularization across multiple graph-views, and enforcing labeling constraints at the bag and graph levels, MGVBL is able to discover most effective subgraph features across all graph-views for learning. Experiments on real-world learning tasks demonstrate the performance of MGVBL for complicated object classification.

1 Introduction

Many real-world objects, such as chemical compounds in biopharmacy [Kong and Yu, 2010], images in webpages [Harchaoui and Bach, 2007], brain regions in brain network [Kong and Yu, 2014] and users in social network [Aggarwal and Subbian, 2014], contain rich features and structure information. In many cases, these objects are represented by using simple features in vector space. Such a simple feature-vector representation inherently loses the structure information of the object, such as chemical bounds regulating the attraction of atoms for chemical compounds and spatial correlations of regions inside an image [Morales-Gonzalez *et al.*, 2014]. A structured representation, *i.e.,* graph, can be used to preserve structure information of the objects.

When using graphs to represent objects for learning, most existing methods construct graphs from a single feature view. For example, for content-based image retrieval, each image can be represented as a single graph-view graph by using colour histogram as a feature, with each node denoting a small region and adjacent regions being connected using an edge [Morales-Gonzalez *et al.*, 2014]. However, using graphs from a single view is inadequate to fully describe the content. For example, colour and texture are different visual channels, and are both commonly used to represent images. Therefore, using graphs constructed from multiple feature views can potentially preserve accurate information to describe the structure and the content of the object. In this paper, we refer to graphs constructed from different feature views as *multi-graph-view graphs*.

In reality, objects may have complicated characteristics, depending on how the objects are assessed and characterized. For example, in content-based image retrieval, an image may be labeled as “tiger” because it contains a tiger inside the image. However, not all regions of the image are relevant to the tighter and background regions may not be directly related to the label of the object. This representation and learning complication is known as “multi-instance” learning [Zhou, 2004], where most existing researches focus on feature-vector represented instances. In order to preserve the structure information of the object, an alternative way is to represent the object (*e.g.* an image) as a bag of graphs with each graph representing the object’s local characteristics. If any region of the image contains an object-of-interest (*e.g.* a tiger), the bag will be labeled as positive. If no region inside the image contains any object-of-interest, the bag is labeled as negative.

The above observations raise a new graph-bag based multi-graph-view learning, where the object is represented as a graph-bag, consisting of graphs collected from multiple graph-views. In order to build effective learning model, the technical challenge is twofold: (1) multiple graph-view representation: how to find effective subgraph features from different graph-views; (2) graph-bag based learning: how to integrate bag constraints, where labels are only available for a bag of graphs, for learning.

Intuitively, when objects are represented as bag of multi-graph-view graphs, a straightforward learning solution is to propagate a bag’s label to all graphs inside the bag. In this case, the problem is degraded as a “multi-graph-view graph learning” [Wu *et al.*, 2014a]. Unfortunately, because not all graphs in a positive bag are positive, simple bag label propagation may cause some negative graphs being mislabeled and deteriorate the learning accuracy. Alternatively, one can first explore some frequent subgraphs to represent graphs into vector space, and transfer the problem to “multi-view multi-instance learning” [Mayo and Frank, 2011]. However, this is still suboptimal because simple frequent subgraph features do not have sufficient discriminative ability for learning.

In this paper, we propose a multi-graph-view bag learning (MGVBL) algorithm for accurate graph-bag classification. More specially, MGVBL progressively selects the most discriminative subgraph across different graph-views. It not only achieves maximum margins between labeled graph bags (positive *vs.* negative), but also has minimum loss on the graphs in negative bags. The key contribution of the paper is threefold:

1) We propose a new object representation model which preserves the structure and the complicated characteristics of the object for learning.
2) MGVBL integrates multiple graph-view subgraph exploration and learning into a unified framework. This is inherently different from many graph learning methods, which treat subgraph exploration and subsequent model learning in two separated processes [Wu *et al.*, 2014c].
3) An upper bound discriminative score for each subgraph is derived to effectively prune subgraph search space.

2 Problem Definition and Overall Framework

Definition 1. (Connected Graph) *A graph is represented as* $G = (\mathcal{V}, E, \mathcal{L}, l)$ *where* $\mathcal{V} = \{v_1, \cdots, v_{n_v}\}$ *denotes vertices, with* $E \subseteq \mathcal{V} \times \mathcal{V}$ *as edges, and* $\mathcal{L}$ *represents symbols 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 must has a path between any pair of vertices.*

Definition 2. (Subgraph) *Let* $G = (\mathcal{V}, E, \mathcal{L}, l)$ *and* $g_i = (\mathcal{V}', E', \mathcal{L}', l')$ *each denotes a connected graph.* g_i *is a subgraph of* G*, i.e.,* $g_i \subseteq G$*, 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_i *is a subgraph of* G*, then* G *is a supergraph of* g_i*. In this paper, subgraphs and subgraph features are equivalent terms.*

Definition 3. (Graph-View) *A graph-view denotes a type of tuple* $(\mathcal{V}, E, \mathcal{L}, l)$ *used to represent an object as a graph. Similarly, multi-graph-view represents multiple types of tuples used to represent the same object.*

Definition 4. (Multi-Graph-View Graph-Bag) *A multi-graph-view graph-bag* $B_i = \{B_i^1, \cdots, B_i^k, \cdots, B_i^v\}$ *consists of some graph-bags, where* B_i^k *denotes a single-view graph-bag from the* k^{th} *view, and each* B_i^k *contains a number of graphs* G_j^k *constructed from the* k^{th} *view. The class label of the graph bag* B_i *is denoted by* $Y_i \in \mathcal{Y}$*, with* $\mathcal{Y} = \{-1, +1\}$*.*

The set of all graph-bags is denoted by $\mathcal{B}$. We can also aggregate all graphs in the negative bags as $\mathcal{G}^-$. In this paper, we use G_j to denote a multi-graph-view graph, and use superscript k to denote the k^{th} view.

Given $\mathcal{B} = \{\mathcal{B}^1, \cdots, \mathcal{B}^k, \cdots \mathcal{B}^v\}$ (*i.e.,* a multi-graph-view bag set) containing labeled graph-bags from v views, the **aim** of multi-graph-view learning for graph-bag classification is to find the optimal subgraphs from the training graph set $\mathcal{B}$ to train classification models, and predict previously unseen multi-graph-view graph-bags with a maximum accuracy.

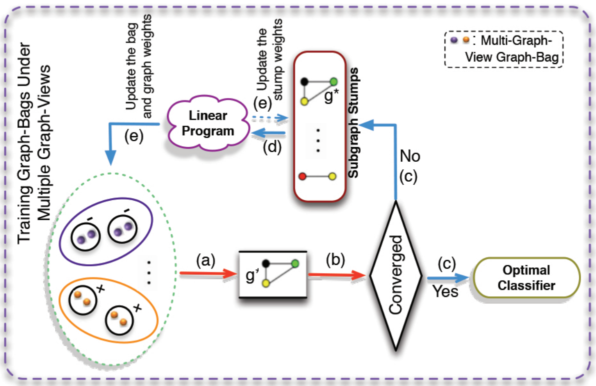

Figure 1: The proposed multi-graph-view learning for graph-bag classification (MGVBL). In each iteration, MGVBL selects an optimal subgraph g_* (step a). If the algorithm does not meet the stopping condition, g_* will be added to the subgraph set $\mathbf{g}$ or terminates otherwise (step c). During the loop, MGVBL solves a liner programming to update the weights for training graph-bags and graphs. The weights are continuously updated until obtaining the optimal classifier.

2.1 Overall Framework

Our proposed multi-graph-view bag classification framework is shown in Fig. 1. It consists of three major steps: 1) Optimal Subgraph Exploration: In each iteration, MGVBL explores a discriminative subgraph to improve the discriminative capability of the graph feature set $\mathbf{g} \ni g_*$; 2) Bag Margin Maximization: Based on the currently selected subgraphs $\mathbf{g}$, a linear programming problem is solved to achieve maximum bag margin for graph bag classification; 3) Updating Bag and Graph Weights: After the linear programming is solved, the weight values for training bags and graphs are updated and repeated until the algorithm converges.

3 Multi-Graph-View Graph-Bag Learning

3.1 Maximum Bag Margin Formulation

In graph-bag constrained learning, bag labels are asymmetric in the sense that all graphs in a negative bag are negative, while labels of graphs in positive bag are unknown. Accordingly, we can aggregate the linear constraints from two levels (bag- and graph- levels) as follows:

$$
\begin{aligned}
&\min_{\mathbf{w},\boldsymbol{\xi},\boldsymbol{\eta}} \sum_k \sum_s^{m_k} w_s^k + C_1 \sum_{i:B_i \in \mathcal{B}} \xi_i + C_2 \sum_{j:G_j \in \mathcal{G}^-} \eta_j \\
&s.t.\ Y_i \sum_k \sum_{s=1}^{m_k} (w_s^B)^k h_{g_s}(B_i^k) \geq 1 - \xi_i, i = 1, \cdots, |\mathcal{B}| \\
&\quad \sum_k \sum_{s=1}^{m_k} (w_s^G)^k h_{g_s}(G_j^k) \leq -1 + \eta_j, j = 1, \cdots, |\mathcal{G}^-| \\
&\quad \mathbf{w}^B \geq 0; \mathbf{w}^G \geq 0; \boldsymbol{\xi} \geq 0; \boldsymbol{\eta} \geq 0
\end{aligned} \tag{1}
$$

where $w_s^k = (w_s^B)^k + (w_s^G)^k$, ξ_i and η_j are the evaluation of the misclassification. m_k is the number of subgraphs selected from the kth graph-view. C_1 and C_2 are tradeoff between hyperplane margin and errors. Because bag labels are known, the weighted errors are $C_1 \sum_{i:B_i \in \mathcal{B}} \xi_i$. In addition, graphs in the negative bags are known being negative. Therefore, for the graph level the weighted errors are $C_2 \sum_{j:G_j \in \mathcal{B}^-} \eta_j$.

In Eq. (1), $h_{g_s}(B_i^k)$ is a weak subgraph classifier, which outputs the class label of the bag B_i^k in the kth view based on subgraph g_s, and $h_{g_s}(G_j^k)$ is a weak subgraph classifier for the graph G_j^k in the kth view based on subgraph g_s. More specially, for a subgraph g_s, we can use it as a decision stump classifier for graph or bag in the kth view as

$$\begin{cases} h_{g_s}(B_i^k) = (\psi_s^B)^k \big(2I(g_s \subseteq B_i^k) - 1\big) \\ h_{g_s}(G_j^k) = (\psi_s^G)^k \big(2I(g_s \subseteq G_j^k) - 1\big) \end{cases} \tag{2}$$

where $g_s \subseteq B_i^k$ iff g_s is a subgraph of any graph G in bag B_i^k, *i.e.*, $\exists G \in B_i^k \wedge g_s \subseteq G$. $(\psi_s^B)^k$ and $(\psi_s^G)^k$ ($\psi_s^B, \psi_s^G \in \Psi = \{-1, +1\}$) are parameters controlling the label of the classifiers, with $I(\cdot)$ being an indicator function. $(w_s^B)^{(k)}$ and $(w_s^G)^{(k)}$ denote the weights of the bag and graph in the kth view. For a subgraph set with size $m = \sum_k m_k$, The prediction rule for a graph-bag B_i or is a linear view combination of the corresponding weak classifiers as

$$\mathcal{H}(B_i) = sign\Big(\sum\nolimits_k \sum\nolimits_{s=1}^{m_k} (w_s^B)^k h_{g_s}(B_i^k)\Big) \tag{3}$$

3.2 Linear Programming Optimization

In order to support multi-graph-view graph-bag classification, a subgraph feature set $\mathbf{g} = \{g_1, \cdots, g_s, \cdots, g_m\}$ is required. One straightforward solution is exhaustive enumeration, which enumerates all subgraphs to find the best ones for learning. In reality, the number of subgraphs may grow exponentially, so exhaustive enumeration is technically infeasible. This problem can be solved by column generation technique [Nash and Sofer, 1996], which works on the Lagrangian dual problem with respect to Eq. (1). Starting from an empty subgraph feature set $\mathbf{g}$, column generation iteratively adds one subgraph g_s, which violates the constraint in the dual problem, to $\mathbf{g}$. Each time the subgraph set $\mathbf{g}$ is updated, column generation resolves the primal problem in Eq. (1) by solving the restricted dual problem. This procedure continues until no more subgraph violates the constraints in the dual problem, which can be formulated as

$$\begin{aligned} \max_{\boldsymbol{\gamma}, \boldsymbol{\mu}} & \sum_{i:B_i \in \mathcal{B}} \gamma_i - \sum_{j:G_j \in \mathcal{G}^-} \mu_j \\ s.t.\ & 0 \le \gamma_i \le C_1, i = 1, \cdots, |\mathcal{B}| \\ & 0 \le \mu_j \le C_2, j = 1, \cdots, |\mathcal{G}^-| \\ & \sum_k \Big(\sum_{i:B_i \in \mathcal{B}} \gamma_i Y_i h_{g_s}(B_i^k) - \sum_{j:G_j \in \mathcal{G}^-} \mu_j h_{g_s}(G_j^k)\Big) \le 2v \end{aligned} \tag{4}$$

where γ_i and μ_j are Lagrange multipliers, with $\sum_k 1 = v$. Note that, the dual problem has a limited number of variables, but a large number of constraints. Among them, each constraint $\sum_k (\sum_{i:B_i \in \mathcal{B}} \gamma_i Y_i h_{g_s}(B_i^k) - \sum_{j:G_j \in \mathcal{G}^-} \mu_j h_{g_s}(G_j^k)) \le 2v$ indicates a subgraph feature g_s over all multi-graph-view graph-bags $\mathcal{B}$, with the first and second terms of the left of constraint being the gain on the labeled graph-bags and graphs in negative bags, respectively. Intuitively, this constraint provides a metric to access the bag constraint based discriminative power of a given subgraph g_s.

Definition 5. (mgScore: Discriminative score) *Given a graph-bag set $\mathcal{B}$ containing multiple graph-view graphs, for a subgraph g_s, its discriminative score can be measured by:*

$$\pounds_{g_s} = \sum_k \Big(\sum_{i:B_i \in \mathcal{B}} \gamma_i Y_i h_{g_s}(B_i^k) - \sum_{j:G_j \in \mathcal{G}^-} \mu_j h_{g_s}(G_j^k)\Big) \tag{5}$$

To learn the multi-graph-view bag classification model, we need to find the most discriminative subgraph which considers each training bag weights and graph weights in negative bags across all graph-views for future learning.

3.3 Optimal Subgraph Exploration

In order to discover subgraphs for validation, one straightforward solution for finding an optimal subgraph set is exhaustive enumeration which enumerates all subgraphs and uses their mgScore scores for ranking. However, the number of subgraphs grows exponentially with the size of graphs in each graph-view, which makes the exhaustive enumeration approach impractical for real-world data.

Alternatively, we employ a Depth-First-Search (DFS) algorithm gSpan [Yan and Han, 2002] to iteratively enumerate subgraphs. The key idea of gSpan is that each subgraph has a unique DFS Code, defined by a lexicographic order of the discovery time during the search process. By employing a depth first search strategy on the tree, gSpan can effectively find all frequent subgraphs efficiently. In this paper, because subgraph search for each graph-view is independent, we derive an upper bound for mgScore in order to prune the search space in the DFS-code tree as follows:

Theorem 1. ***mgScore Upper Bound:*** *Given two subgraphs g_s and g_s', where g_s' is a supergraph of g_s (i.e., $g_s' \supseteq g_s$). The mgScore of g_s', $\pounds_{g_s'}$ is bounded by $\hat{\pounds}_{g_s}$, i.e., $\pounds_{g_s'} \le \hat{\pounds}_{g_s}$, with $\hat{\pounds}_{g_s}$ being defined as:*

$$\hat{\pounds}_{g_s} = max(\pounds_{g_s}^-, \pounds_{g_s}^+) \tag{6}$$

where

$$\begin{aligned} \pounds_{g_s}^- = 2\sum_k & \Big(\sum_{i:Y_i=-1, g_s \subseteq B_i^k} \gamma_i + \sum_{j:g_s \subseteq G_j^k} \mu_j \Big) \\ & + v\Big(\sum_{i:B_i \in \mathcal{B}} \gamma_i Y_i - \sum_{j:G_j \in \mathcal{G}^-} \mu_j \Big) \end{aligned} \tag{7}$$

$$\pounds_{g_s}^+ = 2\sum_k \sum_{i:Y_i=+1, g_s \subseteq B_i^k} \gamma_i - v\Big(\sum_{i:B_i \in \mathcal{B}} \gamma_i Y_i - \sum_{j:G_j \in \mathcal{G}^-} \mu_j \Big) \tag{8}$$

Proof. Considering decision stumps $h_{g_s}(B_i^k)$ and $h_{g_s}(G_j^k)$ defined in Eq. (2), we have

$$\begin{aligned} \pounds_{g_s'} = \sum_k & \Big(\sum_{i:B_i \in \mathcal{B}} \gamma_i Y_i (\psi_s^B)^k \Big(2I\big(g_s' \subseteq B_i^k\big) - 1\Big) \\ & - \sum_{j:G_j \in \mathcal{G}^-} \mu_j (\psi_s^G)^k \Big(2I\big(g_s' \subseteq G_j^k\big) - 1\Big)\Big) \end{aligned} \tag{9}$$

Algorithm 1 Discriminative Subgraph Exploration

```
Require:
    $\mathcal{B} = \{\mathcal{B}^1, \cdots, \mathcal{B}^k, \cdots, \mathcal{B}^v\}$: A multi-graph-view bag set;
    $\boldsymbol{\gamma} = \{\gamma_1, \cdots, \gamma_{|\mathcal{B}|}\}$: A bag weight set;
    $\boldsymbol{\mu} = \{\mu_1, \cdots, \mu_{|\mathcal{G}^-|}\}$: A negative graph weight set;
    $min\_sup$: The threshold of the frequent subgraph;
Ensure:
    $g_*$: The most discriminative subgraph;
 1: $g_* = \emptyset$;
 2: $\mathcal{G} = \{\mathcal{G}^1, \cdots, \mathcal{G}^k, \cdots, \mathcal{G}^v\} \leftarrow$ Aggregate all graphs in $\mathcal{B}$;
 3: for all graph-views $\mathcal{G}^k, k = 1, \cdots, v$ in $\mathcal{G}$ do
 4:     while Recursively visit the DFS Code Tree in gSpan do
 5:         $g_s^k \leftarrow$ current visited subgraph in DFS Code Tree;
 6:         if $freq(g_s^k) < min\_sup$, then
 7:             return;
 8:         Compute the mgScore $\pounds_{g_s^k}$ for subgraph $g_s^k$ using Eq. (5);
 9:         if $\pounds_{g_s^k} \geq \pounds_{g_*}$ or $g_* == \emptyset$ , then
10:             $g_* \leftarrow g_s^k$;
11:         if $\hat{\pounds}_{g_s^k} \geq \pounds_{g_*}$, then
12:             Depth-first search the subtree rooted from node $g_s^k$;
13:     end while
14: end for
15: return $g_*$;
```

For any ψ_s ((ψ_s^G) or (ψ_s^B) in any view), the value will be -1 or +1. When $\psi_s = -1$, $\pounds_{g'_s}$ can be rewritten as

$$\begin{aligned}\pounds_{g'_s} &= \sum_k \Big(-2 \sum_{i:g'_s \subseteq B_i^k} \gamma_i Y_i + \sum_{i:B_i \in \mathcal{B}} \gamma_i Y_i \\ &\quad + 2 \sum_{j:g'_s \subseteq G_j^k} \mu_j - \sum_{j:G_j \in \mathcal{G}^-} \mu_j \Big) \\ &\leq 2 \sum_k \Big(\sum_{i:Y_i=-1, g'_s \subseteq B_i^k} \gamma_i + \sum_{j:g'_s \subseteq G_j^k} \mu_j \Big) \\ &\quad + v \Big(\sum_{i:B_i \in \mathcal{B}} \gamma_i Y_i - \sum_{j:G_j \in \mathcal{G}^-} \mu_j \Big)\end{aligned} \tag{10}$$

Because

$$\sum_{i:Y_i=-1, g'_s \subseteq B_i^k} \gamma_i \leq \sum_{i:Y_i=-1, g_s \subseteq B_i^k} \gamma_i; \quad \sum_{j:g'_s \subseteq G_j^k} \mu_j \leq \sum_{j:g_s \subseteq G_j^k} \mu_j$$

We have

$$\begin{aligned}\pounds_{g'_s} &\leq 2 \sum_k \Big(\sum_{i:Y_i=-1, g_s \subseteq B_i^k} \gamma_i + \sum_{j:g_s \subseteq G_j^k} \mu_j \Big) \\ &\quad + v \Big(\sum_{i:B_i \in \mathcal{B}} \gamma_i Y_i - \sum_{j:G_j \in \mathcal{G}^-} \mu_j \Big) = \pounds_{g_s}^-\end{aligned} \tag{11}$$

For any $g'_s \supseteq g_s$ under $\psi_s = -1$, we have $\pounds_{g'_s} \leq \pounds_{g_s}^-$. Similarly, $\pounds_{g'_s} \leq \pounds_{g_s}^+$ under $\psi_s = +1$. In this case, the maximum one (*i.e.* $max(\pounds_{g_s}^-, \pounds_{g_s}^+)$) will be selected as the final upper bound $\hat{\pounds}_{g_s}$. Once a subgraph g_s is generated, all its supergraphs are upper bounded by $\hat{\pounds}_{g_s}$. Therefore, this theorem can help prune the search space efficiently. □

Algorithm 2 MGVBL: Multi-Graph-View Bag Learning

```
Input:
    $\mathcal{B} = \{\mathcal{B}^1, \cdots, \mathcal{B}^k, \cdots, \mathcal{B}^v\}$: A multi-graph-view bag set;
    $min\_sup$: The threshold of the frequent subgraph;
    $m$: the maximum number of iteration;
Output:
    The target class label $Y_c$ of a test multi-graph-view bag $B_c$;
    // Training Phase:
 1: $\boldsymbol{g} \leftarrow \emptyset$; $t \leftarrow 0$;
 2: while $t \leq m$ do
 3:     $g_* \leftarrow$ Apply $\mathcal{B}$ and $min\_sup$ to obtain the most discriminative subgraph ;  // Algorithm 1
 4:     if $\pounds_{g_*}/2v \leq 1 + \epsilon$ then
 5:         break;
 6:     $\mathbf{g} \leftarrow \mathbf{g} \cup g_*$;
 7:     Solve Eq. (1) based on $\mathbf{g}$ to get $\boldsymbol{w}^B$ and $\boldsymbol{w}^G$, and the Lagrange multipliers of Eq. (4) $\boldsymbol{\gamma}$ and $\boldsymbol{\mu}$;
 8:     $t \leftarrow t + 1$;
 9: end while
    // Testing Phase:
10: $Y_c \leftarrow sign\Big(\sum_k \sum_{g_s \in \mathbf{g}} \big(w_s^B\big)^k h_{g_s}\big(B_c^k\big)\Big)$.
11: return $Y_c$.
```

The above upper bound can be utilized to prune DFS code tree in gSpan by using branch-and-bound pruning, where the complete subgraph feature exploration approach is listed in Algorithm 1. The algorithm enumerates subgraph features by searching the whole DFS code tree in each graph-view. If a current subgraph g_s^k in the kth view is not frequent, both g_s^k and its related subtree will be discarded (lines 6-7). Otherwise, the mgScore of the g_s^k (*i.e.* $\pounds_{g_s^k}$) will be calculated (line 8). If $\pounds_{g_s^k}$ is greater than the current optimal mgScore $\pounds_{g_*}$ or it is the first step (*i.e.* the optimal subgraph $\pounds_{g_*}$ is empty), $\pounds_{g_s^k}$ will be regarded as the current optimal subgraph $\pounds_{g_*}$ (lines 9-10). Subsequently, the upper bound pruning module will check if $\hat{\pounds}_{g_s^k}$ is less than the $\pounds_{g_*}$, which means that the mgScore value of any supergraph $g_s^{k'}$ of g_s^k (*i.e.* $g_s^{k'} \supseteq g_s^k$) will not be greater than $\pounds_{g_*}$. If so, the subtree rooted from g_s^k will be safely pruned. If $\hat{\pounds}_{g_s^k}$ is indeed greater than the mgScore of g_*, the search process will sequentially visit nodes from the subtree of g_s^k (lines 11-12).

3.4 MGVBL

The complete procedures of the proposed subgraph mining and classification framework MGVBL are listed in Algorithm 2, which iteratively extracts informative subgraphs across different graph-views to expand the candidate subgraph set $\mathbf{g}$, by using mgScore. After m iterations, MGVBL boosts the m selected weak classifiers to form a final classification model.

MGVBL starts from an empty subgraph set $\mathbf{g} = \emptyset$ (line 1), and iteratively selects the most discriminative subgraph g_* in each round (line 3) according to Algorithm 1. If the current optimal subgraph no longer violates the constraint, the iteration process terminates (lines 4-5). Because the difference of the optimal values in the last few iterations is relatively small, a threshold ϵ is used to relax the stopping condition, so MGVBL terminates if the difference between two consecutive iterations is less than the threshold ϵ (we set $\epsilon = 0.05$ in our experiments). After that, MGVBL solves the linear programming problem by using the current optimal subgraph set $\mathbf{g}$ to recalculate two groups of weight values: 1)$\boldsymbol{w}^B$ and $\boldsymbol{w}^G$: the weights for bag-level and graph-level weak subgraph decision stumps, respectively; 2) $\boldsymbol{\gamma}$ and $\boldsymbol{\mu}$: the weights of training bags and graphs in negative bags for optimal subgraph mining in the next round, which can be obtained from the Lagrange multipliers of the primal problem (line 7). Once the algorithm converges or the number of maximum iteration is reached, the training phase of MGVBL is finished. During the testing phase, the label Y_c of a test bag B_c is determined by the final classier $sign(\sum_k \sum_{g_s \in \mathbf{g}} (w_s^B)^k h_{g_s}(B_c^k))$.

4 Experiments

4.1 Experimental Settings

All reported results are based on 10 times 10-fold cross-validation. Unless specified otherwise, we set minimum support threshold $min\_sup = 3\%$ for scientific publication data (Section 4.3) and $min\_sup = 2\%$ for content-based image retrieval (Section 4.4).

4.2 Baseline Methods

Because no existing approaches are available to solve the proposed research problem, we use the following two types of baselines (bag-level and graph-level) for comparison studies. Bag-level approaches firstly discover some informative subgraphs to represent graphs in the bag set (*i.e.* transferring multi-graph set to a multi-instance set), and then employ the existing multi-view multi-instance learning MIVBL [Mayo and Frank, 2011] for classification. Graph level methods propagate bag labels to all graphs inside the bag, so the problem is transferred to a recently proposed multi-graph-view graph learning (MGVGL) [Wu *et al.*, 2014a].

Bag-level approach

A number of top-k frequent subgraphs are explored to represent graphs as feature-vector instances, and transfer the problem as multi-view multi-instance learning. After that, MIVBL is applied to directly train multi-instance boosting classifier (MIBoost) [Xu and Frank, 2004] by treating each view independently and combining classifiers across all views for prediction. Previous research [Mayo and Frank, 2011] has indicated that sophisticated combining approaches, such as stacking [Wolpert, 1992], are possible but their initial experiments did not yield improved results and the training times were an order of magnitude greater. So we only use simple combination in our experiments.

Graph-level approach

Because labels are only available for bags, graph-level approaches directly propagate bag labels to graphs inside each bag. By doing so, the problem is transferred to a graph learning task with multiple graph-views. After that, graph-level approaches first explore an optimal set of subgraphs as features to transfer multi-graph-view graphs into feature-vectors, with an AdaBoost [Telgarsky, 2012] classifier being trained for final prediction. It is worth noting that we use AdaBoost because our proposed MGVBL is a boosting formwork, so it makes sense to compare with a boosting based method.

In addition, we also implement a bMVGBL approach (*i.e.* MVGBL without using the graph level constraint) as a baseline to explore the efficiency of the unified two level (bag- and graph- level) framework.

4.3 Scientific Publication Text Categorization

A scientific publication can be represented as multi-graph-view graphs. To build multiple graph-view, we use information from (1) abstract; and (2) paper ID, title, and references, to form two graph-views. More specifically, for abstract view graphs, each paper is converted into an undirected graph by using the correlation of keywords in the abstract with edges denoting keyword correlations. For paper ID, title, and reference view graph, each node denotes Paper ID or a keyword in the title and each edge denotes the citation relationship between papers or keyword relations in the title (detailed in [Pan *et al.*, 2015]). For each paper, each reference cited by the paper is also a graph, so each paper with its citations, corresponds to a graph bag with two views (*i.e.,* abstract view *vs.* reference relationship view). More specifically, assume paper A cites papers A_1, A_2, and A_3, and the label of A is "Positive". For each view, we will first generate one graph from A, A_1, A_2, and A_3, respectively. After that, we put all four graphs as one bag, and label the bag as "Positive".

The Digital Bibliography $\&$ Library Project (DBLP) data set [1] consists of bibliography in computer science. Each record in DBLP contains information such as abstract, authors, year, title, and references. To build a multi-graph-view graph-bag, we select papers published in two relevant research fields: Artificial Intelligence (AI: IJCAI, AAAI, NIPS, UAI, COLT, ACL, KR, ICML, ECML, and IJCNN), and Database (DB: SIGMOD, KDD, PODS, VLDB, ICDE, CIKM, DASFAA, ICDT, and SSDBM) to form a multi-graph-view graph-bag learning task. The objective is to predict whether a paper belongs to the AI or DB field.

For each abstract in the abstract graph-view, a fuzzy cognitive map (E-FCM) [Luo *et al.*, 2011] based approach is used to extract a number of keywords and correlations between keywords, which form nodes (keywords) and edges (keyword correlations) of each graph. A threshold (0.005) is used to remove edges whose correlations are less than the threshold. At the last step, the graph is converted into an unweighted graph by setting the weight of all remaining edges as "1". In the experiments, we choose 600 papers in total (corresponding to 600 multiple graph-views bags) to form positive (AI) bags (300 bags with 1756 graphs) and negative (DB) bags (300 bags with 1738 graphs).

Fig. 2 (A) reports accuracy for multi-graph-view graph-bag classification. When the number of selected subgraphs is less than 10, the performances of all algorithms are comparably low, mainly because a small number of subgraph stumps (*i.e.* weak classifiers) lead to inferior classification accuracies in early iterations. By contrast, MGVBL consistently outperforms baselines when the number of selected subgraphs is 20 or more. During last a few iterations, MGVGL can obtain a high accuracy, but this type of baselines still cannot outperform the best performance achieved by MGVBL.

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

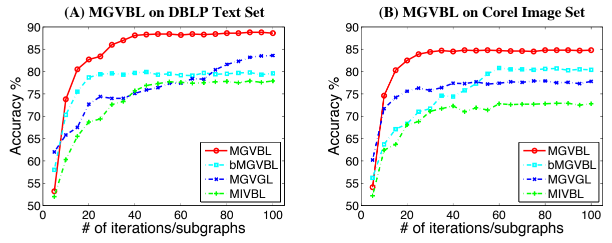

Figure 2: Experimental results for multi-graph-view graph-bag learning on (A) DBLP Text and (B) Corel Image data set.

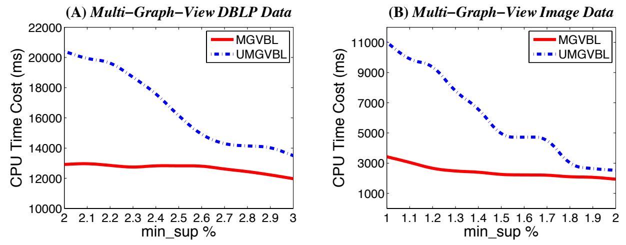

Figure 3: Average CPU runtime comparison between MGVBL *v.s.* unbounded MGVBL (UMGVBL) with respect to different min_sup values.

4.4 Content-based Image Retrieval

In this section, we report MGVBL's performance for content based image retrieval. The original images [Li and Wang, 2008] from Corel data set[2] are preprocessed by using VLFeat segmentation [Vedaldi and Fulkerson, 2008], with each image being segmented into multiple regions and each region corresponding to one graph. For each region, Simple Linear Iterative Clustering (SLIC) [Achanta *et al.*, 2012] is applied to obtain graph representation, so each node indicates one superpixel and each edge represents the adjacency relationship between two superpixels [Wu *et al.*, 2014b].

In order to build multiple graph-views, we employ two types of features, including Hue-Saturation-Value (HSV) in colour space and Local Binary Patterns (LBP) in the texture space. Specifically, HSV is a commonly used color model, where HSV stands for hue, saturation and intensity, and LBP represents texture in a local region. For HSV feature, we first extract a 3-channel HSV feature for each pixel. The extracted HSV representations are fed to k-means clustering to construct a 256-dimensional codebook. After that, a one-dimensional code is assigned to each pixel based on the similarity between the pixel representation and the cluster centers. The HSV-based representation for a superpixel is constructed as a 256-dimensional histogram-based vector by computing the statistics of the occurrences of the codes. For LBP, we adopt the uniform LBP and generate a 59-bin code for each pixel, where each pixel is assigned to one bin according to the local texture pattern. Therefore, a 59-dimensional histogram is constructed for each superpixel encoding its statistics.

To build positive bags, we use category "Cats", which consists of "Tiger", "Lion" and "Leopard", as positive bags (300 bags with 2679 graphs) and randomly draw 300 images of other animals to form negative bags with 2668 graphs (*i.e.* image regions). Fig. 2(B) shows the classification results from 1 to 100 iterations. With the two bag- and graph-level constraints, MGVBL achieves better performance than bMGVBL (only considers bag level constraints), indicating that information in negative bags is very helpful for learning the model. MIVBL has the worst performance, mainly because their frequent subgraphs are not carefully selected and therefore are not discriminative. Although MGVGL is comparable to bMGVBL, MGVGL is inferior compared to the proposed MGVBL, due to the fact that MGVGL directly propagates bag labels to graphs and cause negative graphs in positive bags being mislabeled as positive. Overall, the proposed MGVBL has the best learning performance.

[2]https://sites.google.com/site/dctresearch/Home/content-based-image-retrieval

4.5 Efficiency of the Pruning Strategy

In order to evaluate the efficiency of the proposed pruning module for searching subgraphs as described in Section 3.3, we implement an UMGVBL approach with no pruning module and compare its runtime with MGVBL. In our implementation, UMGVBL first uses gSpan to find a frequent subgraph set, and then selects the optimal subgraph by using the same evaluation criteria as MGVBL.

The results in Fig. 3 show that increasing min_sup values decreases the runtime of UMGVBL, mainly because a larger min_sup value reduces the candidate number for validation. By using pruning strategy (*i.e.* the constraints including threshold min_sup and upper bound $\hat{\pounds}_{g_s} = max(\pounds^-_{g_s}, \pounds^+_{g_s})$ as shown in Algorithm 1), MGVBL's runtime performance is relatively stable *w.r.t.* different min_sup values. Overall, MGVBL demonstrates clear advantage compared to unbounded UMGVBL, especially when min_sup is small.

5 Conclusion

This paper investigated the representation and classification of complicated objects by using graph-bag based multi-graph-view learning. We argued that many real-world objects contain structure information from different views, and multi-graph-view graph-bag representation provides an effective way to preserve structure and complicated features of the object for learning. To build a learning model for multi-graph-view classification, we formulated an objective function to jointly regularize multiple graph-views, and enforce labeling constraints at bag and graph levels, respectively, so our algorithm can discover most effective subgraph features across all graph-views to optimize the learning. The key contribution of the paper, compared to existing works, is threefold: (1) a new multi-graph-view representation model to represent complicated objects; (2) a cross graph-view search space pruning strategy; and (3) a combined cross-view subgraph feature exploration and learning method.

Acknowledgments

The work was supported by Australian Research Council (ARC) Discovery Projects under Grant No. DP140100545 and DP140102206.

References

[Achanta *et al.*, 2012] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurlien Lucchi, Pascal Fua, and Sabine Ssstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 2274–2282, November 2012.

[Aggarwal and Subbian, 2014] Charu Aggarwal and Karthik Subbian. Evolutionary network analysis: A survey. *ACM Computing Surveys*, 47(1):10, May 2014.

[Harchaoui and Bach, 2007] Z. Harchaoui and F. Bach. Image classification with segmentation graph kernels. In *Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'07, pages 1–8, Minneapolis, Minnesota, USA, June 2007.

[Kong and Yu, 2010] Xiangnan Kong and Philip S. Yu. Semi-supervised feature selection for graph classification. In *Proceedings of the 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD'10, pages 793–802, Washington, DC, USA, July 2010.

[Kong and Yu, 2014] Xiangnan Kong and Philip S. Yu. Brain network analysis: A data mining perspective. *SIGKDD Explor. Newsl.*, 15(2):30–38, June 2014.

[Li and Wang, 2008] Jia Li and James Z. Wang. Real-time computerized annotation of pictures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:985–1002, June 2008.

[Luo *et al.*, 2011] X.F. Luo, Z.X., J. Yu, and X. Chen. Building association link network for semantic link on web resources. *IEEE Transactions on Automation Science and Engineering*, 8(3), July 2011.

[Mayo and Frank, 2011] Michael Mayo and Eibe Frank. Experiments with multi-view multi-instance learning for supervised image classification. In *Proceedings of 26th International Conference Image and Vision Computing New Zealand*, IVCNZ'11, pages 363–369, Auckland, New Zealand, November 2011.

[Morales-Gonzalez *et al.*, 2014] Annette Morales-Gonzalez, Niusvel Acosta-Mendoza, Andres Gago-Alonso, Edel Garcia-Reyes, and Jose Medina-Pagola. A new proposal for graph-based image classification using frequent approximate subgraphs. *Pattern Recognition*, pages 169–177, January 2014.

[Nash and Sofer, 1996] Stephen G. Nash and Ariela Sofer. *Linear and Nonlinear Programming*. McGraw-Hill, 1996.

[Pan *et al.*, 2015] S. Pan, J. Wu, X. Zhu, and C. Zhang. Graph ensemble boosting for imbalanced noisy graph stream classification. *Cybernetics, IEEE Transactions on*, 45(5):940–954, May 2015.

[Telgarsky, 2012] Matus Telgarsky. A primal-dual convergence analysis of boosting. *Journal of Machine Learning Research*, 13(1):561–606, March 2012.

[Vedaldi and Fulkerson, 2008] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.

[Wolpert, 1992] David H. Wolpert. Original contribution: Stacked generalization. *Neural Networks*, 5(2):241–259, February 1992.

[Wu *et al.*, 2014a] Jia Wu, Zhibin Hong, Shirui Pan, Xingquan Zhu, Zhihua Cai, and Chengqi Zhang. Multi-graph-view learning for graph classification. In *Proceedings of the 2014 IEEE International Conference on Data Mining*, ICDM'14, pages 590–599, Shenzhen, China, December 2014.

[Wu *et al.*, 2014b] Jia Wu, Zhibin Hong, Shirui Pan, Xingquan Zhu, Chengqi Zhang, and Zhihua Cai. Multi-graph learning with positive and unlabeled bags. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, SDM'14, pages 217–225, Philadelphia, Pennsylvania, USA, April 2014.

[Wu *et al.*, 2014c] Jia Wu, Xingquan Zhu, Chengqi Zhang, and P.S. Yu. Bag constrained structure pattern mining for multi-graph classification. *Knowledge and Data Engineering, IEEE Transactions on*, 26(10):2382–2396, October 2014.

[Xu and Frank, 2004] Xin Xu and Eibe Frank. Logistic regression and boosting for labeled bags of instances. In *Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, PAKDD'04, pages 272–281, Sydney, Australia, May 2004.

[Yan and Han, 2002] Xifeng Yan and Jiawei Han. gspan: Graph-based substructure pattern mining. In *Proceedings of the 2002 IEEE International Conference on Data Mining*, ICDM'02, pages 721–724, Maebashi City, Japan, December 2002.

[Zhou, 2004] Z.H. Zhou. Multi-instance learning: A survey. Technical report, AI Lab, Department of Computer Science & Technology, Nanjing University, Nanjing, China, 2004.