Reinforced GNNs for Multiple Instance Learning

Xusheng Zhao, Xu Bai, Qiong Dai, Jia Wu, Hao Peng, Huailiang Peng, Zhengtao Yu, Philip S. Yu, *Fellow, IEEE*

***Abstract*—Multiple instance learning (MIL) trains models from bags of instances, where each bag contains multiple instances, and only bag-level labels are available for supervision. The application of graph neural networks (GNNs) in capturing intra-bag topology effectively improves MIL. Existing GNNs usually require filtering low-confidence edges among instances and adapting graph neural architectures to new bag structures. However, such asynchronous adjustments to structure and architecture are tedious and ignore their correlations. To tackle these issues, we propose a *Reinforced GNN framework for MIL*, RGMIL, pioneering the exploitation of multi-agent deep reinforcement learning (MADRL) in MIL tasks. MADRL enables the flexible definition or extension of factors that influence bag graphs or GNNs and provides synchronous control over them. Moreover, MADRL explores structure-to-architecture correlations while automating adjustments. Experimental results on multiple MIL datasets demonstrate that RGMIL achieves the best performance with excellent explainability. The code and data are available at https://github.com/RingBDStack/RGMIL.**

***Index Terms*—Multiple instance learning, neural architecture search, graph neural network, deep reinforcement learning.**

I. Introduction

ONE of the disadvantages of standard supervised learning is that it cannot handle training samples with ambiguous or missing labels. As a variant of supervised learning, multiple instance learning (MIL) can address this weakness. Instead of treating labeled instances as training samples, MIL treats bags that contain multiple instances as samples, and only the entire bags are assigned labels. A bag is labeled as positive if it holds at least one positive instance. Otherwise, a bag will be labeled as negative, as shown in Fig. 1. MIL is dedicated to forecasting bag-level labels and identifying instance importance.

MIL is common in real-world applications, especially in the medical field [1]. For example, in breast cancer prediction [2], it is significant for medical assistance to diagnose breast tissue images that contains multiple regions of interest (ROIs), where each ROI can be represented as an instance. Therefore, a tissue image can be represented as a bag containing several instances. An image bag is malignant (positive) if it contains at least one malignant ROI instance. Otherwise, a bag is benign (negative).

Xusheng Zhao, Qiong Dai, Xu Bai, and Huailiang Peng are with the Institute of Information Engineering, Chinese Academy of Sciences and the School of Cyber Security, University of Chinese Academy of Sciences, Beijing, 100084, China. E-mail: {zhaoxusheng, daiqiong, baixu, penghuailiang}@iie.ac.cn.

Jia Wu is with the Department of Computing, Macquarie University, Sydney NSW 2109, Australia. E-mail: jia.wu@mq.edu.au.

Hao Peng is with the School of Cyber Science and Technology, Beihang University, Beijing, 100191, China. E-mail: penghao@buaa.edu.cn.

Zhengtao Yu is with the Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China. E-mail: ztyu@hotmail.com.

Philip S. Yu is with the Department of Computer Science, University of Illinois at Chicago, Chicago, IL 60607, USA. E-mail: psyu@uic.edu.

Corresponding authors: Hao Peng and Qiong Dai

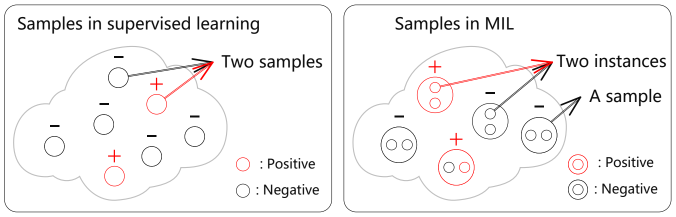


Fig. 1: Examples of training samples from supervised learning or MIL. In supervised learning, samples are labeled instances.

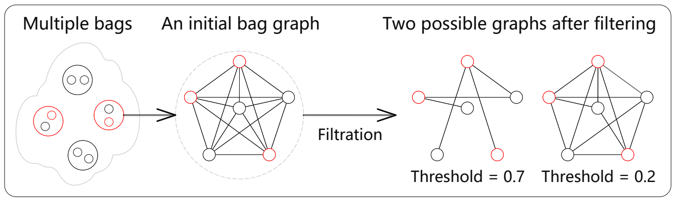


Fig. 2: A toy example of converting a bag into possible graphs based on different edge filtering thresholds.

Under this assumption, MIL is the natural paradigm to address this problem, which has also been utilized for web mining [3], graph mining [4], [5], sentiment analysis [6], etc.

Recent advances in deep learning have prompted many MIL methods using deep neural networks (DNNs) [1], [7], [8], [9], which usually outperform shallow methods. Many DNN-based MIL methods such as AbMIL [1] and LbMIL [10] assume that instances in any bag are independent and identically distributed (IID). However, some studies like [11], [12] point out that this IID assumption is unreliable in many cases because it ignores the structural information encoded by the instances. Therefore, graph neural networks (GNNs) [13], [14], [15], [16] are widely applied in MIL to learn intra-bag topology and yield structure-preserving representations, where instances serve as nodes and edges are correlations among them. Existing GNN-based MIL methods usually require filtering low-confidence edges among instances to obtain reliable bag graphs. As shown in Fig. 2, the higher filtering threshold represents that fewer but more robust edges are preserved, while potentially losing some meaningful but modestly reliable information. Conversely, a lower filtering threshold may introduce too many meaningless edges in a bag. Given that GNNs rely on graph structures for aggregation, such uncertain filtration may result in unstable results. As shown in Fig. 3, more aggregation iterations (reflected in the number of GNN layers) prompt nodes to adopt farther-hop features when fewer edges are retained. With a moderate threshold reduction, edge density increases, and more GNN layers may cause nodes to fuse information from too many other nodes, increasing the risk of over-smoothing [17], [18]. As factors influencing graph

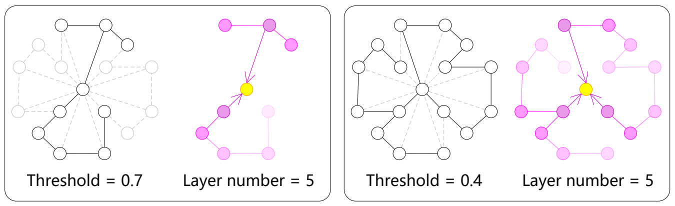


Fig. 3: A toy example of performing the same aggregation with different edge filtering on a bag sample. Yellow dots represent target nodes, while darker to lighter purple dots are neighbors from near to far hops.

structures and GNN architectures are numerous and correlated, manually adjusting them in an asynchronous manner is tedious and inflexible.

To deal with the above challenges, we propose a *Reinforced GNN framework for MIL* (RGMIL), pioneering the application of multi-agent deep reinforcement learning (MADRL) in MIL. MADRL enables the flexible definition or extension of factors that influence bag graphs and GNNs. We target the adjustment of the filtering threshold that influences the edge density within the bag graph and the number of model layers for aggregation to implement RGMIL. The training process of RGMIL is then modeled as a fully cooperative Markov game (MG) [19]. First, we divide the training set into equal-sized blocks, one of which serves as the validation set, and the others are used to construct the MG state space. Then, two agents search for edge filtering thresholds and GNN layers, both of which have discrete action spaces. At a time step, each agent picks out an action according to the corresponding partial observations of the current global state, thereby guiding the construction of bag graphs and GNN layers. Since the purpose of GNNs is to improve representation learning, we regard the difference in adjacent performance on the validation set as the current reward. In other words, agents receive a positive reward if the model trained with the current action combination performs better on the validation data than the previous one, and vice versa. Finally, we introduce a novel heuristic state transition function to determine the next global state based on current actions. When the game reaches a Nash equilibrium, test samples are processed using the final optimal actions, where instances retain the best one-hop neighbors and have the best number of aggregation iterations. In this way, we achieve automatic synchronous control over both structure and architecture.

For practical deployment, we select and improve a MADRL algorithm, i.e., value decomposition network (VDN) [20], that uses reverse decomposition to measure agent contributions and correlations. We utilize the graph attention network (GAT) [21] and design a parameter-sharing mechanism to boost efficiency. In addition, explainability is also significant to advance MIL in engineering applications because we often needs to understand the motivations behind decisions. Taking the colon cancer task as an example again, after accurately identifying the malignant images of colon tissue, we also need to quantitatively calculate the impact of different ROI instances on the bag classification results. Therefore, we leverage attention-based pooling to yield bag-level representations.

The main contributions are summarized as follows:

• This is an initial endeavor to introduce MADRL into MIL, enabling automated and synchronized control of bag structures and GNN architectures, where the factors to be controlled can be flexibly defined and expanded.

• We use the edge threshold and the number of GNN layers as factor cases to construct RGMIL, exploring the correlations between edge density and aggregation range, which have been overlooked in previous MIL studies.

• Extensive experiment results demonstrate that the RGMIL outperforms the state-of-the-art baselines, especially on benchmark and text datasets, with an average accuracy improvement of 2-3%. Besides, RGMIL achieves superior result explainability than existing methods.

This paper is organized as: Sec. II introduces related works. Sec. III describes preliminaries. Sec. IV-V present the methodology and experiments. Sec.VI summarizes this work.

II. Related Works

In this section, we introduce two categories of related works. They are graph neural network (GNN)-based multiple instance learning (MIL) algorithms and graph neural architecture search (GNAS) with reinforcement learning (RL).

A. GNN-based MIL

MIL [22], [23], [24] is receiving broad recognition due to its ability to process ambiguous labels in real-world applications. For example, in the context of Internet engineering, [3] regards the web index recommendation issue as a MIL problem. In this background, the index webpages are treated as bags while their linked webpages are instances without clear annotations. Since effectively capturing correlations among instances is beneficial for GNNs to learn better bag representations and achieve better bag classification, a large number of GNN-based MIL methods have emerged. For example, [13] proposed GNN-MIL, which treats instances as non-IID and leverages traditional GNNs and differentiable pooling to implement MIL. [14] designed GNN-RLHI to incorporate the global information of bag graphs into the instance features. [15] proposed DGC-MIL, which includes multiple modules like image processing, instance-level feature extraction and selection. Through convolution aggregation and feature selection, DGC-MIL achieves outstanding performance in bag-level classification. Different from previous methods of constructing bag graphs with feature similarities, [25] designed a Bayesian GNN framework BGNN-MIL that infers possible inter-bag correlations via bag features. [26] designed NAGCN, which divides the instances in each bag graph into sub-graphs based on the codebook and then aggregates the features within and between the sub-graphs to learn local and global patterns. Though these methods have verified the effectiveness of GNNs in capturing correlations among instances, existing models are either designed for specific MIL contexts or ignore the balance between edge density of bag graphs and aggregation iterations.

B. GNAS with RL

An easy strategy to avoid exploring the interaction between edge density and aggregation iterations is to deepen the GNN

TABLE I: Definitions of all notations.

Notation	Definition
$\mathcal{B}$	The set of bag samples
$\mathcal{G}$	The set of bag graphs corresponding to $\mathcal{B}$
$\mathcal{Y}$	The set of bag-level labels corresponding to $\mathcal{G}$
$\mathcal{M}$	The seven-tuple of the Markov game
$\mathcal{S}$	The state space of $\mathcal{M}$
$\mathcal{O}$	The observation space of $\mathcal{M}$
$\mathcal{A}$	The action space of $\mathcal{M}$
$\mathcal{L}$	The training loss of the agents or GNN model
N	The total number of bag samples in $\mathcal{B}$
M	The total number of instances in a bag sample
L	The total number of GNN layers
T	The total number of time steps
I	The total number of agents
D	The dimension of feature representations
$\mathbf{A}$	The adjacency matrices corresponding to $\mathcal{G}$
$\mathbf{F}$	The instance feature matrices corresponding to $\mathcal{G}$
$\mathbf{E}$	The bag graph feature matrix corresponding to $\mathcal{G}$
$\mathbf{Z}$	The feature transformation matrices
$\mathbf{C}$	The importance coefficient matrices
$i; j; k; l; t$	These notations represent index variables
$s; o; a; r$	A state; An observation; An action; A reward
v	A feature vector of the attention mechanism
γ	The discounted coefficient
α	The learning rate of the agent function
μ	The window size of action (or reward) records
λ	The reward threshold of the termination condition
$\&; \%$	The logical and; The remainder operation
$\oplus$	The concatenation operation
$\lVert\cdot\rVert$	The norm function of the matrix
$\sigma(\cdot)$	The activation function
$\pi(\cdot)$	The agent state-action function (or table) of $\mathcal{M}$
$RWD(\cdot)$	The reward calculation function of $\mathcal{M}$
$TRN(\cdot)$	The Markov state transition function of $\mathcal{M}$
$AGG(\cdot)$	The feature aggregation function
$POL(\cdot)$	The node feature pooling function
$EVL(\cdot)$	The classification performance evaluation function

using skip connections [27], [28]. This strategy enables a large number of GNN layers to be clustered without over-smoothing problems that may occur due to higher edge density. However, many studies on GNAS [29], [30], [31], [32] verify that usually not all neighbors are precious for feature aggregation. Filtering irrelevant neighbor nodes and corresponding edges effectively prevent goal nodes from aggregating interference information. On the other hand, some studies like [33], [34] show that even if layer-fixed deep GNNs via skip-connections are suitable for many applications, further searching the best number of feature aggregations will still improve model performance. Moreover, the vast range of possible choices makes manually finding the best layer settings even more infeasible.

Recently, owing to the advances of RL [35], [36], [37], some studies use RL to further break the performance boundaries of GNN models. Both the edge filtering threshold and the number of GNN layers are critical factors to be searched. For example, studies like [29], [30], [31], [32] first sort the neighbors based on diverse importance calculation methods, and then determine the best number of neighbor nodes through one or more single-agent RL. Besides, works such as [33], [34] also utilize single-agent RL to implement GNN layer number search for different nodes or graph samples. For multiple-agent RL (MARL)-based GNAS works like [38], [39], [40], they usually care more about aggregation functions, activation functions, etc. None of them simultaneously search for the filtering threshold and number of layers, both of which are significant graph neural architectural factors. Besides, there is currently no study introducing GNAS into GNN-based MIL, limiting the ability of GNNs to explore instance-level dependencies, where initial bag graphs are often fully connected. Therefore, RGMIL, which fills the above gap, is meaningful for both GNN-based MIL and GNAS.

III. Preliminaries

The preliminary knowledge consists of two parts. Sec. III-A-III-B are related formulations of multiple instance learning and MIL with graph neural networks. Sec. III-C-III-D describe the Markov game and deep Q-learning. The key notations involved in this work are summarized in Table I.

A. Multiple Instance Learning

We can formulate multiple instance learning (MIL) as a kind of supervised learning with bags of instances as input and bag labels as targets. Given the bag set $\mathcal{B} = \{\mathcal{B}_i \mid i = 1, \dots, N\}$, a bag contains multiple instances $\mathcal{B}_i = \{\mathcal{B}_{i,j} \mid j = 1, \dots, M\}$, N indicates the total number of bags and M is the number of instances in a bag (M is usually different for different bags). There is a two-class bag-level label $\mathcal{Y}_i = \max(\mathcal{Y}_{i,1}, \dots, \mathcal{Y}_{i,M})$ associated with the bag $\mathcal{B}_i$, where $\mathcal{Y}_{i,j} \in \{0, 1\}$ is the assumed label of the instance $\mathcal{B}_{i,j}$. Even though few instance labels may be available in the training set, MIL assumes that they are all unknown during training. Thus, MIL aims to learn a mapping function from bags $\mathcal{B}$ to their labels $\mathcal{Y} = \{\mathcal{Y}_i \mid i = 1, \dots, N\}$, i.e., $\mathcal{B} \rightarrow \mathcal{Y}$.

B. MIL with Graph Neural Networks

For MIL with graph neural networks (GNNs), it first needs to convert all bags $\mathcal{B}$ into a graph set $\mathcal{G} = \{\mathcal{G}_i \mid i = 1, \dots, N\}$, where a bag $\mathcal{B}_i$ corresponds to a bag graph $\mathcal{G}_i = (\mathbf{A}_i, \mathbf{F}_i)$ and an instance $\mathcal{B}_{i,j}$ can be represented as a node. Each adjacency matrix $\mathbf{A}_i \in \mathbb{R}^{M \times M}$ is constructed from original node features and the edge filtering threshold, entries of which indicate one-hop neighbor information. $\mathbf{F}_i \in \mathbb{R}^{M \times D}$ represents the instance node feature matrix with dimension D. Then, the L-layer GNN is used to propagate node feature information. The aggregation process of the GNN at the l-th layer for the i-th graph $\mathcal{G}_i$ can be expressed as follows:

$$\mathbf{F}_i^l = \sigma(AGG^l(\mathbf{A}_i, \mathbf{F}_i^{(l-1)})), \tag{1}$$

where $AGG^l(\cdot)$ represents the aggregation function at the l-th layer like convolution and attention. $\sigma(\cdot)$ denotes the activation function like $ReLU$ and $Tanh$. Then, $\mathbf{F}_i^l$ is the D-dimensional node-level feature matrix at the l-th layer. Next, a node feature pooling function $POL(\cdot)$ is applied on the $\mathbf{F}_i^L$ at the last layer to get the final graph-level feature matrix $\mathbf{E}(i) \in \mathbb{R}^{1 \times D}$ of $\mathcal{G}_i$:

$$\mathbf{E}(i) = POL(\{\mathbf{F}_i^L(j) \mid j = 1, \dots, M\}), \tag{2}$$

where $\mathbf{F}_i^L(j) \in \mathbb{R}^{1 \times D}$ is the final feature vector of the instance node $\mathcal{B}_{i,j}$. Finally, $\mathbf{E}(i)$ is transferred to a bag graph classifier for binary classification. In GNN-based MIL, the mapping will become $\mathcal{B} \rightarrow \mathcal{G} \rightarrow \mathcal{Y}$.

C. Markov Game

As the basis of multi-agent reinforcement learning (MARL), the Markov game (MG) is extended from the Markov decision process (MDP). Specifically, a MG has multiple agents whose actions jointly influence rewards and state transitions. Existing MGs are fully or partially observable according to whether all agents completely obtain the global state information, whereas the latter MGs are more prevalent in formulating. The partially observable MG can always be abstracted as a seven-tuple, i.e., $\mathcal{M} =< \mathcal{S}, \mathcal{O}_i, \mathcal{A}_i, \pi_i(\cdot), RED_i(\cdot), TRN(\cdot), \gamma >$, where $\mathcal{S}$ is the global state space of MG, $\mathcal{A}_i$ indicates the action space of the i-th agent. At each time step $t \in [1, T]$, each agent chooses an action $a_i^t \in \mathcal{A}_i$ with its exclusive state-action function $\pi_i(\cdot)$. All actions form the joint action $a^{*t} = (a_i^t | i \in [1, I])$, where I is the total number of agents and a^{*t} belongs to a joint action space $\mathcal{A}^* = \times_i \mathcal{A}_i$. More concretely, each agent will obtain an independent partial observation $o_i^t \in \mathcal{O}_i$ from the global state, i.e., $\mathcal{S} \to \mathcal{O}_i$. Thus, the action prediction of $\pi_i(\cdot)$ is expressed as $\mathcal{S} \to \mathcal{O}_i \to \mathcal{A}_i$. Moreover, each agent obtains an immediate reward r_i^t using its reward function $RED_i(\cdot)$. Since this study focuses on the cooperative games, all agents receive the same reward $r^{*t} = r_1^t = \ldots = r_I^t$. Such kind of game is also known as the decentralized partially observable MDP (Dec-POMDP) [41], aiming to maximize a cumulative return $\sum_{t=1}^{T} \gamma^{(t-1)} r^{*t}$, where γ represents the discount coefficient that controls future rewards. After that, the state transition function $TRN(\cdot)$ maps the current state s^t along with joint action a^{*t} to the next state $s^{(t+1)}$, i.e., $\mathcal{S} \times \mathcal{A}^* \to \mathcal{S}$.

D. Deep Q-learning

As a basic algorithm of value-based reinforcement learning, Q-learning [42] is a good fit for achieving one-agent sequential decision systems. Q-learning contains a state-action table $\pi(\cdot)$, which records the Q-values associated with all possible actions in diverse states. After the initialization, the agent continuously interacts with an environment and updates $\pi(\cdot)$ via the Bellman equation until convergence. The update process of the $\pi(\cdot)$ can be expressed as follows:

$$\begin{aligned} & x = x + \alpha[r^t + \gamma \max_a \pi(s^{(t+1)}, a) - x], \\ & s.t.\ x = \pi(s^t, a^t), \end{aligned} \tag{3}$$

where $\pi(s^t, a^t)$ is a predicted Q-value and the expected reward for selecting an action a^t in state s^t. r^t indicates the immediate reward at time step t, while $\max_a \pi(s^{(t+1)}, a)$ is the maximum Q-value of the next state $s^{(t+1)}$. Besides, α is the learning rate of the agent state-action table $\pi(\cdot)$.

Since the state spaces of a lot of environments are infinite, it is infeasible to record all values of state-action pairs in a table. Influenced by deep learning, many works have introduced deep neural networks (DNNs) to approximate returns, among which deep Q-learning (DQN) [43] is a direct extension of traditional Q-learning. DQN employs DNNs to construct the action-value function π (a.k.a., Q-function), which maps each state vector to a Q-value vector $\pi(s) \in \mathbb{R}^{1 \times |\mathcal{A}|}$, where $|\mathcal{A}|$ denotes the size of the action space $\mathcal{A}$. Moreover, DQN applies experience replay and target network techniques to update the function $\pi(\cdot)$. For example, given the experience recorded at past time step t in a tuple form $< s^t, a^t, r^t, a^{(t+1)} >$, the temporal-difference loss of π can be calculated as follows:

$$\begin{aligned} & \mathcal{L}_\pi = \mathbb{E}_{<s,a,r,s'>}[(\overline{\pi}(s^t, a^t) - \pi(s^t, a^t))^2], \\ & s.t.\ \overline{\pi}(s^t, a^t) = r^t + \gamma \max_a \overline{\pi}(s^{(t+1)}, a), \end{aligned} \tag{4}$$

where $\pi(\cdot)$ represents the evaluation network applied to predict the Q-value for state s^t and action a^t. $\overline{\pi}(\cdot)$ is a target network with same architectures as $\pi(\cdot)$. Only $\pi(\cdot)$ is optimized and its trained parameters are periodically copied to $\overline{\pi}(\cdot)$. The training stability of $\pi(\cdot)$ is superb because the target Q-values are stable when $\overline{\pi}$ is not updated. To trade off the probability of exploring new actions, DQN also applies a ϵ-greedy algorithm. Hence, it does not always choose the action corresponding to the largest entry in $\pi(s)$, which can be expressed as follows:

$$a = \begin{cases} \text{random action}, & w.p.\ \epsilon \\ \text{argmax}_a \pi(s, a), & w.p.\ 1 - \epsilon \end{cases}, \tag{5}$$

where ϵ denotes the probability of randomly selecting an action (a.k.a., exploration), while $(1-\epsilon)$ means choosing the currently π-based optimal action (a.k.a., exploitation). In doing so, DQN avoids falling into the exploration-exploitation dilemma in RL learning tasks [42], steers clear of local optima, and facilitates the discovery of superior π functions.

IV. Methodology

This section introduces the technical details of the RGMIL, including: **1)** observation generation and interaction techniques that improve game fairness (Sec. IV-A), **2)** action selection and guidance techniques that improve GNN efficiency (Sec. IV-B), **3)** a reward calculation technique that improves game stability (Sec. IV-C), **4)** state transition and termination techniques that ensure game convergence (Sec. IV-D), **5)** multi-agent training (Sec. IV-E). An overview of RGMIL over a time step is shown in Fig. 4, whose left sub-figure corresponds to Sec. IV-A-IV-D and right sub-figure corresponds to Sec. IV-E, respectively.

A. Observation Generation and Interaction

In particular, we model the training process of RGMIL as a cooperative Markov game (MG) involving two agents, which are used to search for the best edge filtering threshold and the number of GNN layers, respectively. Specifically, we leverage an improved value decomposition network (VDN) [20], which is a multi-agent extension of DQN to achieve the MG. First, we divide the training set into multiple equal-sized blocks, where a block is the validation set and the remaining blocks are applied as a state space $\mathcal{S}$. Before the first time step, a training block is randomly selected as the global state. Since the choice of edge filtering threshold is usually related to topological information, we then specify the structural features of the bag graphs in the current state as the observation of the first agent. Moreover, we establish initial edges of instance nodes through their pairwise similarities. Taking the i-th bag $\mathcal{B}_i$ that belongs to the current block as an example, its bag graph $\mathcal{G}_i$ can be abstracted as an adjacency matrix $\mathbf{A}_i$ along with a feature matrix $\mathbf{F}_i$. Given the initial matrix $\mathbf{F}_i^0$, the initial adjacency matrix $\mathbf{A}_i$ is calculated as follows:

$$\mathbf{A}_i(j, j') = \left\| \mathbf{F}_i^0(j) - \mathbf{F}_i^0(j') \right\|_2, \tag{6}$$

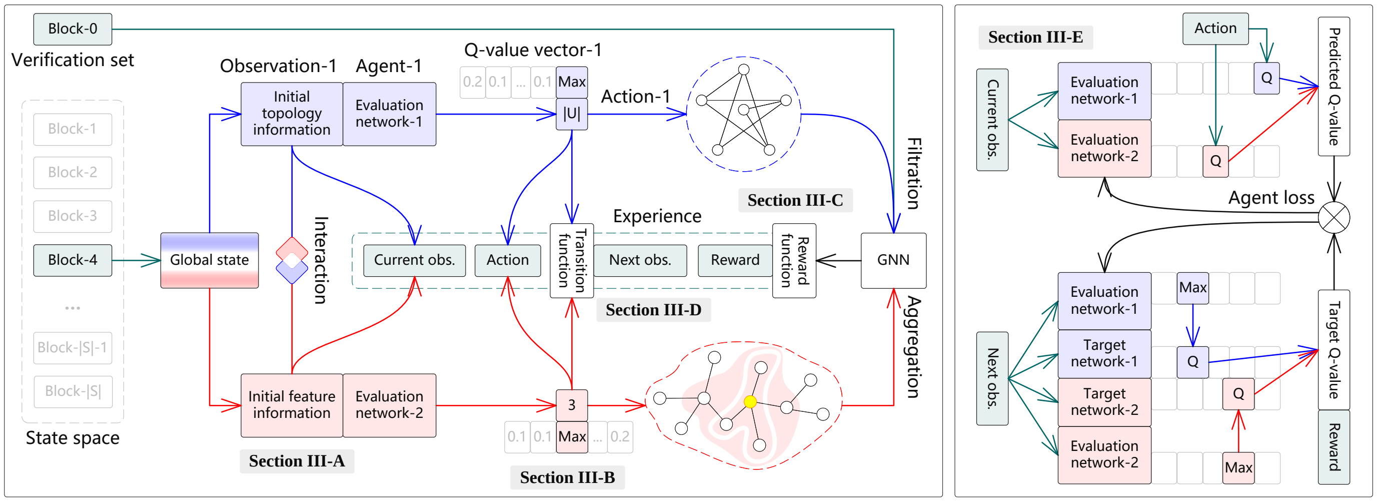


Fig. 4: An overview of RGMIL. The left and right sub-figures respectively show the process of experience collection and agent optimization. At each time step, the initial observations are derived from the current block. Then, the observations are used as inputs to the agents. Each agent chooses an action based on the corresponding observation. Next, it builds reliable bag graphs and feed them into a customized GNN. After GNN training, assess performance through the action combination to determine the current reward. Finally, the transition function with actions as input is used to obtain the next observations. The quadruple is now recorded in memory. Upon reaching a sufficient quadruple count, optimization of the agents is executed based on VDN.

where $\|\cdot\|_2$ is the 2-norm of a matrix, and $\mathbf{A}_i(j,j')$ encodes an Euclidean distance between the j-th and j'-th instance nodes. Thus, the observation of the first agent is calculated as follows:

$$o_1(d) = \frac{1}{N_d}\sum_{i=1}^{N_d}\frac{\exp(-\mathbf{A}_i)}{M_i^2}, \quad s.t.\ M_i = d,\ \ d \in [1, \max_i M_i], \tag{7}$$

where $o_1(d)$ indicates the d-th entry of the vector o_1, and N_d is the number of bags of the current block, and the number of instances in it is equal to d. M_i is the number of instance nodes of the bag graph $\mathcal{G}_i$. Since the number of GNN layers controls feature aggregation iterations, we then acquire the observation for the second agent from the initial node features $\mathbf{F}_i^0$, which is calculated as follows:

$$o_2 = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{M_i}\sum_{j=1}^{M_i}\mathbf{F}_i^0(j)\right), \tag{8}$$

where $\mathbf{F}_i^0(j) \in \mathbb{R}^{1\times D}$ is the feature vector of the j-th instance node. N is the total number of bag graphs of the current block.

To further explore the underlying correlations between edge density and aggregation iterations, we introduce the following observation information interaction:

$$\begin{aligned} o_1 &= o_1 \oplus \sigma((o_1 \oplus o_2)(o_2 \oplus o_1)^{\mathrm{T}} o_2), \\ o_2 &= o_2 \oplus \sigma((o_1 \oplus o_2)(o_2 \oplus o_1)^{\mathrm{T}} o_1), \end{aligned} \tag{9}$$

where $\oplus(\cdot)$ is a concatenation operation for vectors. After this operation, the observations o_1 and o_2 have the same dimension and both encode information from the other. RGMIL alleviates the unequal game challenge in the MG that may be caused by variations in the feature dimension or amount of information of observations. Moreover, to improve the efficiency of this part, RGMIL calculates and records these initial adjacency matrices with observations for each data block only once.

B. Action Selection and Guidance

Second, each agent maps its input observation vector o_i to a Q-value vector $\pi_i(o_i) \in \mathbb{R}^{1\times|\mathcal{A}_i|}$ and yields an action a_i based on the largest Q-value entry or randomly (Eq. (5)). Particularly, the first threshold action $a_1 \in [0,1]$ is a fractional value, while the second layer action a_2 is an integer. Guided by a_1, a more reliable form of the adjacency matrix $\mathbf{A}_i$ can be expressed as:

$$\mathbf{A}_i(j,j') = \begin{cases} 1, & \exp(-\mathbf{A}_i(j,j')) \geq a_1 \\ 0, & \exp(-\mathbf{A}_i(j,j')) < a_1 \end{cases}. \tag{10}$$

After that, RGMIL builds a customized GNN guided by the second action a_2. Taking a graph attention network (GAT) [21] as an example, whose aggregation process of node features can be expressed as follows:

$$\begin{aligned} &\mathbf{C}_i^{(l-1)}(j,j') = v(\mathbf{F}_i^{(l-1)}(j)\mathbf{Z}^{(l-1)} \oplus \mathbf{F}_i^{(l-1)}(j')\mathbf{Z}^{(l-1)})^{\mathrm{T}}, \\ &\mathbf{F}_i^{l}(j) = \sigma(\sum_{j'} x\mathbf{F}_i^{(l-1)}(j')\mathbf{Z}^{(l-1)}), \\ &s.t.\ x = \mathrm{softmax}(\sigma(\mathbf{C}_i^{(l-1)}(j,j')))\ \&\ \mathbf{A}_i(j,j') = 1, \end{aligned} \tag{11}$$

where $l \in [1, a_2]$ represents running a_2 iterative aggregations, $\mathbf{F}_i^l(j)$ is a D^l-dimensional feature vector of the j-th node $\mathcal{B}_{i,j}$ at the l-th GNN layer. $\mathbf{Z}^{(l-1)}$ means the feature transformation matrix with $D^{(l-1)} \times D^l$ shape. Besides, $v \in \mathbb{R}^{1\times 2D^l}$ indicates the feature vector of the self-attention mechanism, $\mathbf{C}_i(j,j')$ is the importance coefficient of a neighbor $\mathcal{B}_{i,j'}$ to its target $\mathcal{B}_{i,j}$, whose normalized form needs to be obtained with softmax [21] function, and $\&$ denotes the logic and operation. Based on the attention-based node feature pooling function, RGMIL obtains the final bag graph feature matrix $\mathbf{E} \in \mathbb{R}^{N\times D^{a_2}}$ of the current training block, which can be calculated as follows:

$$\begin{aligned} &\mathbf{C}_i^{a_2}(j) = \mathrm{softmax}(v'(\mathbf{F}_i^{a_2}(j)\mathbf{Z}')^{\mathrm{T}}), \\ &\mathbf{E}(i) = \sum_{j=1}^{M_i}\mathbf{C}_i^{a_2}(j)\mathbf{F}_i^{a_2}(j), \end{aligned} \tag{12}$$

where $\mathbf{F}_i^{a_2}(j)$ indicates the D^{a_2}-dimensional feature vector of the j-th node at the last layer, whose corresponding importance coefficient is $\mathbf{C}_i^{a_2}(j)$. v' and $\mathbf{Z}'$ are the query vector and linear transformation matrix of the attention mechanism respectively. $\mathbf{E}(i)$ is the i-th row of $\mathbf{E}$, also the feature vector of graph $\mathcal{G}_i$. Along with bag graph labels $\mathcal{Y}$, the GNN loss is expressed as:

$$\mathcal{L}_{GNN} = -\sum_{i=1}^{N} \overline{\mathcal{Y}}_i \log(\mathbf{E}(i)\overline{\mathbf{Z}})^{\mathrm{T}}, \tag{13}$$

where $\overline{\mathbf{Z}}$ is the graph classifier, $\overline{\mathcal{Y}}_i$ is the label vector extended by the label $\mathcal{Y}_i \in \mathcal{Y}$ of the bag graph $\mathcal{G}_i$. The backpropagation algorithm is then applied to optimize the GNN.

In order to improve the GNN efficiency, RGMIL introduces a parameter-sharing mechanism in the GNN framework, whose number of layers is fixed to the maximum action value. In this way, RGMIL only needs to use and fine-tune the first a_2^t layers of the GNN framework at each time step t. The RGMIL avoids consuming a lot of time and space resources for rebuilding and retraining new GNNs every time. Besides, RGMIL records the number of occurrences of each action combination. If (a_1, a_2) is recorded more than the predefined number, the current GNN training process will be omitted.

C. Reward Calculation

Third, after obtaining the action combination and optimizing the GNN, RGMIL will evaluate the combination by computing an immediate reward on the validation data block. Specifically, in fully cooperative MG modeled by RGMIL, all agents have the same joint reward (a.k.a., team reward). Since GNN models aim to improve representation learning, the reward is computed based on the difference in bag graph classification performance at adjacent time steps. Similarly, RGMIL processes validation samples according to action a_1 and feeds them into the model with a_2 layers. The reward function $RWD(\cdot)$ is expressed as:

$$\begin{aligned} &r^* = RWD(a_1, a_2) = EVL(t) - \frac{1}{\mu}\sum_{t'}^{t} EVL(t'), \\ &s.t.\ t' = t - \mu + 1, \end{aligned} \tag{14}$$

where t indicates the current step, $EVL(\cdot)$ is the classification performance evaluation function, μ represents the window size of historical records. The RGMIL averages μ historical records to ensure the reliability of the reward as well as the stability of the game. In particular, μ also serves as the predefined number of records for action combinations.

D. State Transition and Termination

Finally, RGMIL introduces a novel heuristic state transition function to obtain the next global state and observations. More specifically, inspired by the previous work [34], which treats an action as a hop range to guide the transition, RGMIL calculates the block index of the next global state according to the current action combination. Considering a_1 and a_2 belong to decimals and integers, respectively, RGMIL treats them as different state transition dependencies. The data block index k corresponding to the next state is calculated as follows:

$$\begin{aligned} &k = ((\text{round}(a_1) + a_2)\ \%\ |\mathcal{S}|) + 1, \\ &s.t.\ k \in [1, |\mathcal{S}|]\ \&\ \text{round}(a_1) \in \{0, 1\}\ \&\ a_2 > |\mathcal{S}|, \end{aligned} \tag{15}$$

where round$(\cdot)$ denotes the mathematical rounding function, % is the remainder operation. The action a_2 is larger to ensure the coverage of the training blocks, and round(a_1) then provides a small offset to increase the variation. Since k is mainly affected by a_2, RGMIL avoids the game non-convergence problem that may be caused by the simultaneous violent fluctuations of the two actions in the later stage. After this, RGMIL will construct the next observations through Sec. IV-A. Thus, the experience $<(o_1, o_2), (a_1, a_2), r^*, (o_1', o_2')>$ over one time step is stored.

The transition will not terminate until either reaching the last time step T or, alternatively, meeting the following termination condition at an earlier intermediate step t (where $t \leq T$):

$$|\frac{1}{\mu}\sum_{t'}^{t} r^{*t'}| < \lambda,\ \ s.t.\ t' = t - \mu + 1, \tag{16}$$

where the inequality symbol indicates that the mean of the past μ rewards has not improved beyond a predefined threshold, λ. $r^{*t'}$ is the joint reward at the past time step t'.

E. Multi-agent Training

When the number of historical experiences is greater than μ and the game is not over, RGMIL needs to train the two agents through experience replay after completing the aforementioned process at a time step. Since the VDN justifies that the joint Q-function can be decomposed into different Q-functions across agents, it updates the agents in a value-decomposition manner. Therefore, RGMIL decomposes the joint Q-value to each agent through backpropagation. Both agents will be actively working towards a common goal by measuring their contribution to the joint Q-value. Given the experience tuple collected at time step t, $<(o_1^t, o_2^t), (a_1^t, a_2^t), r^{*t}, (o_1^{(t+1)}, o_2^{(t+1)})>$, the joint loss of the agents can be calculated as follows:

$$\mathcal{L}_{\pi^*} = \mathbb{E}_{<s,a,r,s'>}[(\overline{\pi}^*(o^{*t}, a^{*t}) - \pi^*(o^{*t}, a^{*t}))^2], \tag{17}$$

where $\pi^*(o^{*t}, a^{*t})$ indicates the predicted joint Q-value, which can be expressed as follows:

$$\pi^*(o^{*t}, a^{*t}) \approx \pi_1(o_1^t, a_1^t) + \pi_2(o_2^t, a_2^t), \tag{18}$$

where $\pi_1(\cdot)$ and $\pi_2(\cdot)$ are evaluation networks (Q-functions) of the first and second agents. $\overline{\pi}^*(o^{*t}, a^{*t})$ denotes the target joint Q-value, which can be formulated in a summation form similar to Eq. (18), where each additive component, i.e., $\overline{\pi}_i(o_i^t, a_i^t)$ can be expressed as follows:

$$\overline{\pi}_i(o_i^t, a_i^t) = r^{*t} + \gamma \text{max}_a \overline{\pi}_i(o_i^{(t+1)}, a), \tag{19}$$

where $\overline{\pi}_i(\cdot)$ is a target network of the i-th agent, which has the same architectures as the $\pi_i(\cdot)$. It is worth noting that RGMIL only trains the evaluation networks and copies their parameters to the target networks every μ time step.

To alleviate the possible over-estimation issue [43], RGMIL employs the traditional double DQN algorithm to calculate the target Q-value, which determines the action with the evaluation network and computes the Q-value with the target network, as shown in the right sub-figure of Fig. 4. Therefore, the Eq. (19) will be transformed as follows:

$$\overline{\pi}_i(o_i^t, a_i^t) = r^{*t} + \gamma \overline{\pi}_i(o_i^{(t+1)}, \text{argmax}_a \pi(o_i^{(t+1)}, a)). \tag{20}$$

Algorithm 1 RGMIL: Reinforced GNN Framework for MIL

```
Input: Bag samples $\mathcal{B}$ and their corresponding labels $\mathcal{Y}$
Output: The bag-level feature matrix $\mathbf{E}$ of $\mathcal{B}$
 1: $\{\mathbf{A}_i | i \in [1, N], \mathbb{N}^+\} \leftarrow$ Eq. (6) // All adjacency matrices
 2: // Observations for all training blocks
 3: $\{(o_1, o_2) | i \in [1, |\mathcal{S}|], \mathbb{N}^+\} \leftarrow$ Eqs. (7, 8, 9)
 4: for $t = 1$ to $T$ do // Operations over a time step
 5:     $(a_1, a_2) \leftarrow$ Eq. (5) // The current action combination
 6:     // More reliable adjacency matrices
 7:     $\{\mathbf{A}_i | i \in [1, N^t], \mathbb{N}^+\} \leftarrow$ Eq. (10)
 8:     for $l = 1$ to $a_2$ do // Instance-level feature matrices
 9:         $\{\mathbf{F}_i^l | i \in [1, N^t], \mathbb{N}^+\} \leftarrow$ Eq. (11)
10:     end for
11:     $\mathbf{E}^t \leftarrow$ Eq. (12) // The bag-level feature matrix
12:     $\mathcal{L}_{GNN} \leftarrow$ Eq. (13) // The loss of the GNN model
13:     $r^* \leftarrow$ Eq. (14) // The current joint reward
14:     $k \leftarrow$ Eq. (15) // The index of the next training block
15:     if the termination condition Eq. (16) then break
16:     end if
17:     $\mathcal{L}_{\pi^*} \leftarrow$ Eqs. (17, 18, 20) // The joint loss of the agents
18: end for
```

F. Computational Complexity

RGMIL improves MIL performance while still having good computational complexity. It first requires calculating N initial adjacency matrices of all samples, whose computational complexity is $\mathcal{O}(NM^2D)$. Then, it will construct observations for all data blocks, which takes $\mathcal{O}(|\mathcal{S}|DNM^2)+\mathcal{O}(|\mathcal{S}|NMD)+\mathcal{O}(|\mathcal{S}|D)$. During each time step, the choice of the current two actions takes $\mathcal{O}(D|\mathcal{A}|)$, the generation of N reliable adjacency matrices costs $\mathcal{O}(NM^2)$, the feature learning of samples takes $\mathcal{O}(NL(MD^2+M^2D)+N(MD+MD^2+D^2))$, both reward calculation and state transitions cost constant time, and the last optimization of two agents takes $\mathcal{O}(D|\mathcal{A}|)$. Hence, the overall complexity of RGMIL is $\mathcal{O}(TNLMD^2)$, which is acceptable. Due to the excellent performance of the RGMIL, we argue that it is worth sacrificing some time to augment GNN-based MIL. Furthermore, the computational burden mainly comes from the GNN-based model rather than multi-agent deep reinforcement learning (MADRL). Because the efficiency of GNNs has been a long-standing problem, high-performance GNNs can be used to replace GAT to further improve the performance of RGMIL, which is not the focus of this study. The algorithm process and notations are available in Alg. 1 and Table I.

V. EXPERIMENTS

This section first describes the experimental configurations, including MIL datasets, baselines, and implementation settings (Sec. V-A-V-B). Then, we depict and analyze the experimental results of bag-level classification (Sec. V-C-V-E).

A. Datasets and Baselines

The experiments apply three types of datasets, including five benchmark datasets, five news text datasets, and three medical image datasets, all of which are publicly available and popular in MIL works. Table II summarizes the statistics of all datasets. More details are described as follows:

TABLE II: Statistics for all datasets applied in the experiments. The right columns show the total number of bags (positive and negative), the total number of instances, and the dimensions of instance features. NEWS-GROUPS provides average statistics.

Dataset	Bag			Instance	Dimension
	Total	Positive	Negative		
MUSK1 [44]	92	47	45	476	166
MUSK2 [44]	102	39	63	6598	166
FOX [45]	200	100	100	1320	230
TIGER [45]	200	100	100	1220	230
ELEPHANT [45]	200	100	100	1391	230
NEWS-GROUPS [12]	100	50	50	3197	200
MESSIDOR [46]	1200	654	546	12352	687
UCSB-BREAST [2]	58	26	32	2002	708
COLON-CANCER [48]	100	51	49	22448	729

1) Benchmark Datasets: The first category covers five MIL benchmark datasets, where MUSK1 and MUSK2 [44] are both datasets for identifying whether a molecule has a musky smell or not. A molecule has different conformations. However, not all conformations can cause it to smell musky. For a molecule, it is labeled as positive if and only if it has at least one musky conformation. Therefore, a molecule can be regarded as a bag, and the conformations are instances. The remaining three, i.e., FOX, TIGER, and ELEPHANT [45] are animal detection tasks that treat an image as a bag and the marked ROIs in the image are instances. Images containing the target animal in ROIs are marked as positive, otherwise negative. All these datasets have predefined features and a small number of bags and instances.

2) Text Datasets: The second category is five text datasets. Concretely, we select five datasets from NEWS-GROUPS [12] that contains 20 corpus from the same interval. Each of the five datasets consists of 100 bags, with the same number of positive and negative samples. In a dataset, a news article is an instance that may come from 20 different topics. Besides, each instance is represented by the top 200 term frequency-inverse document frequency (TF-IDF). A bag is an article group containing about 40 instances. A positive bag has about 3% of positive instances belonging to the target topic. A consistent purpose across these tasks is to correctly judge the category of a given article group (bag) or whether it has articles (instances) with the target topic.

3) Image Datasets: The third category covers three clinical medical image datasets. The first dataset, named MESSIDOR, was originally collected by [46], which consists of 1200 fundus images from 654 diabetic patients (positive) with 546 healthy patients (negative). Here we follow the processing steps in [47] to model the classification of fundus images as a MIL problem, where an image (bag) is rescaled to the fixed shape (700×700 pixels) and contains multiple ROIs (instances) of uniform size (135×135). For more information about the image processing, such as ROI selection or feature extraction, please refer to [47]. The second dataset is UCSB-BREAST [2] which comprises 58 tissue images of 896×768 pixels yielded from 26 breast cancer patients and 32 benign controls, where each image is regarded as a bag whose ROIs with 7×7 size are extracted as instances. A malignant image consists of at least one malignant ROI, and a benign image comprises all benign ROIs. The third dataset is COLON-CANCER [48] which consists of 100 images of colon tissue of size 500×500. Each image contains a number of ROIs of 27×27 in four categories, namely epithelial, inflammatory,

TABLE III: Classification results (average accuracy ± standard deviation) on benchmark datasets. The best results of all methods are shown in bold, while the best results of all baselines are in italics.

Method		Dataset					Mean[1]
		MUSK1 [44]	MUSK2 [44]	FOX [45]	TIGER [45]	ELEPHANT [45]	
Shallow Baseline	mi-SVM [45]	0.874 ± N/A	0.836 ± N/A	0.582 ± N/A	0.784 ± N/A	0.822 ± N/A	0.780 ± 0.103
	MI-SVM [45]	0.779 ± N/A	0.843 ± N/A	0.578 ± N/A	0.840 ± N/A	0.843 ± N/A	0.777 ± 0.102
	MI-Kernel [49]	0.880 ± 0.031	0.893 ± 0.015	0.603 ± 0.028	0.842 ± 0.010	0.843 ± 0.016	0.812 ± 0.107
	EM-DD [50]	0.849 ± 0.044	0.869 ± 0.048	0.609 ± 0.045	0.730 ± 0.043	0.771 ± 0.043	0.766 ± 0.093
	mi-Graph [12]	0.889 ± 0.033	0.903 ± 0.039	0.620 ± 0.044	0.860 ± 0.037	0.869 ± 0.035	0.828 ± 0.105
	miVLAD [51]	0.871 ± 0.043	0.872 ± 0.042	0.620 ± 0.044	0.811 ± 0.039	0.850 ± 0.036	0.805 ± 0.095
	miFV [51]	0.909 ± 0.040	0.884 ± 0.042	0.621 ± 0.049	0.813 ± 0.037	0.852 ± 0.036	0.816 ± 0.103
DNN-based Baseline	mi-Net [9]	0.889 ± 0.039	0.858 ± 0.049	0.613 ± 0.035	0.824 ± 0.034	0.858 ± 0.037	0.808 ± 0.100
	MI-Net [9]	0.887 ± 0.041	0.859 ± 0.046	0.622 ± 0.038	0.830 ± 0.032	0.862 ± 0.034	0.812 ± 0.097
	MI-Net with DS [9]	0.894 ± 0.042	0.874 ± 0.043	0.630 ± 0.037	0.845 ± 0.039	0.872 ± 0.032	0.823 ± 0.098
	MI-Net with RC [9]	0.898 ± 0.043	0.873 ± 0.044	0.619 ± 0.047	0.836 ± 0.037	0.857 ± 0.040	0.817 ± 0.101
	AbMIL [1]	0.892 ± 0.040	0.858 ± 0.048	0.615 ± 0.043	0.839 ± 0.022	0.868 ± 0.022	0.814 ± 0.101
	AbMIL-Gated [1]	0.900 ± 0.050	0.863 ± 0.042	0.603 ± 0.029	0.845 ± 0.018	0.857 ± 0.027	0.814 ± 0.107
	MIVAE [52]	0.904 ± 0.050	0.890 ± 0.062	0.626 ± 0.055	0.850 ± 0.051	0.870 ± 0.064	0.828 ± 0.103
	ARP-MINN [53]	0.910 ± 0.038	0.891 ± 0.041	0.622 ± 0.026	0.858 ± 0.020	0.879 ± 0.024	0.832 ± 0.106
GNN-based Baseline	GNN-MIL [13]	0.917 ± 0.048	0.892 ± 0.011	0.679 ± 0.007	0.876 ± 0.015	0.903 ± 0.010	0.853 ± 0.088
	GNN-RLHI [14]	0.924 ± 0.031	0.901 ± 0.032	0.683 ± 0.026	0.876 ± 0.019	0.905 ± 0.010	0.858 ± 0.088
	DGC-MIL [15]	*0.926 ± 0.023*	*0.902 ± 0.025*	*0.686 ± 0.010*	*0.883 ± 0.018*	*0.912 ± 0.008*	*0.862 ± 0.089*
	BGNN-MIL [25]	0.915 ± 0.023	0.894 ± 0.014	0.677 ± 0.012	0.881 ± 0.018	0.903 ± 0.011	0.854 ± 0.089
Ours	RGMIL	**0.939 ± 0.020**	**0.926 ± 0.010**	**0.712 ± 0.019**	**0.898 ± 0.014**	**0.926 ± 0.010**	**0.880 ± 0.085**

fibroblast, and miscellaneous. Epithelial ROIs are often highly associated with colon cancer, as the latter often arise from the former. Hence, we regard epithelial ROIs as positive instances, while ROIs belonging to the remaining categories are negative instances. Therefore, an image bag is positive if it contains one or more ROIs from the epithelial class.

4) Baselines: We compare RGMIL with popular baselines, including mi-SVM & MI-SVM [45], MI-Kernel [49], EM-DD [50], mi-Graph [12], miVLAD & miFV [51]. Since the above are all shallow methods, we also utilize DNN-based baselines, containing mi-NET & MI-NET & MI-NET with DS or RC [9], AbMIL & AbMIL-Gated [1], MIVAE [52], ARP-MINN [53]. Moreover, we treat GNN-based baselines as a separate branch, comprising GNN-MIL [13], GNN-RLHI [14], DGC-MIL [15], BGNN-MIL [25], NAGCN [26]. Among these methods, mi- & MI-SVM are support vector machine (SVM)-based algorithms, which make SVM follow the MIL assumption that the positive bag contains at least one positive instance. MI-Kernel is based on kernel techniques. EM-DD combines the EM with the DD. mi-Graph achieves MIL by performing kernel learning on bag graphs. For DNN-based MIL baselines, NET and AbMIL use mean- and attention-based pooling to obtain bag-level features. MIVAE is the model with the variational auto-encoder (VAE). ARP-MINN is a model based on dynamic pooling. For GNN-based MIL algorithms, GNN-MIL [13] is the first GNN-based MIL model. GNN-RLHI [14] introduces global information to instance features and then creates bag graphs. DGC-MIL [15] is the method based on a deep GCN [54] and feature selection. BGNN-MIL [25] is a Bayesian GNN framework that proposes a way to construct likely inter-bag relationships. NAGCN [26] captures the hierarchical patterns of bag graphs by constructing sub-graphs. Because RGMIL implements MADRL through the optimized VDN, here we introduce the independent Q-learning (IQL) [55]-based control. IQL is another widely used MADRL algorithm, where each agent regards other agents as part of the environment and has no interaction with each other.

B. Implementation Settings

For fairness, we utilize the evaluation techniques commonly used in previous studies, applying ten-fold cross-validation and repeating each task five times. Then, we utilize the mean with a standard deviation of accuracy and F1-score, etc. as metrics to verify the performance of models in these MIL tasks. Because ten-fold cross-validation divides each dataset into equal-sized blocks, nine of them naturally form a state space $\mathcal{S}$ as well as a validation set of MG. Due to the small number of samples like UCSB-BREAST, training resources may be insufficiency after cutting out a verification set. Hence, for any repetition, we only search for actions using the first cross-validation and guide the subsequent validations based on the action combination.

Since the edge filtering threshold is a decimal, it falls within the interval $[0, 1]$, where 0 indicates retaining all edges, while 1 signifies the removal of all edges. Therefore, we partition $[0, 1]$ into smaller intervals using a finer granularity of 0.05, creating the action space $\{0.05i|i \in [1, 20], i \in \mathbb{N}^+\}$ for the first agent. Since increasing the number of GNN layers will exponentially expand the node aggregation scope, usually shallow GNNs are sufficient to unlock the full model performance, the upper limit of layers is set to 10. The action space of the second agent is $\{j|j \in [1, 10], i \in \mathbb{N}^+\}$. Besides, we intentionally set the final time step to be challenging to reach ($T = 10000$). As a result, we reserve sufficient time space to evaluate the effectiveness of the proposed termination condition (Eq. 16), which is expected to stop MG early. At each time step, early termination requires that the fluctuations of past μ reward values to be stable (mean value of μ rewards tends to 0). Due to the difficulty of precisely achieving a mean value of 0, we utilize a value close to 0, i.e., $\lambda = 0.0001$, to ensure convergence of the MG. Since μ is also the maximum number of recorded action combinations, we set

[1]The "Mean" columns of all the tables in this work present the performance "mean ± standard deviation" of each method under the datasets or metrics in the previous columns.

TABLE IV: Classification results (average accuracy ± standard deviation) on textual news datasets.

Method		Dataset [12]					Mean
		comp.graphics	comp.windows.x	rec.sport.baseball	sci.med	talk.politics.mideast	
Shallow	mi-Graph [12]	0.778 ± 0.016	0.698 ± 0.021	0.647 ± 0.031	0.621 ± 0.039	0.736 ± 0.026	0.696 ± 0.057
	miFV [51]	0.594 ± 0.063	0.768 ± 0.069	0.779 ± 0.066	0.783 ± 0.056	0.793 ± 0.060	0.743 ± 0.075
DNN-based	MIVAE [52]	0.800 ± 0.042	0.754 ± 0.032	0.764 ± 0.036	0.745 ± 0.025	0.840 ± 0.020	0.781 ± 0.035
	ARP-MINN	0.820 ± 0.055	*0.838 ± 0.041*	*0.818 ± 0.062*	0.782 ± 0.058	*0.858 ± 0.051*	*0.823 ± 0.025*
GNN-based	GNN-RLHI [14]	0.826 ± 0.044	0.828 ± 0.035	0.810 ± 0.046	0.780 ± 0.061	0.852 ± 0.054	0.819 ± 0.024
	DGC-MIL [15]	*0.828 ± 0.048*	0.830 ± 0.023	0.814 ± 0.067	*0.782 ± 0.046*	0.856 ± 0.036	0.822 ± 0.024
Ours	RGMIL	**0.840 ± 0.030**	**0.861 ± 0.047**	**0.846 ± 0.041**	**0.804 ± 0.022**	**0.858 ± 0.034**	**0.842 ± 0.020**

TABLE V: Classification results (average accuracy ± standard deviation) on COLON-CANCER.

Method		Metric					Mean
		Accuracy	Precision	Recall	F1-score	AUC	
DNN-based	MIVAE [52]	0.926 ± 0.020	0.933 ± 0.019	0.932 ± 0.012	0.928 ± 0.008	0.929 ± 0.015	0.930 ± 0.003
	ARP-MINN [53]	0.928 ± 0.066	0.934 ± 0.006	0.939 ± 0.108	0.928 ± 0.072	0.933 ± 0.064	0.932 ± 0.004
GNN-based	GNN-RLHI [14]	0.934 ± 0.005	0.929 ± 0.015	0.938 ± 0.017	0.926 ± 0.004	0.968 ± 0.002	0.939 ± 0.015
	DGC-MIL [15]	0.940 ± 0.063	0.933 ± 0.019	0.939 ± 0.014	*0.931 ± 0.006*	0.962 ± 0.009	0.941 ± 0.011
	NAGCN [26]	*0.940 ± 0.014*	***0.935 ± 0.105***	*0.943 ± 0.026*	0.931 ± 0.016	*0.973 ± 0.004*	*0.944 ± 0.015*
Ours	RGMIL	**0.942 ± 0.007**	0.930 ± 0.015	**0.958 ± 0.027**	**0.938 ± 0.008**	**0.974 ± 0.003**	**0.948 ± 0.016**

TABLE VI: Results on MESSIDOR and UCSB-BREAST.

Dataset	Method		
	GNN-RLHI [14]	DGC-MIL [15]	RGMIL
MESSIDOR [46]	0.725 ± 0.025	*0.737 ± 0.035*	**0.746 ± 0.025**
BREAST [2]	0.902 ± 0.026	*0.910 ± 0.018*	**0.917 ± 0.019**

it to a smaller value ($\mu = 10$) to speed up the training of GNN and the termination of the game. Generally, the discount factor γ in Q-learning is typically set within the interval of $[0, 1]$ [42]. When the γ is close to 0, the agent places more emphasis on immediate rewards. In order to prioritize long-term cumulative rewards, as done in previous studies [20], [33], [34], we set γ to be close to 1, i.e., $\gamma = 0.95$. Furthermore, RGMIL involves two methods that are widely used to enhance the reliability of MADRL. For ϵ-greedy strategy (Eq. 5), we follow the classical work [43] and equidistantly reduce the initial probability $\epsilon = 1$ to 0 in the short term, i.e., the first 50 time steps. In this way, RGMIL prevents the agents from falling into the local optimal solution while accelerating game convergence. For experience replay (Sec. IV-E), since replaying newer experiences involves the risk of over-estimation [20], and earlier yielded experiences become unreliable over time, we set the memory stack capacity to be relatively modest 20, similar to previous studies like [34], [43], [55]. Each agent $\pi(\cdot)$ utilizes the same DNN architecture as [33]. Then, we train the GNN and agents based on the Adam optimizer with a learning rate of 0.0005 and a decay factor of 0.001, which are common settings of deep learning framework. The activation functions $\sigma(\cdot)$ all use $LeakyRelu$ with a slope rate of 0.1, which is often followed by the dropout with a drop rate of 0.2, For all baselines, we follow the settings mentioned in their studies to tune hyperparameters and report the highest performance. Since COLON-CANCER does not have instance feature vectors like other datasets, we use 27x27 ROI instances as inputs. Considering that methods like MIVAE and RGMIL are unable to handle images, we utilize a two-layer LeNet [56] with a kernel size of 5 as the front network for instance feature generation. All experiments are performed on the same server with four NVIDIA GeForce RTX 3080 (10240 MiB).

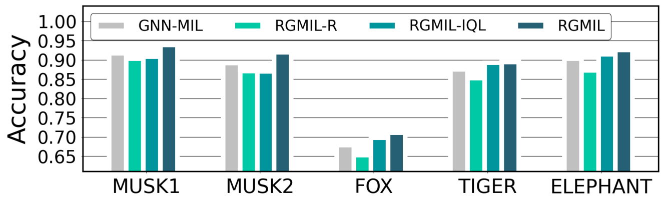
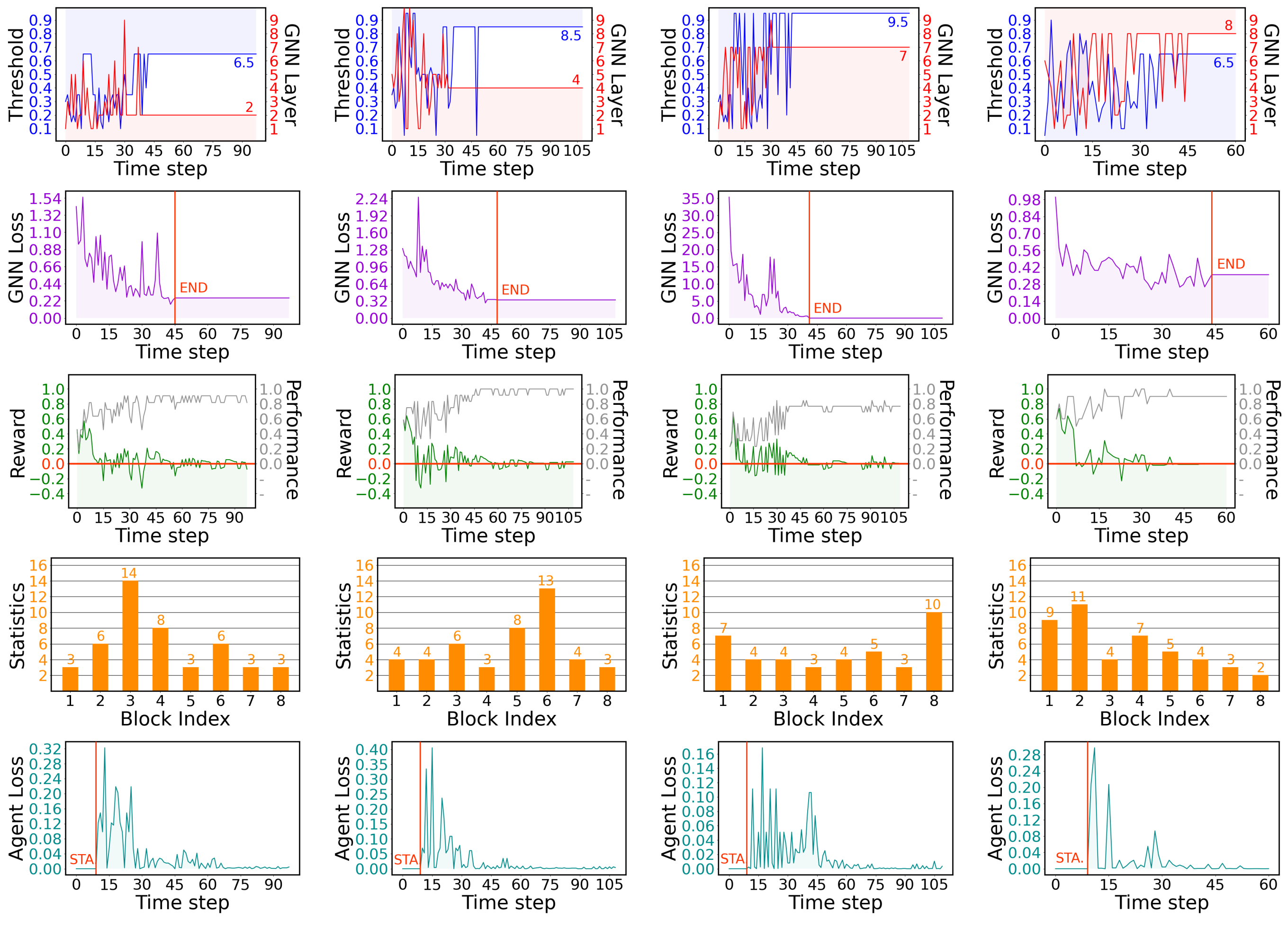


Fig. 5: Visualization of ablation studies on benchmark datasets.

C. *Results and Analysis*

The bag classification results on the five benchmark datasets are shown in Table III, from which we can gather the following six conclusions. First of all, RGMIL outperforms baselines in accuracy across all datasets (MUSK1, MUSK2, FOX, TIGER, ELEPHANT), which signifies that there is valuable correlation information among instances in the bag graphs of five datasets. At the same time, RGMIL locates the best combination of edge filtering threshold and GNN layer number in MG, promoting a harmonious correlation between edge density and aggregation. Second, compared to other non-DNN-based shallow baselines, mi-Graph, which considers inter-instance correlations achieves better results. This observation indicates that instances in each bag are often not independent but have an underlying topology. Third, MIVAE and ARP-MINN stand out among DNN-based baselines. This is because non-IID MIVAE explicitly organizes dependencies between instances, while ARP-MINN introduces adaptive recurrent pooling, both of which further improve bag-level feature representations. Fourth, GNN-based baselines are comprehensively superior to shallow or DNN-based baselines, including correlation-aware mi-graph and MIVAE. A possible explanation is that GNNs have better ability to learn and utilize topological representations, which facilitates the breakthrough of MIL performance boundaries. Fifth, DGC-MIL outperforms the baselines in the GNN category. This is because DGC-MIL proposes an effective instance feature selection strategy, which improves the discriminability of different bag samples. On the other hand, DGC-MIL (same as GNN-RLHI) has tuned layers because their articles lack an explicit setting. Last but not least,



(a) MUSK1 (b) MUSK2 (c) UCSB-BREAST (d) COLON-CANCER

Fig. 6: Visualization of MADRL in RGMIL on four datasets. The first row shows the action decision process of the two agents. The second row depicts the optimization process of the GNN model. The third row shows the transformation of the reward and verification performance. The fourth row depicts the number of traversals of each training block (state). The last row shows the joint optimization of the two agents. "END" and "STA." (red words) mean the time step to end and start training, respectively.

we believe that one-layer GNN-MIL and BGNN-MIL may not adequately capture inter-instance correlations, even though the latter considers inter-bag correlations.

Table IV presents the performance comparison of bag-level classification on five news text datasets, from which we obtain the following three conclusions. Firstly, RGMIL also achieves the highest average accuracy on five textual datasets. Secondly, MIVAE and GNN-based baselines do not achieve better results than IID-based ARP-MINN. One explanation is that instances originate from too many topics, resulting in a large number of meaningless edges encoded in a bag graph. At the same time, since RGMIL achieves customized edge filtering for different news datasets, it always learns the best instance-level features on the optimal bag topology. Hence, high-quality edge filtering is an effective method to improve GNN-based MIL algorithms. Finally, unlike the results on benchmark datasets, DNN-based baseline shows significant performance improvement over the shallow baseline. This is because instances of news data equip sparse features that are not processed or precomputed, leaving enough room for DNNs to tune and optimize.

Moreover, classification experiments are performed on three medical image datasets to test the performance of the RGMIL. Specifically, we compare different methods across five metrics on COLON-CANCER. From Table V we observe that RGMIL achieves the best mean results, and the other three GNN-based baselines also have excellent results. This shows that RGMIL is still effective on image MIL tasks, whose instances are also non-IID. It is worth noting that compared with these baselines, RGMIL improves the recall while maintaining a high precision value, which meets the requirements of disease detection tasks. Unlike other GNN-based baselines for whole graphs, NAGCN divides bag graphs into multiple sub-graphs and then performs feature aggregation and connection. The preassigned codebook required by NAGCN consists of four colonic tissue categories. Even with a suspicion of label leakage, NAGCN is still inferior to the RGMIL. The experimental results from MESSIDOR and UCSB-BREAST are shown in Table VI, from which we obtain a similar conclusion. Although RGMIL does not customize the image processing strategy like these baselines, it performs best on both tasks. More analysis will be given in the next section.

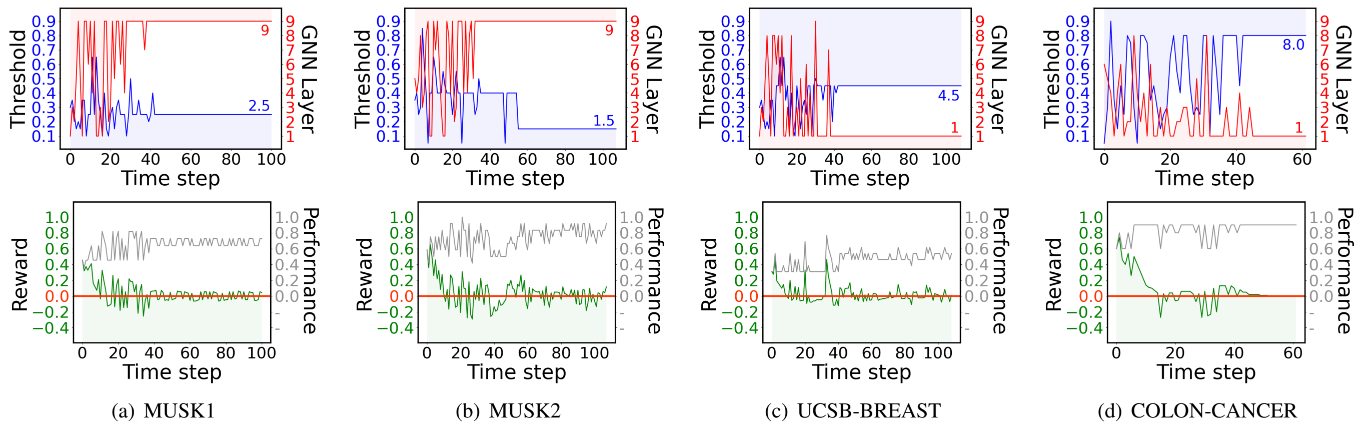


Fig. 7: Visualization of MADRL in RGMIL-IQL. The first and second show action selection and reward calculation, respectively.

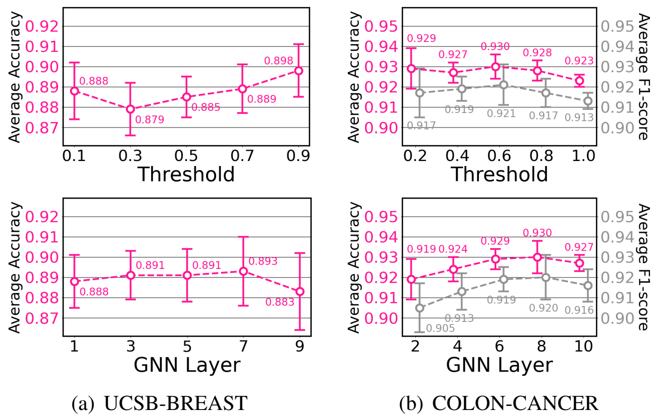


Fig. 8: Visualization of performance of RGMIL on two image datasets. The x-axes represent fixed edge filtering thresholds or GNN layers, while the y-axes indicate the average performance over all possible choices in another action space.

D. Ablation Studies

We conduct ablation studies to analyze the specific effects of each module in Sec. IV-A-IV-D. Because RGMIL innovatively uses MADRL (i.e., VDN algorithm) to simultaneously control the bag structures and GNN architectures in MIL, we construct an IQL-based RGMIL as a control, RGMIL-IQL, whose game tuple $\mathcal{M}$ is consistent with that of RGMIL, but each agent runs independently. Besides, we build RGMIL-R without MADRL (VDN and IQL), which randomly chooses action combinations before each repetition. Fig.5 shows the prediction performance of RGMIL, RGMIL-IQL, RGMIL-R, and GNN-MIL across all benchmark datasets. For the two MUSK datasets, RGMIL-IQL and RGMIL-R show comparable accuracy, but fall short when compared to GNN-MIL and RGMIL. On the one hand, these observations justify the effectiveness of observation interaction and joint reward calculation in the RGMIL. On the other hand, RGMIL-IQL may be tough to find the best action combination, resulting in even worse performance than baselines with fixed or random settings. For the other datasets, RGMIL-IQL is only inferior to RGMIL, which means MADRL has the potential to improve GNN-based MIL. RGMIL-R is still the worst method, reflecting the high complexity of manual tuning.

To more intuitively display the performance of each module in RGMIL, we visualize the MG process on MUSK1, MUSK2, UCSB-BREAST, and COLON-CANCER, as shown in Fig. 6. Specifically, the first two rows show how the RGMIL performs action selection and guides GNN training. The third row shows the reward calculation process using the historical verification performance. Since the termination condition depends on these reward records, the third row also shows the final convergence of the game. The fourth row counts the access statistics of all states (training blocks) in the state space. The last row presents the optimization process of the agents, which differentiates and optimizes the two agents by value decomposition.

The figures in Fig. 6 can be divided into three stages. For the beginning stage (time steps before “STA.”), the agents explore different action combinations with high probability. Therefore, the action selection at this stage fluctuates greatly. At the same time, the verification performance of the model with large loss changes is not stable, resulting in large fluctuations in rewards. After gathering sufficient experiences through continuous state transfer, RGMIL optimizes the agents in the second stage (time steps between “STA.” and “END”). At this stage, the gradually stable agents drive the game to gradually converge. Then GNN completes optimization under relatively stable action guidance. Rewards at the final stage (time steps after “END”) still change due to slight fluctuations in performance until MG terminates, where the mean of the last μ rewards approaches 0. This game converges to the Nash equilibrium, where the last actions form the optimal combination.

In addition to the stage division, we also draw the following observations. First, since RGMIL selects actions that maximize Q-values with an increasing probability $(1-\epsilon)$, the fluctuations in action selection (sub-figures in the first row) are increasingly affected by fluctuations in agent optimization (in the last row). Second, because Sec. IV-B dictates the maximum μ of training times of GNN using the same action combination, the RGMIL often stops the GNN training (in the second row) quickly after the actions stabilize. In other words, no new combinations will appear in the time steps after “END”. Third, the improvement of GNN training efficiency makes the verification performance gradually stabilize. Since the reward function is done based on historical performance in Sec. IV-C, rewards and performance yield similar fluctuations (in the third row), improving reward reliability and game stability. Finally, the global state transition

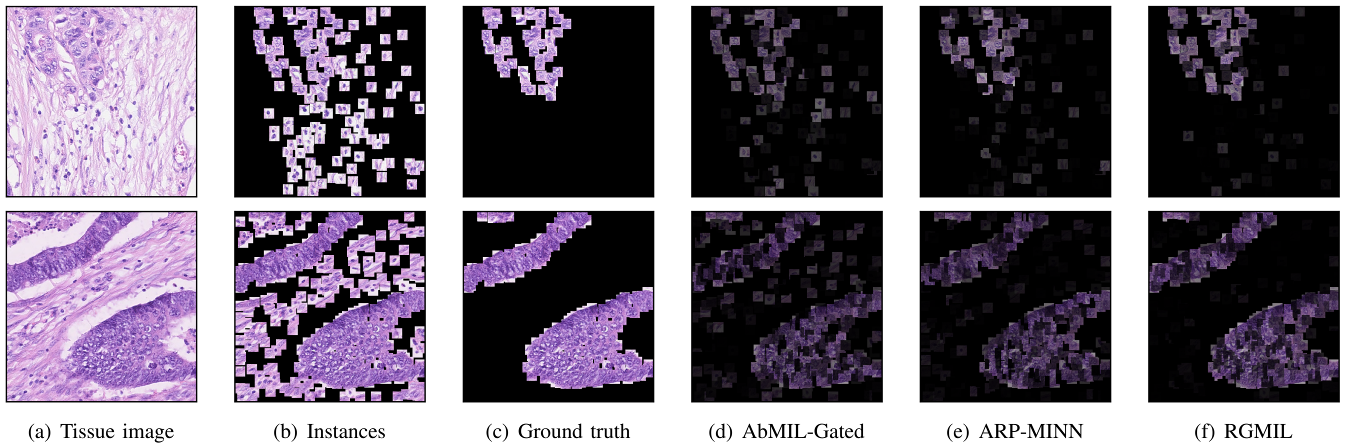


Fig. 9: Visualization of instance weights of two randomly selected image bags from COLON-CANCER. (a) The initial colon tissue images. (b) All ROI instances of shape 27×27. (c) Ground truths: ROIs that belong to the class epithelial. (d)-(f) Heatmaps from the three models, i.e., AbMIL-Gated, ARP-MINN, and RGMIL.

in Sec. IV-D traverses all training blocks differently depending on the frequency of different actions (in the fourth raw).

For the observation interaction and termination functions in sec. IV-A and sec. IV-D, we further depict the MADRL control of RGMIL-IQL in Fig. 7. Since the two agents in RGMIL-IQL do not interact in observations and contributions, they tend to select extreme actions. Extreme action combinations are often prone to embedding issues, which is why RGMIL-IQL is even worse than RGMIL-R on MUSK1/2 datasets in Fig. 5. Besides, we find that RGMIL-IQL, which uses the termination function in Sec. IV-D, also ensures game convergence. Since we tie the termination condition to the historical rewards that are tied to the performance, the bottlenecks in classification accuracy will force the game to complete the convergence.

E. Correlation Studies

After illustrating Fig. 6 and Fig. 7, we qualitatively ascertain that RGMIL using the optimized VDN is better than RGMIL-IQL. Therefore, we quantitatively inspect the quality of action combinations yielded by RGMIL, and analyze the correlations between edge densities in the bags and aggregation iterations. Fig. 8 shows the mean performance of RGMIL over all actions in another action space after fixing the edge filtering threshold (first row) or the number of layers (second row). For BREAST (first column), we find that when the threshold is fixed at about 0.9 or the number of GNN layers is fixed at about 7, RGMIL does best, which is consistent with the final results of MADRL in Fig. 6. Similar conclusions can also be obtained on COLON, which shows that the combination results of RGMIL are highly accurate. These results show that when the edge density is low, more iterations are needed to aggregate farther hop features for better GNN performance. In contrast, higher densities typically require fewer aggregation iterations, aligning with the analysis in Sec. I. Since changes in thresholds or layers produce varying effects on performance across different datasets, the above two correlations are task-specific and manual adjustment is tedious. Overall, RGMIL achieves synchronized and reliable control of bag graphs and GNN models using MADRL, without ignoring structure-to-architecture correlations.

TABLE VII: Time consumption (in minutes) of a repetition of six GNN-based MIL algorithms on the animal detection tasks.

Dataset [45]	Method					
	GNN -MIL [13]	GNN -RLHI [14]	DGC -MIL [15]	BGNN -MIL [25]	NAGCN [26]	BGMIL
FOX	**3.56**	*4.41*	5.23	8.05	5.76	6.54
TIGER	**3.15**	*4.28*	5.76	7.96	5.53	6.63
ELEPHANT	**4.21**	*4.58*	5.86	8.15	5.40	6.27

F. Case Studies

It is important to understand the extent to which instances in a bag affect the classification results. In other words, we need to explain why the bag classification results were made. Thus, we perform case studies on COLON-CANCER in this section. As shown in Fig. 9, we visually compare the instance weights assigned by diverse models in two correctly classified positive image bags. The heatmaps of each image bag are obtained by multiplying the region pixels with their corresponding instance weights, which are normalized to $[0, 1]$. Then, we observe that the heatmaps of RGMIL (in the sixth column) are aligned with the ground truths (in the third column). Specifically, compared to the similarly attention-based global pooling of RGMIL, the AbMIL-Gated (fourth column) struggles to effectively conceal irrelevant instances. One possible reason is that AbMIL-Gated disregards modeling inter-instance dependencies and sufficient instance-level feature discrimination. Though ARP-MINN (in the fifth column) proposes a more advanced adaptive recurrent pooling, RGMIL allocates higher weights (brighter heatmaps) to the positive epithelial instances than ARP-MINN. This may be attributed to the better instance-level feature representation learning ability of RGMIL. The above three conclusions justify that RGMIL achieves high-quality bag classification while also having excellent result explainability. Besides, it is particularly worth noting that RGMIL only has result explainability but not model explainability. For instance, the attention-based pooling in RGMIL enables focusing on essential instances that lead to bag prediction results, but the explainability of its mechanism remains a subject of ongoing debate [57], [58].

G. Efficiency Studies

We compare the time consumption of GNN-based strategies, which are more complex but provide superior performance. As an crucial factor in engineering applications, time consumption often impacts system efficiency, user experience, and resource utilization. Table. VII gives the time-consuming comparison of the six algorithms in the three animal detection scenarios, from which we obtain the following conclusions. Compared with the basic GNN-MIL, RGMIL brings additional time consumption due to the introduction of MADRL. In light of the outstanding performance exhibited by RGMIL, it is worthwhile to explore the innovative application of RL to augment GNN-based MIL.

VI. Conclusion

In this paper, we propose a new GNN-based MIL framework RGMIL, enabling automated and synchronized control of bag structures (the edge filtering threshold) and GNN architectures (the number of GNN layers). RGMIL presents a novel avenue for subsequent graph data mining studies, allowing the use of MADRL to search for nodes beyond the scope of graph neural architectures, an aspect unattainable through traditional GNAS methods. The experimental findings indicate that balancing the edge density and the aggregation scope enhances the untapped potential of GNNs. In the future work, we will explore refining instance-level reinforcement control in MIL, striving to utilize the MADRL while simultaneously reducing time consumption.

Acknowledgments

This work is supported by the National Key R&D Program of China through grant 2021YFB1714800, NSFC through grants 62322202, 61932002, U21B2027, U23A20388, and 62266028, Beijing Natural Science Foundation through grant 4222030, Guangdong Basic and Applied Basic Research Foundation through grant 2023B1515120020, and Shijiazhuang Science & Technology Plan through grant 231130459A.

References

[1] M. Ilse, J. Tomczak, and M. Welling, "Attention-based deep multiple instance learning," in *ICML*, 2018, pp. 2127-2136.

[2] M. Kandemir, C. Zhang, and F. A. Hamprecht, "Empowering multiple instance histopathology cancer diagnosis by cell graphs," in *MICCAI*, 2014, vol. 8674, pp. 228-235.

[3] Z. Zhou, K. Jiang, and M. Li, "Multi-instance learning based web mining," in *Applied Intelligence*, 2005, vol. 22, pp. 135-147.

[4] D. Zhang, Y. Liu, L. Si, J. Zhang, and R. Lawrence, "Multiple instance learning on structured data," in *NeurIPS*, 2011, vol. 24, pp. 145–153.

[5] J. Wu, X. Zhu, C. Zhang, and S. Y. Philip, "Bag constrained structure pattern mining for multi-graph classification," *IEEE TKDE*, 2014, vol. 26, no. 10, pp. 2382-2396.

[6] Y. Ji, H. Liu, B. He, X. Xiao, H. Wu, and Y. Yu, "Diversified multiple instance learning for document-level multi-aspect sentiment classification," in *EMNLP*, 2020, pp. 7012-7023.

[7] J. Wu, Y. Yu, C. Huang, and K. Yu, "Deep multiple instance learning for image classification and auto-annotation," in *CVPR*, 2015, pp. 3460-3469.

[8] G. Yu, G. Zhou, X. Zhang, C. Domeniconi, and M. Guo, "DMIL-IsoFun: predicting isoform function using deep multi-instance learning," in *Bioinformatics*, 2021, vol. 37, no. 24, pp. 4818–4825.

[9] X. Wang, Y. Yan, P. Tang, X. Bai, and W. Liu, "Revisiting multiple instance neural networks," in *Pattern Recognition*, 2018, vol. 74, pp. 15-24.

[10] X. Shi, F. Xing, Y. Xie, Z. Zhang, L. Cui, and L. Yang, "Loss-based attention for deep multiple instance learning," in *AAAI*, 2020, vol. 34, No. 4, pp. 5742-5749.

[11] Z. Zhou and J. Xu, "On the relation between multi-instance learning and semi-supervised learning," in *ICML*, 2007, pp. 1167-1174.

[12] Z. Zhou, Y. Sun, and Y. Li, "Multi-instance learning by treating instances as non-iid samples," in *ICML*, 2009, pp. 1249-1256.

[13] M. Tu, J. Huang, X. He, and B. Zhou, "Multiple instance learning with graph neural networks," in *ICML Workshop on Learning and Reasoning with Graph-Structured Data*, 2019.

[14] M. Adnan, S. Kalra, and H. R. Tizhoosh, "Representation learning of histopathology images using graph neural networks," in *CVPR Workshops*, 2020, pp. 988-989.

[15] Y. Zhao, F. Yang, Y. Fang, H. Liu, N. Zhou, J. Zhang, ..., and J. Yao, "Predicting lymph node metastasis using histopathological images based on multiple instance learning with deep graph convolution," in *CVPR*, 2020, pp. 4837-4846.

[16] J. Xiong, Z. Li, G. Wang, Z. Fu, F. Zhong, T. Xu, ..., and M. Zheng, "Multi-instance learning of graph neural networks for aqueous pKa prediction," in *Bioinformatics*, 2022, vol. 38, no. 3, pp. 792–798.

[17] K. Oono and T. Suzuki, "Graph neural networks exponentially lose expressive power for node classification," in *ICLR*, 2019.

[18] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun, "Measuring and relieving the over-smoothing problem for graph neural networks from the topological view," in *AAAI*, 2019, vol. 34, no. 4, pp. 3438-3445.

[19] L. S. Shapley, "Stochastic games," in *NAS*, 2020, vol. 39, no. 10, pp. 1095-1100.

[20] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, ..., T. Graepel, "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *AAMAS*, 2018, vol. 3, pp. 2085-2087.

[21] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *ICLR*, 2018.

[22] Y. Xing, G. Yu, J. Wang, C. Domeniconi, and X. Zhang, "Weakly-supervised multi-view multi-instance multi-label learning," in *IJCAI*, 2021, pp. 3124–3130.

[23] Y. Xing, G. Yu, C. Domeniconi, J. Wang, Z. Zhang, and M. Guo, "Multi-view multi-instance multi-label learning based on collaborative matrix factorization," in *AAAI*, 2019, vol. 33, no. 01, pp. 5508–5515.

[24] G. Yu, Y. Xing, J. Wang, C. Domeniconi, and X. Zhang, "Multiview multi-instance multilabel active learning," in *IEEE TNNLS*, 2021, vol. 33, no. 9, pp. 4311–4321.

[25] S. Pal, A. Valkanas, F. Regol, and M. Coates, "Bag graph: Multiple instance learning using bayesian graph neural networks," in *AAAI*, 2022, vol. 36, no. 7, pp. 7922-7930.

[26] Y. Guan, J. Zhang, K. Tian, S. Yang, P. Dong, J. Xiang, ..., and X. Han, "Node-aligned graph convolutional network for whole-slide image representation and classification," in *CVPR*, 2022, pp. 18813-18823.

[27] K. Xu, C. Li, Y. Tian, T. Sonobe, K. I. Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," in *ICML*, 2018, pp. 5453-5462.

[28] G. Li, M. Muller, A. Thabet, and B. Ghanem, "Deepgcns: Can gcns go as deep as cnns?" in *ICCV*, 2019, pp. 9267-9276.

[29] H. Chen, Y. Xu, F. Huang, Z. Deng, W. Huang, S. Wang, ..., and Z. Li, "Label-aware graph convolutional networks," in *CIKM*, 2019, pp. 1977-1980.

[30] H. Peng, R. Zhang, Y. Dou, R. Yang, J. Zhang, and P. S. Yu, "Reinforced neighborhood selection guided multi-relational graph neural networks," *ACM TOIS*, 2021, vol. 40, no. 4, pp. 1-46.

[31] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters," in *CIKM*, 2021, pp. 315-324.

[32] X. Zhao, Q. Dai, J. Wu, H. Peng, M. Liu, X. Bai, ..., and P. S. Yu, "Multi-view tensor graph neural networks through reinforced aggregation," *IEEE TKDE*, 2022, vol. 35, no. 4, pp. 4077–4091.

[33] X. Zhao, J. Wu, H. Peng, A. Beheshti, J. J. Monaghan, D. McAlpine, ..., and L. He, "Deep reinforcement learning guided graph neural networks for brain network analysis," in *Neural Networks*, 2022, vol. 154, pp. 56-67.

[34] K. H. Lai, D. Zha, K. Zhou, and X. Hu, "Policy-gnn: Aggregation optimization for graph neural networks," in *KDD*, 2020, pp. 461-471.

[35] Y. Yang, Y. Pan, C. Xu, and D. C. Wunsch, "Hamiltonian-driven adaptive dynamic programming with efficient experience replay," in *IEEE TNNLS*, 2022, pp. 1–13.

[36] Y. Yang, B. Kiumarsi, H. Modares, and C. Xu, "Model-free λ-policy iteration for discrete-time linear quadratic regulation," in *IEEE TNNLS*, 2023, vol. 34, no. 2, pp. 635–649.

[37] Y. Yang, H. Modares, K. G. Vamvoudakis, and F. L. Lewis, "Cooperative finitely excited learning for dynamical games," in *IEEE Transactions on Cybernetics*, 2023, pp. 1–14.

[38] Y. Gao, H. Yang, P. Zhang, C. Zhou, and Y. Hu, "Graph neural architecture search," in *IJCAI*, 2020, pp. 1403–1409.

[39] M. Yoon, T. Gervet, B. Hooi, and C. Faloutsos, "Autonomous graph mining algorithm search with best speed/accuracy trade-off," in *IEEE ICDM*, 2020, pp. 751-760.

[40] J. You, Z. Ying, and J. Leskovec, "Design space for graph neural networks," in *NeurIPS*, 2020, vol. 33, pp. 17009-17021.

[41] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of Markov decision processes," in *Mathematics of Operations Research*, 2002, vol. 27, no. 4, pp. 819-840.

[42] C. J. Watkins, and P. Dayan, "Q-learning," in *Machine Learning*, 2002, vol. 8, no. 3, pp. 279-292.

[43] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, ..., and D. Hassabis, "Human-level control through deep reinforcement learning," in *Nature*, 2015, vol. 518, no. 7540, pp. 529-533.

[44] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, "Solving the multiple instance problem with axis-parallel rectangles," in *Artificial Intelligence*, 1997, vol. 89, no. 1-2, pp. 31-71.

[45] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," in *NeurIPS*, 2002, vol. 14, pp. 561-568.

[46] E. Decencière, X. Zhang, G. Cazuguel, B. Lay, B. Cochener, C. Trone, ..., and J. C. Klein, "Feedback on a publicly distributed image database: the Messidor database," in *Image Analysis & Stereology*, 2014, vol. 33, no. 3, pp. 231-234.

[47] M. Kandemir and F. A. Hamprecht, "Computer-aided diagnosis from weak supervision: A benchmarking study," in *Computerized Medical Imaging and Graphics*, 2015, vol. 42, pp. 44-50.

[48] L. R.-Vitiani, D. G. Lombardi, E. Pilozzi, M. Biffoni, M. Todaro, C. Peschle, and R. De Maria, "Identification and expansion of human colon-cancer-initiating cells," in *Nature*, 2007, vol. 445, no. 7123, pp. 111-115.

[49] T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola, "Multi-instance kernels," in *ICML*, 2014, vol. 2, no. 3, pp. 179-186.

[50] Q. Zhang and S. Goldman, "EM-DD: An improved multiple-instance learning technique," in *NeurIPS*, 2001, vol. 14, pp. 1073-1080.

[51] X. Wei, J. Wu, and Z. Zhou, "Scalable algorithms for multi-instance learning," *IEEE TNNLS*, 2016, vol. 28, no. 4, pp. 975-987.

[52] W. Zhang, "Non-I.I.D. multi-instance learning for predicting instance and bag labels with variational auto-encoder," in *IJCAI*, 2021, pp. 3377–3383.

[53] Y. Ding, L. Zhao, L. Yuan, and X. Wen, "Deep multi-instance learning with adaptive recurrent pooling for medical image classification," *IEEE BIBM*, 2022, pp. 3335-3342.

[54] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017, pp. 1–14.

[55] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, ..., and R. Vicente, "Multiagent cooperation and competition with deep reinforcement learning," in *PLOS ONE*, 2017, vol. 12, no. 4, pp. e0172395.

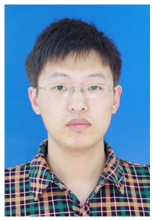
[56] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, 1998, vol. 86, no. 11, pp. 2278-2324.

[57] S. Wiegreffe and Y. Pinter, "Attention is not not explanation," in *EMNLP-IJCNLP*, 2019, pp. 11–20.

[58] S. Jain and B. C. Wallace, "Attention is not explanation," in *NAACL-HLT*, 2019, pp. 3543–3556.

Xusheng Zhao is currently a Ph.D. candidate in the Institute of Information Engineering, Chinese Academy of Sciences, and the School of Cyber Security, University of Chinese Academy of Sciences. His research interests include representation learning and reinforcement learning.

Qiong Dai is currently an Associate Professor at the Institute of Information Engineering, Chinese Academy of Sciences. Her current research interests include data mining, knowledge graph, and collaborative computing.



Xu Bai is an Engineering at the Institute of Information Engineering, Chinese Academy of Sciences. His current research interests include network security and social computing.

Jia Wu received the Ph.D. degree in computer science from the University of Technology Sydney, Australia. Dr. Wu is currently the Research Director for the AI-enabled Processes (AIP) Research Centre and an ARC DECRA Fellow in the School of Computing, Macquarie University, Sydney, Australia. He is the Associate Editor of the ACM Transactions on Knowledge Discovery from Data (TKDD) and Neural Networks (NN).

Hao Peng is currently a Professor at the School of Cyber Science and Technology, Beihang University. His research interests include representation learning, social network mining, and reinforcement learning. To date, Dr. Peng has published over 150 research papers in top-tier journals and conferences, including the IEEE TPAMI, TKDE, TPDS, TC, JMLR, ACM TOIS, TKDD, and Web Conference. He is the Associate Editor of the International Journal of Machine Learning and Cybernetics (IJMLC).

Huailiang Peng is an Engineer at the Institute of Information Engineering, Chinese Academy of Sciences. His current research interests include social network analysis and graph representation learning.

Zhengtao Yu received the Ph.D. degree in computer application technology from the Beijing Institute of Technology, Beijing, China, in 2005. He is currently a Professor at the School of Information Engineering and Automation, Kunming University of Science and Technology, China. His research interests include natural language processes, image processing, and machine learning.

Philip S. Yu is a Distinguished Professor and the Wexler Chair in Information Technology at the Department of Computer Science, University of Illinois at Chicago. Before joining UIC, he was at the IBM Watson Research Center, where he built a world-renowned data mining and database department. He is a Fellow of the ACM and IEEE. Dr. Yu was the Editor-in-Chiefs of ACM Transactions on Knowledge Discovery from Data (2011-2017) and IEEE TKDE (2001-2004).