Reinforcement Learning on Graphs: A Survey

Mingshuo Nie, *Graduate Student Member, IEEE*, Dongming Chen, *Member, IEEE*, and Dongqi Wang

***Abstract*—Graph mining tasks arise from many different application domains, including social networks, biological networks, transportation, and E-commerce, which have been receiving great attention from the theoretical and algorithmic design communities in recent years, and there has been some pioneering work employing the research-rich Reinforcement Learning (RL) techniques to address graph mining tasks. However, these fusion works are dispersed in different research domains, which makes them difficult to compare. In this survey, we provide a comprehensive overview of these fusion works and generalize these works to Graph Reinforcement Learning (GRL) as a unified formulation. We further discuss the applications of GRL methods across various domains, and simultaneously propose the key challenges and advantages of integrating graph mining and RL methods. Furthermore, we propose important directions and challenges to be solved in the future. To our knowledge, this is the latest work on a comprehensive survey of GRL, this work provides a global view and a learning resource for scholars. Based on our review, we create a collection of papers for both interested scholars who want to enter this rapidly developing domain and experts who would like to compare GRL methods.**

***Index Terms*—Graph reinforcement learning, Graph mining, Reinforcement learning, Graph neural networks.**

I. Introduction

THE recent success of Reinforcement Learning (RL) has solved challenges in different domains such as robotics [1], games [2], Natural Language Processing (NLP) [3], etc. RL addresses the problem of how agents should learn to take actions to maximize cumulative reward through interactions with the environment [4]. The rapid development of RL in cross-disciplinary domains has motivated scholars to explore novel RL models to address real-world applications, e.g., financial and economic [5], [6], [7], biomedical [8], [9], and transportation [10], [11]. While the effectiveness of RL had been widely successful in substantial applications, many real-world scenarios can be modeled as graph-structured data. Specifically, these scenarios can be modeled as environments that are represented by implicit or explicit relationships in graph structures, and graph-related downstream tasks have received extensive research, such as link prediction [12], [13], node classification [14], and graph classification [15]. Various strategies have been proposed to tackle graph-structured data, including novel information aggregation functions [16], graph structure pooling [17], and neighborhood sampling methods [12].

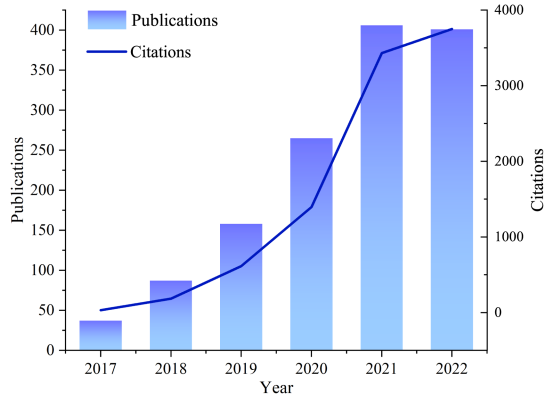

Fig. 1. Times cited and publications over time of GRL (January 2017-November 2022).

Recently, researchers have been exploring the advantages of fusing graph mining methods with powerful RL methods [18], [19], [20] to efficiently tackle applications based on graph-structured environments. We show the times cited and publications over time of Graph Reinforcement Learning (GRL) ranging from January 2017 to November 2022 in Fig. 1 to prove the fusion of graph mining methods with RL for graph-structured environments has attracted a lot of attention.

The traditional methods and deep learning-based models for graph mining tasks have major differences in terms of model design and training process from RL-based methods, and scholars are facing many challenges in employing RL methods to analyze graph-structured data: (1) Machine learning models used in computer vision and NLP effectively capture hidden patterns of data in the Euclidean space (e.g., images and speech), but these models will not be available for graph-structured data due to the data complexity. (2) RL methods are difficult to accurately establish the environment or state space, and design action space or reward functions for specific graph mining tasks [21]. Specifically, graphs are massive, high-dimensional, discrete objects, and are challenging to work within the RL modeling context. (3) The specific objectives of graph mining tasks require RL methods to design effective architectures that are adapted to graph-structured data. (4) The emergence of large-scale graph-structured data imposes limitations on the application of RL methods. The historical data-driven RL methods require constant learning from the environment to update parameters, and

Manuscript received 27 April 2022; revised 12 August 2022; accepted 4 November 2022. Date of publication 8 May 2023; date of current version 24 July 2023. This work was supported in part by the Applied Basic Research Project of Liaoning Province under Grant 2023JH2/101300185, in part by the Key Technologies Research and Development Program of Liaoning Province in China under Grant 2021JH1/10400079, and in part by the Fundamental Research Funds for the Central Universities under Grant N2217002. *(Corresponding author: Dongming Chen.)*

The authors are with the Software College, Northeastern University, Shenyang, Liaoning 110169, China (e-mail: niemingshuo@stumail.neu.edu.cn; chendm@mail.neu.edu.cn; wangdq@swc.neu.edu.cn).

Digital Object Identifier 10.1109/TETCI.2022.3222545

the growing data scale causes inefficiencies in RL methods. (5) Many interdisciplinary data in the real world can be modeled as graph-structured environments, and interdisciplinary data analysis integrating domain knowledge will prompt models toward expansion and complexity, which leads to significant difficulties in designing RL methods.

To address the above challenges, researchers have fused graph mining methods with RL methods and have been successful in many applications. We define GRL as solutions and measures for solving graph mining tasks by analyzing critical components such as nodes, links, and subgraphs in graphs with RL methods to achieve exploration of topological structure and attribute information of the graphs. We believe the advantages of the fusion between graph mining and RL methods can be summarized as follows:

1) *Auxiliary information:* Graph mining methods can capture topological and attribute information, and this information can be modeled as environments representing explicit or implicit structural relationships in graphs. *Graph mining methods can provide auxiliary information for RL, which improves RL methods to be applicable to graph-structured data.*
2) *Generalization:* RL methods enable learning specific objectives in networks with different topologies without considering the effect of network structure on prediction performance. *Since RL methods allow for adaptive learning by representing graph mining tasks as the MDP with sequential decision characteristics.*
3) *Adaptability:* Due to the natural structure of graphs, graph mining tasks can be solved with RL methods. Therefore, we believe that using RL to tackle graph mining is valuable. *The process of exploring graph structure can be modeled easily as a sequential decision problem, which is not available in other domains.*
4) *Data-driven:* Existing graph mining methods introduce extensive expert knowledge or require several rules to be developed manually. *However, RL methods are driven by input data and allow fast learning without expert knowledge, which allows these methods to achieve learning objectives more efficiently.*

There are a limited number of comprehensive reviews available to summarize the works using RL methods to solve graph mining. A review of these fused works would be beneficial in determining challenges and future research directions. Therefore, it is necessary to provide this systematic review of methods and frameworks for GRL. We aim to provide an intuitive understanding and high-level insight into the different methods so that scholars can choose the suitable direction to explore. To the best of our knowledge, this is the first work to provide a comprehensive survey of GRL in the current literature.

Our paper makes notable contributions summarized as follows:

1) *Comprehensive review:* We provide a systematic and comprehensive overview of modern GRL methods for graph-structured data, and we categorize these methods according to research domains and characteristics of RL methods, which focus on solving data mining problems on graphs with RL.
2) *Current literature:* We provide a summary of recent works about GRL and an investigation of real-world applications. By doing the survey, we hope to provide a useful resource and global perspective for the graph mining and RL communities.
3) *Future directions:* We discuss five possible future directions in terms of automated RL, multi-agent RL, subgraph pattern mining, explainability, and evaluation metrics.
4) *Abundant resources:* We collect abundant resources on GRL, including state-of-the-art models, benchmark datasets, and open-source codes. We create a collection of papers for GRL, which contains numerous papers on GRL in recent years. In this collection, we provide links to the papers and code where available to allow scholars to get a faster overview of the research contents and methods and to provide more inspiration for research on GRL.

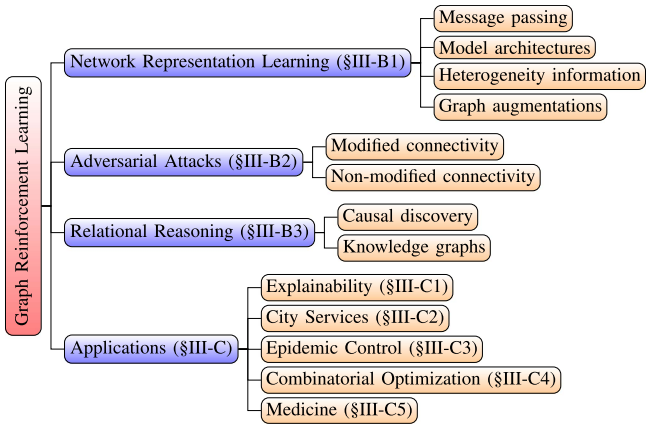

Fig. 2. Overview of Graph Reinforcement Learning in this survey.

The rest of this survey is organized as follows. Section II provides a brief introduction to the graph mining concepts discussed in this paper and a brief review of the key algorithms commonly used in GRL. In Section III, we review some current research on the topic of GRL and list the state, action, reward, termination, RL algorithms, and evaluation metrics. In Section IV, we discuss and propose some future research directions and challenges. Finally, we summarize the paper in Section V. For each aspect of GRL, we comprehensively summarise typical methods and provide methodology categorization in Fig. 2.

II. Preliminaries

Graphs are the natural abstraction of complex correlations found in numerous domains, different types of graphs have been the main topic in many scientific disciplines such as computer science, mathematics, engineering, sociology, and economics [22]. We summarize the common graph mining problems in GRL as follows. Furthermore, we provide a list of commonly used notations in papers in Appendix A.

A. Graph Neural Networks

Recently, Graph Neural Networks (GNNs) have been employed to model and compute graph-structured data to improve the ability to reason and predict relationships in graphs. These models have a strong relational induction to effectively learn the relationships between nodes and links in graphs and establish rules for graph-structured data, ranging from social media to biological network analysis and network modeling and optimization [23], [24], [25]. GNN is an essential tool to tackle graph mining problems, and most of the existing GRL methods employ GNN as the embedding method for networks/nodes/links. The basic principle of the GNN model can be concluded as follows: Given a graph $G = (A, X)$ with m nodes, the G can be represented by the adjacency matrix $A \in \{0, 1\}^{m\times m}$ and the feature matrix $X \in \mathbb{R}^{m\times d}$, where d indicates that each node corresponds to a d-dimensional feature vector, the information aggregation operation of GNN is defined as (1).

$$X_{i+1} = \sigma\left(D^{-\frac{1}{2}}\hat{A}D^{-\frac{1}{2}}X_iW_i\right) \tag{1}$$

where X_i denotes the input feature matrix of the i-th layer Graph Convolutional Network (GCN). $X_0 = X$ denotes the original feature vector of the input graph, and $X_i \in \mathbb{R}^{m\times d_i}$ is transformed into $X_{i+1} \in \mathbb{R}^{m\times d_{i+1}}$ by the message passing mechanism.[1] $\hat{A} = A + I$ denotes that self-loops are added to the adjacency matrix of the input graph. D denotes the diagonal node degree matrix after the normalization operation is executed on $\hat{A}$. In addition, $W_i \in \mathbb{R}^{d_i\times d_{i+1}}$ is the learnable weight matrix and $\sigma(\cdot)$ denotes the nonlinear activation function. Motivated by Weisfeiler-Lehman graph isomorphism test, learning a GNN includes three main components [26], [27]: (1) **Initialization.** Initialize the feature vectors of each node by the attribute of the vertices, (2) **Neighborhood Detection.** Determiner the local neighborhood for each node to further gather the information and (3) **Information Aggregation.** Update the node feature vectors by aggregating and compressing feature vectors of the detected neighbors.

Many various message passing mechanisms like the novel information aggregation functions [16], graph structure pooling [17], and neighborhood sampling methods [12] have been proposed to perform graph mining. Different GNN models provide high-level variability in the design of neighborhood detection and information aggregation, and these various models define the different node neighborhood information and the aggregation compression methods to achieve graph learning and they commonly consider that the features of the target nodes are obtained by aggregating and combining the features of their neighbors.

B. Network Representation Learning

Network representation learning has been recently proposed as a new learning paradigm to embed network nodes into a low-dimensional vector space, by preserving network topology structure, node content, and other side information.

[1]Message passing mechanism means that it aggregates neighbors' information and updates the representation of the center node.

The network representation learning is defined as: Given a network $G = (V, E, X, Y)$, where E is the set of edges in the network, V is the set of nodes in the network, X is the attribute feature of the nodes, and Y denotes the label of the nodes, the network structure with node labels is commonly employed for node classification tasks. The task of network representation learning is to learn a mapping function $\mathcal{F}$ as (2).

$$\mathcal{F} : v \to e_v \in \mathbb{R}^d \tag{2}$$

where e is the learned vector representation of node v, and $d \ll |V|$. The transformation $\mathcal{F}$ is employed to map the network structure into the low-dimensional vector space and preserve the topological and attribute information of the graph, to ensure that nodes with similar structure and attributes have higher proximity in the vector space.

In GRL, scholars have introduced RL into the design principles of network representation learning, and have formulated the network representation learning problem as the Markov Decision Process (MDP), they have also employed RL methods in multi-relational networks and heterogeneous information networks for constructing input data for network representation learning models or optimizing the selection scheme of neighbors.

C. Adversarial Attacks

The powerful GNN models for solving graph mining problems inherit the drawback that traditional Deep Neural Networks are easily fooled by adversarial attacks. The attacker can manipulate the graph structure or node features to create graph adversarial perturbations, which can fool the GNN model. The aim of the adversarial attack on the graph is to minimize the attack loss to misguide the victim model. We formulate the graph adversarial attack as: Given $G = (A, X, Y)$, and $V_t \subseteq V$ is the set of victim nodes in G. the task of an attacker is to find a perturbed graph $\hat{G} = (\hat{A}, \hat{X}, Y)$ that minimize the attack objective $\mathcal{L}$ as (3).

$$\begin{aligned} \min \quad & \mathcal{L}(f_\theta(G')) = \sum_{u\in V_t} \mathcal{L}_u(f_{\theta^*}(\hat{G})_u, Y) \\ s.t. \quad & \theta^* = \arg\min_\theta \mathcal{L}'(f_\theta(G')) \end{aligned} \tag{3}$$

where $\mathcal{L}_u$ is a loss function for attacking and G' can either be G or $\hat{G}$. $\Phi(G)$ denotes the space of perturbation on the original graph. We denote $\Phi(G)$ as (4).

$$||\hat{A} - A||_0 + ||\hat{X} - X||_0 \leq \Delta \tag{4}$$

where Δ is the fixed perturbation budget.

D. Knowledge Graphs

Knowledge Graphs (KGs), which preserve rich human knowledge and facts, are commonly employed in downstream artificial intelligence applications, and large-scale KGs such as DBpedia [28], Freebase [29], and Yago [30] are proved to be the infrastructure for tasks such as recommendation systems, dialogue generation [31], and Question and Answer (Q&A)

tasks. KG is generally formalized as $G = (h, r, t)$ representing (head entity, relation, tail entity), e.g., (Beijing, capital of, China), which preserves the multi-relationship graph with a large number of facts. The goal of KG completion is to infer the missing relationships and entities in KGs with explicit information. KG completion methods are divided into three categories [32]: (1) **Path ranking-based methods** [33]. Such methods are to a path to connect two entities as a feature to predict the relationship between two entities. (2) **Representation learning-based methods** [34], [35]. Such methods aim to represent the semantic information of the research object as a dense low-dimensional real-value vector and reason via vector operations. (3) **RL-based methods** [3]. These methods define the KG completion task as MDP.

E. Reinforcement Learning Methods

RL methods have recently achieved huge success in a variety of applications [36], which automatically tackles sequential decision problems in real-world environments via goal-directed learning and decision making, and such methods achieve great success in many real match games [37], [38].

The RL is formulated as an MDP, which is a sequential decision mathematical model in which actions affect the current short-term rewards, the subsequent states, and future rewards. MDP serves to simulate the random strategies and rewards of agents. In MDP, the prediction and control problem can be implemented by dynamic programming. The formal definition of MDP is the tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, p(s_0), \gamma\}$, where $\mathcal{S}$ denotes the set of all possible states, which is the generalization of the environment, $\mathcal{A}$ denotes the set of actions that can be adopted in the states, which is the all possible actions of the agent, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathcal{R}$ denotes the reward function, which is the rewards that the environment returns to the agent after executing an action, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to p(\mathcal{S})$ is identified by the state transition function, $\gamma \in [0, 1]$ denotes the discount factor, which can be treated as a hyperparameter of the agent and serves to promote the faster availability of short-term rewards to the agent. The process of interaction between agent and environment is shown in Fig. 3.

Our goal is to find a policy π that maximizes our expected return/action-value function $Q(s, a)$, the target policy is defined as (5) [39].

$$\begin{aligned}\pi^* &= \arg\max_{\pi} Q(s, a) \\ &= \arg\max_{\pi} E_{\pi,\mathbb{T}}\left[\sum_{k=0}^{K} \gamma^k r_{t+k} \mid s_t = s, a_t = a\right] \end{aligned} \tag{5}$$

where π^* is better or equal to all other policies.

Deep Reinforcement Learning (DRL) is poised to revolutionize the field of artificial intelligence and represents a step toward building autonomous systems with a higher-level understanding of the visual world [40]. Deep learning enables RL to scale to decision-making problems that were previously intractable, i.e., settings with high-dimensional state and action spaces.

In the following, we will provide fundamental concepts of different popular RL methods.

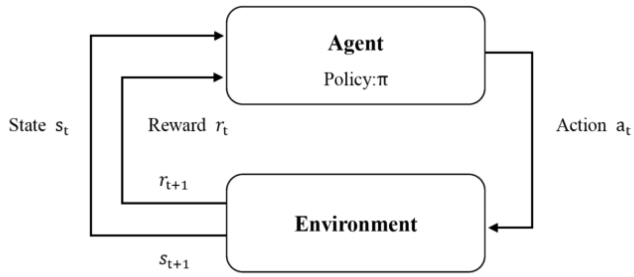

Fig. 3. The interaction process between agent and environment. The interaction process between agent and environment is described as: The first state is sampled from the initial state distribution $p(s_0)$. At timestep t, the agent makes a calculated decision based on the state $s_t \in \mathcal{S}$ returned from the environment and chooses the corresponding action $a_t \in \mathcal{A}$, which will be applied to the environment, resulting in the changes in the environment. The agent obtains the state representation $s_{t+1} \sim \mathcal{T}(\cdot \mid s_t, a_t)$ and a reward $r_t = \mathcal{R}(s_t, a_t, s_{t+1})$ according to the changes in the environment. The agent acts in the environment according to a policy $\pi : \mathcal{S} \to p(\mathcal{A})$. By repeatedly selecting actions and transitioning to a next state, we can sample a trace $\{s_0, a_0, r_0, s_1, a_1, r_1, \ldots\}$ through the environment.

1) Q-Learning: Q-learning [41] is an off-policy learner and Temporal Difference (TD) learning method, which is an important result of early research on RL. The target policy in Q-learning updates the values directly on the Q-table to achieve the selection of the optimal policy, whereas, the behavior policy employs the ε-greedy policy for semi-random exploration of the environment. The learning goal of the action-value function Q to be learned in Q-learning is that the optimal action-value function q^* is learned by direct approximation, so that the Q-learning algorithm can be formulated as (6).

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[\mathcal{R}_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right] \tag{6}$$

where the equation denotes that state s_t explores the environment with a behavior policy based on the values in the Q-table at timestep t, i.e., it performs action a, the reward R and a new state s_{t+1} is obtained based on the feedback from the environment. The equation is executed to update to obtain the latest Q-table, it will continue to update the new state s_{t+1} after completing the above operation until the termination time t.

2) REINFORCE: REINFORCE [42] does not optimize directly on the policy space but learns the parameterized policy without the intermediate value estimation function, which uses the Monte Carlo method for learning the policy parameters with the estimated returns and the full trace. The method creates a neural network-based policy that takes states as inputs and generates probability distributions in the operation space as outputs. The policy π is parameterized with a set of weights θ so that $\pi(s; \theta) \equiv \pi(s)$, which is the action probability distribution on the state, and REINFORCE is updated with (7).

$$\Delta\omega_{i,j} = \alpha_{i,j}(r - b_{i,j}) \frac{\vartheta}{\vartheta\omega_{i,j}} \ln g_i \tag{7}$$

where $a_{i,j}$ is a non-negative learning factor, r denotes the discounted reward value, and $b_{i,j}$ is a representation function of the state for reducing the variance of the gradient estimate. Williams [42] points that $b_{i,j}$ could have a profound effect on the

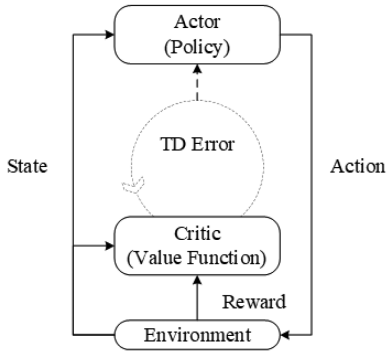

Fig. 4. The framework of Actor-Critic algorithm. Actor receives a state from the environment and selects an action to perform. Meanwhile, Critic receives the current state and the state generated by the previous interaction and calculates the TD error to update Critic and Actor.

convergence speed of the algorithm. g_i is the probability density function for randomly generating unit activation-based actions.

3) Actor-Critic: Actor-Critic algorithm [43] employs a parameterized policy and a value function, where the value function provides a better $\hat{A}(s,a)$ estimate for the computation of the policy gradient. The Actor-Critic algorithm studies both policy and state value functions, combining the advantages of value function-based and policy gradient-based algorithms. In this method, the Actor denotes the policy function for learning the policy that can obtain as many rewards as possible, and the Critic denotes the estimated value function for evaluating the estimated value of the current policy. Fig. 4 is the framework of the Actor-Critic algorithm.

The Actor-Critic architecture is widely employed as the basic framework for RL algorithms like Asynchronous Advantage Actor-Critic (A3C) [44], Advantage Actor-Critic (A2C) [45], Deterministic Policy Gradient (DPG) [46], Deep Deterministic Policy Gradient (DDPG) [47]. The A3C develops for single and distributed machine setups combines advantage updates with an actor-critic formulation that relies on asynchronous update policy and value function networks trained in parallel on several processing threads [40].

4) Deep Q-Network: Q-table updates with the Q-learning algorithm in large-scale graph-structured data suffer from the problem of a high number of intermediate state values, leading to dimensional disasters. To address the above challenges, the main method proposed by scholars is the Deep Q-Network (DQN) [2] which combines value function approximation and neural networks through function approximation and state/action space reduction. DQN is widely exploited to learn policies by leveraging deep neural networks.

The scholars propose to employ Q-learning algorithms to represent states with graph embeddings according to the property of graph-structured data in order to minimize the impact of non-Euclidean structures on the Q-table scale. Overall, DQN combines deep learning with Q-learning and approximates the action value function with deep neural networks, and obtains the trace $\{s_t, a_t, r_t, s_{t+1}\}$ from the interaction of the agent with the environment. This method can learn successful policies directly from high-dimensional sensory inputs using end-to-end RL [2]. The loss function $L(\theta_t)$ of this method is given by (8).

$$L(\theta_t) = \mathbb{E}\left[\left(r + \gamma \max_{a+1} Q_{\theta_{t-1}}(s_{t+1}, a_{t+1}) - Q_{\theta_t}(s_t, a_t)\right)^2\right] \tag{8}$$

DQN introduces two techniques to stabilize the training process: (1) A replay buffer to reuse past experiences. (2) A separate target network that is periodically updated. DQN has many variations to improve the current version. Double Deep Q-Network (DDQN) [48] reduces the risk of high estimation bias in Q-learning by decoupling selection and evaluation. the DDQN addresses the problem of not sufficiently considering the importance of different samples in the DQN algorithm by calculating the priority of each sample in the experience pool and increasing the probability of valuable training samples. Moreover, scholars have proposed more improved versions of DQN [49], [50] to address problems such as the lack of long-term memory capability of DQN.

III. Reinforcement Learning on Graphs

Existing methods to solve graph mining problems with RL methods focus on network representation learning, adversarial attacks, and relational reasoning. Furthermore, many real-world applications study the GRL problem from different perspectives. We provide a detailed overview of the research contents in GRL in this section.

A. Datasets & Open-Source

1) Datasets: We provide a summary of the experimental datasets employed in these studies cited in this survey and a list of the statistical information of datasets, sources, tasks, and citations, as shown in Table I and II. In addition, we summarize the experimental results of node classification on Cora [51], CiteSeer [51], and PubMed [51] and the results of KG completion on NELL-995 [3], WN18RR [52], and FB15K-237 [53], where the classification accuracy is the evaluation metric for the node classification task and Hit@1, Hit@10, and MRR are the evaluation metrics for the KG completion, as shown in Table III and IV.

2) Open-Source: To facilitate research into GRL, we provide a collection of papers on GRL models and datasets for the understanding of the fusion works of graph mining and RL methods, which are available in public.[2] The papers in this collection cover various domains of graph mining problems. We believe that this collection of papers provides a comprehensive and effective summary of methods in the field of GRL.

Furthermore, we summarize the open-source implementations of GRL methods reviewed in the survey and provide the links to the codes of these models in Appendix B.

[2][Online]. Available: https://github.com/neunms/Reinforcement-learning-on-graphs-A-survey

TABLE I
COMMONLY USED DATASETS FOR GRL

Dataset	$\|V\|$	$\|E\|$	Source	Citation
Cora*	2708	5429	[51]	Policy-GNN [27], RL-S2V [19], GraphNAS [54], GPA [14], AGNN [55], GDPNet [56], NIPA [57], RLNet [58]
CiteSeer	3312	4732	[51]	Policy-GNN [27], RL-S2V [19], GraphNAS [54], SparRL [22], GPA [14], AGNN [55], GDPNet [56]
Facebook	4039	88234	[59]	SparRL [22], IMGER [60], RLNet [58]
Amazon	11944	4398392	[61]	RioGNN [62], CARE-GNN [63], SparRL [22]
FB15K-237*	14541	237	[53]	RF [64], AttnPath [65], RLH [66], PAAR [67], Zheng *et al.* [68], ADRL [69], GRL [32], ConvE [70], MINERVA [71], KBGAN [72], RLPath [73], DAPath [74], MemoryPath [75]
PubMed	19717	44338	[51]	Policy-GNN [27],NIPA [57], RL-S2V [19], GraphNAS [54], GPA [14], AGNN [55], GDPNet [56]
WN18RR	40943	11	[52]	RF [64], RLH [66], Zheng *et al.* [68], ADRL [69], GRL [32], MARLPaR [76], ConvE [70], MINERVA [71], KBGAN [72]
Yelp	45954	3846979	[77]	RioGNN [62], CARE-GNN [63], UNICORN [78]
NELL-995*	75492	237	[3]	RF [64], AttnPath [65], RLH [66], PAAR [67], Zheng *et al.* [68], ADRL [69], GRL [32], MARLPaR [76], ConvE [70], MINERVA [71], RLPath [73], DAPath [74], MemoryPath [75]
Twitter*	81306	1768149	[59]	SparRL [22], AdRumor-RL[24], IMGER [60]
YouTube	1134890	2987624	[79]	SparRL [22], IMGER [60]

Note: * These datasets are used at different scales, yet they share the same data sources. We marked the maximum number of nodes and edges.

TABLE II
COMMONLY USED DATASETS FOR GRL IN GRAPH CLASSIFICATION

Dataset	# Graphs	# Nodes(Avg.)	# Edges(Avg.)	Source	Citation
MUTAG	188	17.93	19.79	[80]	SUGAR [15], SubgraphX [81], XGNN [18], GraphAug [82]
PTC	344	14.29	14.69	[83]	SUGAR [15]
PROTEINS	1113	39.06	72.82	[84]	SUGAR [15], GraphAug [82]
BBBP	2039	24.06	25.95	[85]	SubgraphX [81]
NCI1	4110	29.87	32.30	[86]	SUGAR [15], GraphAug [82]
NCI109	4127	29.68	32.13	[86]	SUGAR [15], GraphAug [82]

TABLE III
REPORTED EXPERIMENTAL RESULTS FOR NODE CLASSIFICATION ON THREE FREQUENTLY USED DATASETS

Method	Cora	CiteSeer	PubMed
GCN [87]*	81.5±0.0	70.3±0.0	79.0±0.0
GAT [88]*	83.0±0.7	72.5±0.7	79.0±0.3
LGCN [89]*	83.3±0.5	73.0±0.6	79.5±0.2
DGI [90]*	82.3±0.6	71.8±0.7	76.8±0.6
Policy-GNN [27]	91.9±1.4	89.7±2.1	92.1±2.2
GraphNAS [54]	-	73.1±0.9	79.6±0.4
AGNN [55]	83.6±0.3	73.8±0.7	79.7±0.4

Note: * These methods are non-GRL methods.
Missing values ("-") in this table indicate that this method has no experimental results reported on the specific dataset.

TABLE IV
REPORTED EXPERIMENTAL RESULTS FOR KG COMPLETION ON THREE FREQUENTLY USED DATASETS

Method	Metrics	NELL-995	WN18RR	FB15K-237
TransE [34]*	Hit@1	0.608	0.491	0.392
	Hit@10	0.793	0.626	0.614
	MRR	0.715	0.557	0.494
TransR [91]*	Hit@1	0.631	0.526	0.405
	Hit@10	0.806	0.641	0.634
	MRR	0.727	0.581	0.532
ComplEX [92]*	Hit@1	0.614	0.319	0.318
	Hit@10	0.815	0.462	0.542
	MRR	0.652	0.428	0.374
RLH [66]	Hit@1	0.692	0.453	0.342
	Hit@10	0.873	0.516	0.648
	MRR	0.723	0.481	0.460
ADRL [69]	Hit@1	0.808	0.683	0.574
	Hit@10	0.975	0.723	0.796
	MRR	0.916	0.704	0.712
ConvE [70]	Hit@1	0.672	0.402	0.313
	Hit@10	0.864	0.519	0.601
	MRR	0.747	0.438	0.410
MINERVA [71]	Hit@1	0.663	0.413	0.217
	Hit@10	0.831	0.513	0.456
	MRR	0.725	0.448	0.293

Note: * These methods are non-GRL methods.

B. Graph Mining With Reinforcement Learning

1) Network Representation Learning: Network representation learning is to learn a mapping that embeds the nodes of a graph as low-dimensional vectors while encoding a variety of structural and semantic information. GNNs receive increasing interest due to their effectiveness in network representation learning. Generally, the message passing architecture predefined by scholars is probably not applicable to all nodes in graphs, and the multi-hop message passing mechanism is poor for passing important information to other modules. Therefore, the novel and effective node sampling strategies and innovation of new message passing mechanisms for graph-structured data attract more attention from scholars [16], [58], [93], [94], [95]. Batch sampling and importance sampling are commonly employed in GNNs to obtain the local structure of graph-structured data, however, such sampling methods may cause information loss. Lai et al. [27] propose Policy-GNN to define the meta-policy module and GNN module for learning relationships and network representations between node attributes and aggregation iterations. Policy-GNN solves the challenge of determining the aggregation range of nodes in large-scale networks with the DQN algorithm. SUGAR [15] focuses on the finer substructures' semantics and proposes GNNs for hierarchical subgraph-level selection and embedding to learn more discriminative subgraph representations, which reduces the loss of representation quality from GNN sampling strategies. However, the constraints, features, and optimization procedures in problem solving are determined manually, Yan et al. [21] employ the popular parallel policy gradient training method to ensure the efficiency and robustness of the sampling training experience while optimizing the learning agent and employ GCN to automatically extract spatial features in an irregular graph topology in learning agent. Furthermore, the hierarchical structure of the network is crucial

for exploring the requirements of different downstream tasks for feature and topological information. ACE-HGNN [96] leverages multi-agent RL method to learn the optimal curvature of the network to improve model quality and generalizability.

Existing representation learning methods based on GNNs and their variants depend on the aggregation of neighborhood information, which makes them sensitive to noise in the graph, and several scholars have improved the performance of network representation learning in terms of removing noisy data from the graph. GDPNet [56] employs two phases of signal neighborhood selection and representation learning to remove noisy neighborhoods as the MDP to optimize the neighborhood of each target node, and learns a policy with task-specific rewards received from the representation learning phase, allowing the model to perform graph denoising just with weak supervision from the task-specific reward signals. GAM [97], which also focuses on noisy graph-structured data, enables to limit the learning attention of the model to small but informative parts of the graph with attention-guided walks, The graph attention model employs the Partially Observable Markov Decision Process (POMDP) method for training to selectively process informative portions of the graph. Moreover, NetRL [98] performs network enhancement with detecting noisy links and predicting missing links to improve network analysis and modeling capabilities.

Slight modifications to the GNN architecture can modify the learning capability of the model. The design principles of the architecture require substantial domain knowledge to guide, e.g., the manual setting of hidden dimensions, aggregation functions, and parameters of the target classifier. GraphNAS [54] designs a search space covering sampling functions, aggregation functions, and gated functions and searches the graph neural architectures with RL. The algorithm first uses a recurrent network to generate variable-length strings that describe the architectures of GNNs. Furthermore, AGNN [55], which has the same objective as GraphNAS, verifies the architecture in the predefined search space via small steps with RL-based controllers, decomposing the search space into the following six classes of actions: Hidden dimension, Attention function, Attention head, Aggregate function, Combine function, and Activation function.

However, the above algorithms fail to consider heterogeneous neighborhoods in the aggregation of nodes [62], causing neglect or simplification of the diverse relationship of nodes and edges in real networks, which brings a loss of heterogeneity information containing multiple types of nodes and relationships. Relational GNNs based on artificial meta-paths [99] or meta-graphs [100] rely on inherent entity relationships and require the support of substantial domain knowledge. Zhong et al. [101] solve the dependence on the handcrafted meta-paths via proposing the RL enhanced heterogeneous GNN model to design different meta-paths for nodes in heterogeneous information networks, and replacing the manual design of meta-paths with agent. This method discovers many meaningful meta-paths that are ignored by human knowledge. Furthermore, a novel neighborhood aggregation technique is adopted to effectively improve the performance of heterogeneous network representation learning. RioGNN [62] learns multi-relational node representations with the semi-supervised method, which leverages relational sampling, message passing, metric learning, and RL to guide neighbor selection within and between different relations. These above works further illustrate the massive role of RL methods in multi-relational/heterogeneous networks.

The graph-structured data augmentations provide further improvements in terms of the performance of network representation learning, and scholars believe that invariance of the learning mechanism is critical to data augmentations. GraphAug [82] optimizes the model to maximize the estimated label-invariance probability via RL, which results in label-invariance augmentations. Experimental results show that GraphAug outperforms existing graph augmentation methods. More GRL methods on network representation learning could be found in Table V.

2) Adversarial Attacks: Recent studies have shown that attackers can perturb or poison the graph-structured data used for training to affect GNNs in node classification. Existing researches on adversarial attacks on GNNs focus on modifying the connectivity between two existing nodes. RL-S2V [19] learns how to modify the graph structure by sequentially adding or removing links to the graph only using feedback from downstream tasks. The RL-S2V architecture is shown in Fig. 5.

However, such attack policies that require frequent modifications to the network structure need a high level of complexity to avoid the attack being caught. NIPA method avoids manipulating the connections between existing nodes, but injects the fake nodes into the graph. This algorithm represents the adversarial connections and adversarial labels of the injected false nodes as MDP and designs an appropriate reward function to guide the RL agent to reduce the node classification performance of GNNs. The graph rewiring method [102] also avoids adding or removing connections resulting in significant changes to the graph structure.

Adversarial attack research on GNNs also includes fraud entity detection tasks, such as opinion fraud and financial fraud [63], [103]. More GRL methods on adversarial attacks could be found in Table VI.

3) Relational Reasoning: Discovering and understanding causal mechanisms could be defined as searching for Directed Acyclic Graphs (DAGs) that minimize the defined score functions. RL methods have achieved excellent results in terms of causal discovery from observed data, but searching the DAGs space or discovering the implied conditions usually has high complexity. The agent in RL with random policies could automatically define the search policy based on the learn the uncertain information of the policy, which can be updated rapidly with the reward signal [104]. However, this approach results in poor computational accuracy, and the action space composed of directed graphs is commonly difficult to be explored completely. CORL [105] describes the ordering search problem as a multi-step MDP and implements the ordering generation process with encoder-decoder structures, and optimizes the proposed model with RL based on the reward mechanism designed for each ordering. Finally, the causal graph is obtained by processing the results of the ordering. Furthermore, Sun et al. [106] combine transfer learning and RL for co-learning to leverage the prior causal knowledge for solving causal reasoning tasks.

TABLE V
GRL METHODS ON NETWORK REPRESENTATION LEARNING TASKS

Method	Action	State	Reward	Termination	RL	Metrics
SUGAR [15]	$+/-$ a fixed value $\Delta k \in [0, 1]$ from k.	The middle graph consists of the subgraphs selected in each training epoch.	The discrete reward function based on the classification accuracy of the previous epoch.	The change of k among ten consecutive epochs no more than Δk.	Q-learning	Average accuracy, Standard deviation, Training process, Testing performance, and Result visualization.
RL-HGNN [101]	The set of available relation types.	All nodes involved in the meta-path.	The performance improvement on the specific task compared with the historical performances.	-	DQN	Micro-F1, Macro-F1, and Time-consuming
Policy-GNN [27]	The number of hops of the current node.	The attribute of the current node.	The performance improvement on the specific task compared with the last state.	-	DQN	Accuracy
GraphNAS [54]	Select the specified parameters for the GNN architecture.	GNN architecture.	Define the corresponding reward value based on the special task accuracy.	To maximize the Expected accuracy of the generated architectures.	MDP	Micro-F1 and Accuracy
GPA [14]	Select a node v_t from $V_{train}/V_{label}^{t-1}$ at timestep t.	The state S_G^t in graph G at step t is defined as a concatenation of several commonly used heuristics criteria in active learning.	The classification GNN's performance score on the validation set after convergence.	Performance convergence of classifiers.	MDP	Micro-F1 and Macro-F1
GDPNet [56]	Select neighbors.	Embedding of information about the current node and its neighbors.	Define the corresponding reward value based on the special task accuracy.	-	MDP	Accuracy

Missing values ("-") in this table indicate that the personalized termination conditions contribute to the stabilization of the training process.

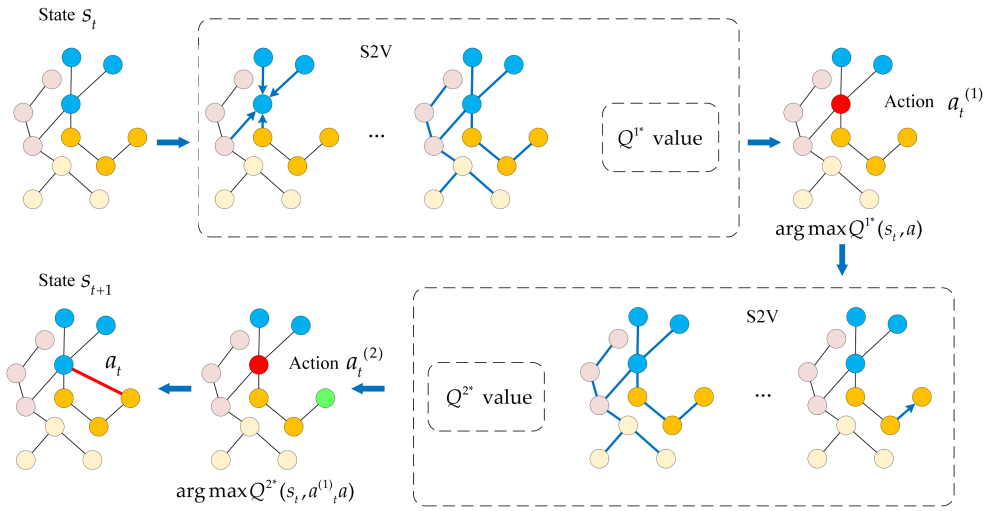

Fig. 5. An illustration of the RL-S2V architecture [19]. The algorithm trains the original graph structure with Structure2Vec (S2V) to obtain the representation of each node and then parameterizes each node with a GNN to obtain the Q-value, and the candidate links that need to modify are obtained with Q-learning to achieve a black-box attack, which is formulated as follows: RL-S2V employ S2V for calculating Q-value, and the adding a single edge a_t is decomposed into two decision steps $a^{(1)}$ and $a^{(2)}$, with Q^{1*} and Q^{2*}. (Redrawn from [19]).

TABLE VI
GRL METHODS ON ADVERSARIAL ATTACKS TASKS

Method	Action	State	Reward	Termination	RL	Metrics
NIPA [57]	Add the adversarial edges within the injected nodes between the injected nodes and the clean node and designing the adversarial labels of the injected nodes.	The intermediate poisoned graph and labels information of the injected nodes.	Guiding reward based on classifier success rate.	The agent adds the budget number of edges.	DQN	Accuracy, Key statistics of the poisoned graphs, Average Degrees of Injected Nodes, and Sparsity of the Origin Graph.
RL-S2V [19]	Add or delete edges in the graph.	The modified graph.	The prediction confidence of the target classifier.	The agent modifies the specified number of links.	Q-learning	Accuracy
CARE-GNN [63]	$+/-$ a fixed small value.	The selection range of neighbor nodes.	The average distance differences between two consecutive epochs.	The RL converges in the recent ten epochs and indicates an optimal threshold until the convergence of GNN.	BMAB	ROC-AUC and Recall
ReWatt [102]	The valid rewiring operations	The intermediate graphs generated after all the possible rewiring operations.	The reward function is developed based on the performance of the classifier.	The number of actions reaches the budget K or the attacker successfully changed the label of the slightly modified graph.	MDP	Attacking performance

Task-oriented spoken dialogue system is a system that can continuously interact with a person to accomplish a predefined task. However, many methods are not sample-efficient. An appropriate method [107] employs a neural network structure for DRL-based dialogue management, which consists of several sub-networks corresponding to nodes on a directed graph defined according to the domain ontology, and the joint optimization of graph structure and parameters is also adopted to speed up internal message exchange. Furthermore, several natural question generation models provide more training data for the Q&A task to further improve the performance of the task. Graph2Seq [108] provides a generator for embedding text leveraging a Bidirectional Gated GNN and a hybrid evaluator for generating valid text. The algorithm employs the self-critical sequence training algorithm to directly optimize the evaluation metrics.

KGs are the data infrastructure for downstream real-world applications and are increasingly being used for dialogue systems [107], KG completion [64], recommendation systems [109], social networks [110] and information extraction. However, false-negative training data in incomplete KGs will affect the decision-making of the model [70], and fixed multi-hop or single-hop methods [71] consume massive computational resources. KGQA [64] employs dynamic rewards to build an end-to-end dynamic KG reasoning framework. Long short-term memory (LSTM) and MDP are adopted to model path inference, and Actor-Critic is employed to optimize model parameters.

Most existing methods focus on reasoning in static KGs to complete missing facts, However, the introduction of temporal features enables more reliable modeling of the relationships between facts in the real world. TITer [111] captures temporal information about the agent with a relative time encoding function and guides model learning with time-shaped rewards. This explainable method has fewer computations and parameters compared to the benchmarks.

Motivated by the hierarchical structure commonly employed by humans when dealing with fuzzy scenarios in cognition, Wan et al. [66] suggest a hierarchical RL method to simulate human thinking patterns, where the whole reasoning process is decomposed into two steps of RL policies. Furthermore, the policy gradient approach [112] and REINFORCE [42] are employed to select two policies for encoding historical information and learning a structured action space. DeepPath [3] for hierarchical reasoning of KGs based on RL requires manual rules to process for obtaining more adequate hierarchical relationships. On the other hand, PAAR [67] employs a multi-hop reasoning model based on hyperbolic KG embedding and RL for hierarchical reasoning. Hierarchical information of the KG plays an important role for downstream tasks, and another RL method [68] employing hierarchical information is to reason about the KG from a methodological perspective.

Interactive recommender system is one of the practical applications of relational reasoning, which has received substantial attention as its flexible recommendation policy and optimal long-term user experience, and scholars have introduced DRL models such as DQN [113] and DDPG [47] into the interactive recommender system for decision-making and long-term planning in dynamic environments. However, these RL methods require extensive interaction data for training, which causes a large action space. KGQR [109] enables to incorporate graph learning and sequential decision problems in interactive recommender systems as a whole, to select candidate objects and learn user preferences from user feedback with prior knowledge, which is capable of obtaining better recommendation performance with fewer user-item interactions. The KGQR architecture is shown in Fig. 6. More GRL methods on relational reasoning could be found in Table VII.

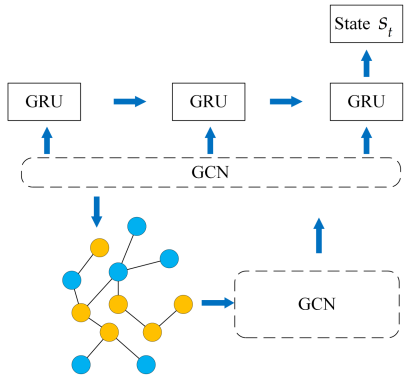

Fig. 6. An illustration of the KGQR architecture [109]. The knowledge-enhanced state representation module preserves user preferences with recurrent neural network and GNN. At each timestep t, the algorithm sequentially recommends items to the user, and the agent updates the recommendation policy based on the user's feedback r_t. The algorithm uses the GCN module and the state representation module to model the user's preference s_t based on the interaction history o_t. Finally, the algorithm calculates the item with the highest score in the candidate set as the recommended item through the Q-network. (Redrawn from [109], and we skip the candidate selection module and Q-value network in KGQR.).

C. Real-World Applications With Reinforcement Learning on Graphs

The research on GRL is booming in the past few years, and GRL methods play an important role in solving various real-world applications and attract substantial attention from scholars. We summarize abundant real-world applications in transportation network optimization, recommendation systems in E-commerce networks, drug structure prediction and molecular structure generation in medical research, and network diffusion models for controlling the COVID-19 virus to further prove the hot development in the field of GRL. More GRL methods on real-world applications could be found in Table VIII.

1) Explainability: In exploring GNNs, scholars often treat them as black boxes, and this assumption lacks explanations that are readily understood by humans, which prevents their use in critical applications involving medicine, privacy, and security [115], [116]. A great number of scholars working on deciphering the explainability of such black-box models of GNNs commonly focus on explaining the explainability of the node, edge, or node feature-level in the graph-structured data, but ignore the GNN model and the frequent subgraph. The interpretation at the subgraph-level enables the more intuitive

TABLE VII
GRL METHODS ON RELATIONAL REASONING TASKS

Method	Action	State	Reward	Termination	RL	Metrics
TITer [111]	The set of optional actions is sampled from the set of outgoing edges.	The state space is represented by a special quintuple.	The dynamic reward function is defined based on the search results.	-	MDP	Mean Reciprocal Rank (MRR), Hits@1/3/10
RLH [66]	The set of outgoing edges of the current entity.	The tuple of entities and the relationships between them.	The reward for each step is defined based on whether the agent reaches the target entity.	-	REINFORCE	Mean average precision (MAP) and MRR
CORL [105]	The variables at each decision step	The embedding of the input data pre-processed with the encoder module.	The reward function consists of episodic and dense rewards.	-	MDP	True Positive Rate (TPR), Structural Hamming Distance (SHD)
IMUP [114]	The actions are defined as the visit event.	The state is to describe the environment composed by users and the spatial knowledge graph.	The reward is defined as the weighted sum of three parts about Points of Information (POI).	-	DQN	Precision on Category, Recall on Category, Average Similarity, and Average Distance
ADRL [69]	The set of outgoing edges	The state space consisting of the source entity, the query relationship and the entity that is accessed.	The rewards depend on the value function.	-	A3C	MRR and Hits@1/10
GRL [32]	The agent can select its neighboring relational path to another neighboring entity at each state.	The vector representation of entities and relationships obtained with GNN.	The Adversarial rewards are defined by employing Wasserstein-GAN.	The agent reaches the target entity	DDPG	MRR, Hits@1/10, MAP, and MAE

Missing values ("-") in this table indicate that the personalized termination conditions contribute to the stabilization of the training process.

TABLE VIII
GRL METHODS ON REAL-WORLD APPLICATIONS

Method	Action	State	Reward	Termination	RL	Metrics
XGNN [18]	Add an edge to current state G_t at step t.	The partially generated graph G_t at step t.	The reward consists of guidance from the trained GNNs and validated graph rules.	-	MDP	Accuracy and Metrics for explanation
Marl-eGCN [10]	The toll set by the transit authority on the special road that has an ETC gantry.	The number of vehicles that travel to the special zone on road.	The number of vehicles which arrive at destinations.	-	MDP	Traffic throughput
GTPN [9]	The action consists of three consecutive sub-actions.	The intermediate graph.	The reward is determined based on whether the model is successful in predicting	-	A2C	Coverage@k, Recall@k, and precision@K
IG-RL [124]	The action is defined as whether to switch to the next phase or prolong the current phase.	Current connectivity and demand in the network.	The reward for a given agent is defined as the negative sum of local queue lengths.	An episode either ends as soon as all trips are completed or after a fixed amount of time.	Q-learning	Trips Duration and Total Delay Evolution
DeepOpt [145]	The action value can be denoted with a binary list indicating the selection status of each network node.	The state information including the input of the node attributes and edge attributes.	The reward function consists of custom penalty terms.	-	MDP	The Reject Ratio of Service Function Chain Requests, Influence of Topology Change, and Time Consumption
A3C+GCN [21]	An action is a valid embedding process that allocates virtual network requests onto a subset of substrate network components.	The state is the real-time representation of special network status.	Reward shaping	-	A3C	Acceptance Ratio, Long-Term Average Revenue, and Running Time
GCN-RL [146]	The continuous action space is defined to determine the transistor sizes.	The state consisting of transistor index, component type, and the selected model feature vector for the component.	The reward is defined as a weighted sum of the normalized performance metrics.	-	DDPG	FoM and the performance of Voltage Amplifier and Voltage Amplifier
GMETAEXP [147]	Each action is a test input to the program, which can be text (sequences of characters) or images (2D array of characters).	The state contains the full interaction history in the episode.	Customized step-by-step rewards.	Once the agent has visited all nodes or met a custom training termination condition	MDP	Coverage performance and generalization performance

Missing values ("-") in this table indicate that the personalized termination conditions contribute to the stabilization of the training process.

and effective description of the GNN [117], [118], [119]. SubgraphX [81] proposes to explain GNNs by identifying important and intuitive substructures of graphs. which employs a Monte Carlo tree search to explicitly explore different subgraphs of the given input graph, and for each subgraph, we propose to use Shapley values to measure its importance by considering the interactions between different subgraph structures. Qualitative and quantitative experiments prove the effectiveness and efficiency of SubgraphX. The SubgraphX architecture is shown in Fig. 7.

Furthermore, another approach [62] to learning the differential features in graph-structured data achieves the explainability of multi-relational GNNs from the perspective of the individual importance of different relations. This approach constructs a multi-relational graph to reflect the heterogeneity of nodes, edges, attributes, and labels based on the practical task and employs RL to optimize the neighbor selection process when embedding aggregated neighbor information for the central node. Finally, all neighborhood information from different relations is aggregated to obtain the eventual node embedding. The scholars believe that explaining GNNs from the model level could enhance human trust for some application domains. XGNN [18] explains GNNs by formulating graph generation as RL, and the generated graph structures enable verification, understanding, and improvement of trained GNN models.

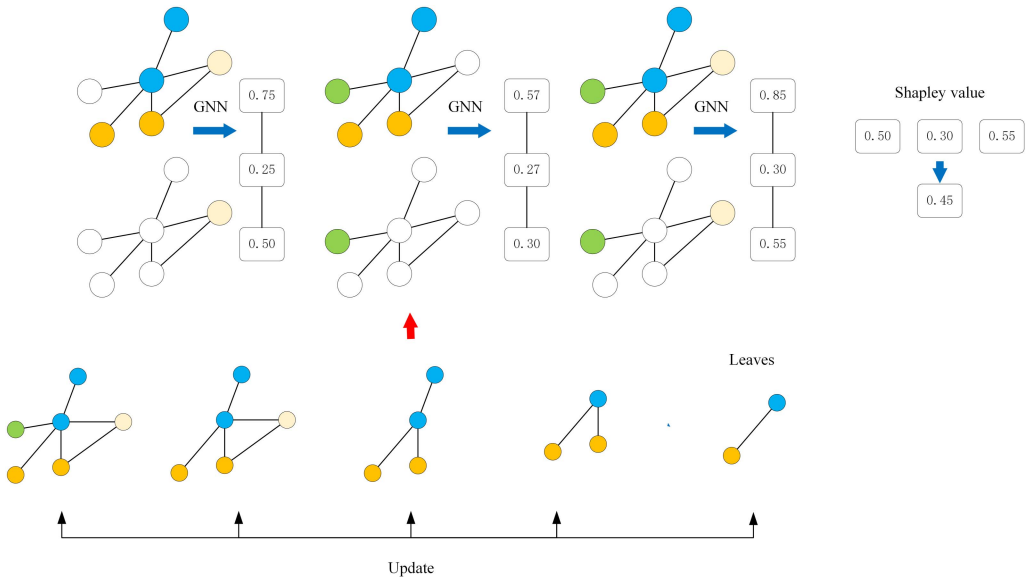

Fig. 7. An illustration of the SubgraphX architecture [81]. In SubgraphX, the algorithm selects a specified path from the search tree in root-to-leaf order corresponding to one iteration of Monte Carlo Tree Search (MCTS). The root of the search tree is associated with the input graph. Each node corresponds to a connected subgraph in which Shapley values are employed to measure the importance of interactions between different subgraph structures. Furthermore, the edges in the search tree represent pruning actions$a$. The MCTS algorithm records the statistics of the number of visits and rewards to guide the search of the agent in the environment. (Redrawn from [81]).

2) City Services: The growth of modern cities and urban populations causes problems like traffic congestion and inefficient communication. Several scholars have modeled road networks and communication networks as graph-structured data that can provide support to government agencies or telecommunication service providers to better optimize traffic roads or communication links [120].

Traffic flow prediction focus on building some prediction models to estimate the traffic flow of specific roads based on the statistical information of traffic flow within or between these regions and traffic data provided by relevant government institutions or organizations. Peng et al. [121] address the problem of incompleteness and non-temporality of most traffic flow data by modeling existing traffic flow graphs with GCPN [122] and extracting temporal features with LSTM [123]. IG-RL [124] for dynamic transportation networks focuses on the traffic signal control problem, which uses GCN to learn the entities as node embeddings, and the Q-learning algorithm is employed to train the model. This method enables the representation and utilization of traffic demand and road network structure in an appropriate way regardless of the number of entities and their location in the road network.

Electronic Toll Collection (ETC) system serves an important role in alleviating the urban traffic congestion problem. Qiu et al. [10] employ a GCN to represent the value function and policy function in the Dynamic Electronic Toll Collection (DETC) problem and solve the DETC with collaborative multi-agent RL algorithm. Furthermore, GRL-based applications like optimization of on-demand ride-sharing services [125], lane change decision for connected autonomous vehicles [126], [127], and adaptive traffic signal control [128], [129] could also improve the operational efficiency of traffic flow in cities.

Topology learning and channel allocation in Wireless Local Area Networks (WLANs) are crucial to improve the quality of wireless communication, but the traditional solutions cannot tackle the above problems. Almasan et al. [130] propose to integrate GNNs with the DQN. The proposed DRL+GNN architecture enables learning and generalize over arbitrary network topologies, and is evaluated in SDN-based optical transport network scenarios. For WLANs, Nakashima et al. [131] propose a DRL-based channel allocation scheme. This algorithm employs a graph convolutional layer to extract features of channel vectors with topological information and learns DDQN [48] for channel allocation policy formulation.

3) Epidemic Control: In the fields of epidemic containment, product marketing, and fake news detection, scholars have employed a limited number of interventions to control the observed dynamic processes partially on the graph. Meirom et al. [132] proposes to define the diffusion process on the temporal evolving graph as a POMDP, and employs a GNN to calculate the score of each node, where each node is contained in its m-hop neighborhood. A high-quality subset of nodes is then selected by ordering the nodes' scores, where the Actor-Critic algorithm is adopted and used to dynamically intervene for the selection of a subset of nodes. In another epidemic control application, the RAI algorithm [133] leverages social relationships between mobile devices in the Social Internet of Things (SIoT) to assist in controlling the spread of the virus by allocating limited protection resources to influential individuals through early identification of suspected COVID-19 cases. Furthermore, the algorithms for

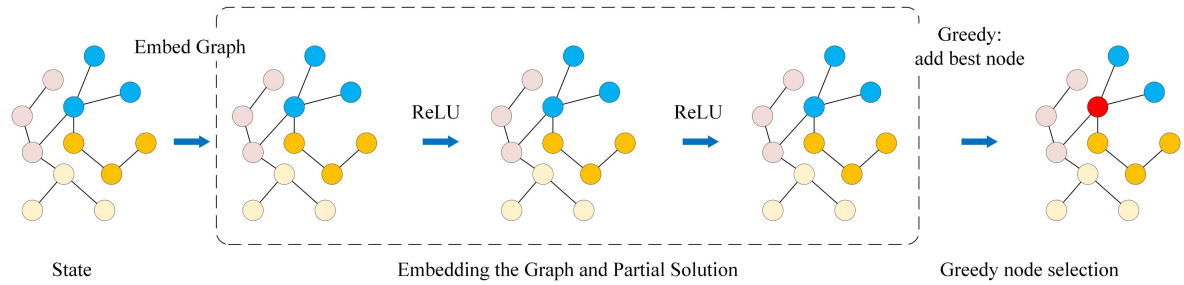

Fig. 8. An illustration of the S2V-DQN architecture [138]. This framework illustrates an instance of the S2V-DQN framework for Minimum Vertex Cove (MVC). Formally, given a graph $G = (V, E)$, our goal is to find a subset of nodes $V' \subseteq V$, where $(u, v) \in E \Leftrightarrow u \in V'$ or $v \in V'$, and $|V'|$ is minimized. S2V-DQN constructs a solution by sequentially adding nodes to the partial solution based on maximizing the evaluation function that measures the quality of nodes in the current partial solution, and the termination criterion checks whether all edges have been covered [138]. (Redrawn from [138]).

key node finding [134], [135] and network dismantling [136] enable the application to the epidemic control problem.

4) Combinatorial Optimization: Most real-world combinatorial optimization problems could be represented by graphs. GRL methods for combinatorial optimization allow learning automatic strategies that return feasible and high-quality outputs based on the original input data and the guidance process [137]. S2V-DQN [138] learn a policy for building solutions incrementally with Q-learning. in which the agent incrementally constructs efficient solutions based on the graph structure through adding nodes, and this greedy algorithm is a popular pattern for approximating algorithms and heuristics for combinatorial optimization, and the S2V-DQN architecture is shown in Fig. 8. Toyer et al. [139] propose an Action Schema Network for learning generalized policies for probabilistic planning problems. The AS-Net can employ a weight sharing scheme through simulating the relational structure of the planning problem, allowing the network to be applied to any problem in a given planning domain.

5) Medicine: Graph mining methods and reinforcement learning techniques have demonstrated significant potential in medicine combination prediction (MCP) [8], chemical reaction product prediction [9], and brain network analysis tasks [140]. To this end, CompNet [8] proposes a graph convolutional reinforcement learning model for MCP. Specifically, the authors define the MCP task as an order-free MDP problem, and they first employ a Dual Convolutional Neural Network to obtain patient representations, and then introduce medical knowledge associated with the predicted medicines to construct medicine KG, and R-GCN is used for KG encoding. Finally, the patient representation and medical KG are combined for medicine selection, where DQN is used for learning correlations and adverse interactions between medicines. Graph Transformation Policy Network (GTPN) [9] addresses the problem of chemical reaction product prediction. They represented the entity system consisting of input reactants and reagent molecules as a labeled graph with multiple connected components by using a GNN, and the process of generating product molecules from reactant molecules is formulated as a series of graph transformation processes. GTPN uses A2C to find the optimal sequence of bond changes that transforms the reactants into products. Moreover, BN-GNN [140] is a brain network representation framework, which enables determining the optimal number of featuret aggregations with DDQN, and the combination of abundant medical KG and DRL methods could support patients to describe disease characteristics and provide accuracy in diseases diagnosis [141].

In the pharmaceutical industry, design and discovery aids for de novo drugs serve an important research value, and many scholars have conducted studies on chemical molecule generation and optimization with RL methods [142], [143]. These methods that use RL [97], [122], [144] to pre-design the molecular structure of drugs can significantly save working time, money, and labor costs in chemical laboratories and provide an efficient aid for the design and discovery of new drugs.

IV. Open Research Directions

Although several GRL methods have been explored and proposed by scholars at present, we believe that this field is still in the initial stage and has much exploration space and research value, and we suggest some future research directions that might be of interest to scholars in this field.

Automatic design for model architecture: Modeling of the environment, selection of RL algorithms, and hyperparameter design of the model in GRL methods commonly require detailed expert exploration. The different definitions of state and action spaces, batch sizes and frequency of batch updates, and the number of timesteps will result in different experimental performances. Automatic RL provides a solution for automatically making appropriate decisions about the settings of the RL process before starting to learn, enabling a non-expert way to perform, and improving the range of applications of RL. We believe that applying the automatic methods for model architecture design to graph mining will significantly improve the efficiency of the methods and provide more efficient learning strategies for exploring optimal combinations of graph mining algorithms and RL methods. In our survey, there is a limited number of scholars applying automatic RL to graph mining tasks, but it is an extremely important and promising research direction.

Multi-agent interaction in graph-structured environment. Single agent usually ignore the interests of other agents in the process of interacting with the environment. Individual agents

often consider other agents in the environment as part of the environment and will fail to adapt to the instability environment during the learning process as the interaction of other agents with the environment. Multi-agent RL considers the communication between multiple agents, allowing them to collaborate learning effectively. Existing multi-agent RL methods have proposed several policies such as bidirectional channels [148], sequential transfer [149] and all-to-all channels [150] to achieve interaction between agents. We believe that the learned features from multi-agent and the relational information between these agents can be fused to improve the performance of graph mining methods in graph mining tasks. Furthermore, collaborative strategies in multi-agent can contribute to GRL models to solve graph mining problems with parallel or partitioned strategies.

Subgraph Patterns Mining: In the process of network local structure analysis, many scholars have explored and discovered many novel graph mining methods that exploit the local structures of the graphs with different strategies [15], [27], [151], [152]. Furthermore, they have employed batch sampling and importance sampling to obtain the local structure for network representation learning. Graph local structure selection provides natural advantages for GRL. The process of sequential construction for graph local structure allows being represented as an MDP and guided by the performance of downstream classification and prediction tasks. These adaptive neighbor selection methods enable the performance of GNNs to be improved and the importance of different relationships in messages passing to be analyzed in heterogeneous information networks.

Explainability of GRL methods: The explainability of deep learning models has revolutionary potential for understanding the inherent meaning and potential mechanisms of the models, however, these models are still treated as black boxes, which lack human-understandable explanations. The untrusted GRL models cannot be employed in specific domains like critical applications involving fairness, privacy, and security. Most explainability methods in GRL are example-level methods that explain the model by determining the important features in the input or the decision process for the input, where the Hierarchical RL (HRL) proposes that the top-level policies focus on higher-level objectives, while the sub-policies are responsible for fine-grained control. The example-level methods designed following the principles of HRL allow the design of explanation methods with a hierarchical structure to enhance the explainability and robustness of the models. Furthermore, influential subgraphs in graph-structured data have also been adopted to explain graph mining methods. Different from the example-level methods, the model-level methods study what graph patterns can maximize certain predictions. We believe that both model-level and example-level methods are important for explaining and understanding deep models and a limited number of GRL models contain explanation analysis for models, which is an important direction for future research in this field.

Uniform GRL evaluation framework: It is crucial to define the reward function in RL. A reasonable reward function could enable RL algorithms to converge faster and better to obtain the expected goal. However, existing GRL methods commonly employ the performance of downstream classifiers or predictors for designing reward functions. These methods allow GRL methods to be evaluated based on the performance of downstream tasks, but the execution of GRL methods includes multiple steps, and evaluating the results of each action using downstream tasks would cause substantial resource consumption. Although many scholars have proposed novel methods to reduce this consumption or to design more efficient reward functions to guide the training process of RL, the problem of resource consumption still remains, and we believe that designing reward functions that rely on actions or providing a unified GRL evaluation framework can effectively improve the performance of the algorithm. This is an urgent problem to be solved. Furthermore, GRL methods for the same graph mining problem lack uniform evaluation metrics, and the design of appropriate and effective evaluation metrics has significant importance for evaluating specific tasks.

V. Conclusion

Network science has developed into an interdisciplinary field spanning physics, economics, biology, and computer science, with many critical real-world applications, and has allowed the modeling and analysis of the interaction of entities in complex systems. This paper provides a comprehensive overview of the state-of-the-art methods in GRL. We summarize various solutions for different graph mining tasks and highlight the key advantages and challenges of fusing graph mining methods with RL methods. Although several GRL methods have been employed across many domains, the combination of the more efficient RL with the powerful GNN promises to provide better generalization and explainability as well as improve the ability to tackle sample complexity. We review existing methods in terms of several graph mining tasks. We believe that GRL methods can reveal valuable information in the growing graph-structured data, and allow scholars and engineers to analyze and exploit graph-structured data with greater clarity. Furthermore, we hope that this survey can help scholars understand the key concepts of GRL and promote the development of the field in the future.

Appendix A
Notations

Here we provide the commonly used notations and explanations in different domains in Table IX.

Appendix B
Open-Source Implementations

Here we summarize the open-source implementations of GRL in this survey in Table X.

TABLE IX
COMMONLY USED NOTATIONS

Notation	Explanation	Domain
$\lvert\cdot\rvert$	The length of a set.	GNN
G	A graph.	GNN
$A \in \{0,1\}^{m\times m}$	The graph adjacency matrix.	GNN
$X_i \in \mathbb{R}^{m\times d_i}$	The feature matrix of a graph in layer i.	GNN
E	The set of edges in a graph, $e \in E$.	GNN
V	The set of nodes in a graph, $v \in V$.	GNN
D	The degree matrix of A, $D_{ii} = \sum_{j=1}^{n} A_{ij}$.	GNN
n	The number of nodes, $n = \lvert V\rvert$.	GNN
m	The number of edges, $m = \lvert E\rvert$.	GNN
d	The dimension of a node feature vector.	GNN
$W_i \in \mathbb{R}^{d_i\times d_{i+1}}$	Learnable model parameters.	GNN
$\sigma(\cdot)$	The sigmoid activation function.	GNN
$\mathcal{F}: v \to e_v \in \mathbb{R}^d$	Mapping functions in graph embedding.	GNN
(h, r, t)	h denotes the head entity, t denotes the tail entity, r denotes the relationship between h and t.	KG
$s_t \in \mathcal{S}$	The set of all possible states.	RL
$a_t \in \mathcal{A}$	The set of actions that are available in the state.	RL
$r_t \in \mathcal{R}$	The reward is given to the agent by the environment after the agent performs an action.	RL
$\mathcal{T}$	The state transfer function is defined by the environment.	RL
γ	Discount Factor.	RL
$\pi: \mathcal{S} \to p(\mathcal{A})$	Policy for agent to perform in the environment.	RL
$a_{i,j}$	Non-negative learning factor.	REINFORCE
$b_{i,j}$	Representation function of the state for reducing the variance of the gradient estimate.	REINFORCE
g_i	The probability density function for randomly generating unit activation-based actions.	REINFORCE
$L(\theta_i)$	Loss function.	DQN

TABLE X
A SUMMARY OF OPEN-SOURCE IMPLEMENTATIONS

Model	Year	Framework	Link	Source
AGILE	2022	PyTorch	https://github.com/clvrai/agile	[153]
GTA-RL	2022	PyTorch	https://github.com/udeshmg/GTA-RL	[154]
SubgraphX	2021	PyTorch	https://github.com/divelab/DIG	[81]
SUGAR	2021	Tensorflow	https://github.com/RingBDStack/SUGAR	[15]
CORL	2021	PyTorch	https://github.com/huawei-noah/trustworthyAI/tree/master/gcastle	[105]
RioGNN	2021	PyTorch	https://github.com/safe-graph/RioGNN	[62]
IG-RL	2021	PyTorch	https://github.com/FXDevailly/IG-RL	[124]
SparRL	2021	PyTorch	https://github.com/rwickman/SparRL-PyTorch	[22]
PAAR	2021	PyTorch	https://github.com/seukgcode/PAAR	[67]
Policy-GNN	2020	PyTorch	https://github.com/lhenry15/Policy-GNN	[27]
CARE-GNN	2020	PyTorch	https://github.com/YingtongDou/CARE-GNN	[63]
KG-A2C	2020	PyTorch	https://github.com/rajammanabrolu/KG-A2C	[155]
GPA	2020	PyTorch	https://github.com/ShengdingHu/GraphPolicyNetworkActiveLearning	[14]
GAEA	2020	Tensorflow	https://github.com/salesforce/GAEA	[156]
CompNet	2019	PyTorch	https://github.com/WOW5678/CompNet	[8]
DRL+GNN	2019	PyTorch	https://github.com/knowledgedefinednetworking/DRL-GNN	[130]
GPN	2019	PyTorch	https://github.com/qiang-ma/graph-pointer-network	[157]
PGPR	2019	PyTorch	https://github.com/orcax/PGPR	[158]
DGN	2018	PyTorch	https://github.com/PKU-AI-Edge/DGN	[159]
KG-DQN	2018	PyTorch	https://github.com/rajammanabrolu/KG-DQN	[160]
ASNets	2018	Tensorflow	https://github.com/qxcv/asnets	[139]
DeepPath	2017	Tensorflow	https://github.com/xwhan/DeepPath	[3]
MINERVA	2017	Tensorflow	https://github.com/shehzaadzd/MINERVA	[71]
KBGAN	2017	PyTorch	https://github.com/cai-lw/KBGAN	[72]

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their careful reading and useful comments that helped us to improve the final version of this paper.

REFERENCES

[1] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *Int. J. Rob. Res.*, vol. 37, no. 4–5, pp. 421–436, 2018.

[2] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[3] W. Xiong, T. Hoang, and W. Y. Wang, "Deeppath: A reinforcement learning method for knowledge graph reasoning," in *Proc. Conf. Empir. Methods Natural Lang. Process.*, 2017, pp. 564–573.

[4] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1329–1338.

[5] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai, "Deep direct reinforcement learning for financial signal representation and trading," *IEEE Trans. Neural. Netw. Learn. Syst.*, vol. 28, no. 3, pp. 653–664, Mar. 2017.

[6] J. B. Heaton, N. G. Polson, and J. H. Witte, "Deep learning for finance: Deep portfolios," *Appl. Stoch. Models. Bus. Ind.*, vol. 33, no. 1, pp. 3–12, 2017.

[7] F. Soleymani and E. Paquet, "Deep graph convolutional reinforcement learning for financial portfolio management-deeppocket," *Expert Syst. Appl.*, vol. 182, 2021, Art. no. 115127.
[8] S. Wang, P. Ren, Z. Chen, Z. Ren, J. Ma, and M. d. Rijke, "Order-free medicine combination prediction with graph convolutional reinforcement learning," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Beijing, China, 2019, pp. 1623–1632.
[9] K. Do, T. Tran, and S. Venkatesh, "Graph transformation policy network for chemical reaction prediction," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, Anchorage, AK, USA, 2019, pp. 750–760.
[10] W. Qiu, H. Chen, and B. An, "Dynamic electronic toll collection via multi-agent deep reinforcement learning with edge-based graph convolutional networks," in *Proc. Int. Joint Conf.* Artif. Intell., Macao, China, 2019, pp. 4568–4574.
[11] M. Obara, T. Kashiyama, and Y. Sekimoto, "Deep reinforcement learning approach for train rescheduling utilizing graph theory," in *Proc. IEEE Int. Conf. Big Data*, New York, NY, USA, 2018, pp. 4525–4533.
[12] M. Zhang and Y. Chen, "Link prediction based on graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst*, Montréal, CA, 2018, pp. 5171–5181.
[13] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *Proc. Int. Conf. Mach. Learn.*, New York, NY, USA, 2016, vol. 48, pp. 2071–2080.
[14] S. Hu et al., "Graph policy network for transferable active learning on graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, BC, CA, 2020, vol. 33, pp. 10174–10185.
[15] Q. Sun et al., "Sugar: Subgraph neural network with reinforcement pooling and self-supervised mutual information mechanism," in *Proc. Web Conf., Ljubljana, SI*, 2021, pp. 2081–2091.
[16] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2017, pp. 1025–1035.
[17] H. Yuan and S. Ji, "Structpool: Structured graph pooling via conditional random fields," in *Proc. Int. Conf. Learn. Representations*, Addis Ababa, Ethiopia, 2020.
[18] H. Yuan, J. Tang, X. Hu, and S. Ji, "XGNN: Towards model-level explanations of graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, Virtual Event, CA, USA, 2020, pp. 430–438.
[19] H. Dai et al., "Adversarial attack on graph structured data," in *Proc. Int. Conf. Mach. Learn.*, Stockholm, SE, 2018, vol. 80, pp. 1115–1124.
[20] P. T. A. Quang, Y. Hadjadj-Aoul, and A. Outtagarts, "A deep reinforcement learning approach for VNF forwarding graph embedding," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 4, pp. 1318–1331, Dec. 2019.
[21] Z. Yan, J. Ge, Y. Wu, L. Li, and T. Li, "Automatic virtual network embedding: A deep reinforcement learning approach with graph convolutional networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1040–1057, Jun. 2020.
[22] R. Wickman, "SPARRL: Graph sparsification via deep reinforcement learning," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Philadelphia, PA, USA, 2022, pp. 2521–2523.
[23] K. Rusek, J. Suárez-Varela, A. Mestres, P. Barlet-Ros, and A. Cabellos-Aparicio, "Unveiling the potential of graph neural networks for network modeling and optimization in SDN," in *Proc. ACM SIGCOMM Symp. SDN Res.*, San Jose, CA, USA, 2019, pp. 140–151.
[24] J. Suárez-Varela et al., "Challenging the generalization capabilities of graph neural networks for network modeling," in *Proc. ACM SIGCOMM Conf. Posters Demos*, Beijing, China, 2019, pp. 114–115.
[25] R. Wang, F. Fang, J. Cui, and W. Zheng, "Learning self-driven collective dynamics with graph networks," *Sci. Rep.*, vol. 12, no. 1, pp. 1–11, 2022.
[26] N. Shervashidze, P. Schweitzer, E. J. v. Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler-lehman graph kernels," *J. Mach. Learn. Res.*, vol. 12, pp. 2539–2561, 2011.
[27] K.-H. Lai, D. Zha, K. Zhou, and X. Hu, "Policy-GNN: Aggregation optimization for graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, Virtual Event, CA, USA 2020, pp. 461–471.
[28] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "DBpedia: A nucleus for a web of open data," in *The Semantic Web*, Berlin, Germany: Springer, 2007, pp. 722–735.
[29] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, New York, NY, USA, 2008, pp. 1247–1250.
[30] F. Mahdisoltani, J. Biega, and F. Suchanek, "YAGO3: A knowledge base from multilingual wikipedias," in *Proc. Bienn. Conf. Innov. Data Syst. Res.*, Asilomar, CA, USA, 2015.
[31] X. Zhao, L. Chen, and H. Chen, "A weighted heterogeneous graph-based dialog system," *IEEE Trans. Neural. Netw. Learn. Syst.*, early access, Nov. 18, 2021, doi: 10.1109/TNNLS.2021.3124640.
[32] Q. Wang, Y. Ji, Y. Hao, and J. Cao, "GRL: Knowledge graph completion with GAN-based reinforcement learning," *Knowl. Based Syst.*, vol. 209, 2020, Art. no. 106421.
[33] M. Gardner, P. Talukdar, B. Kisiel, and T. Mitchell, "Improving learning and inference in a large knowledge-base using latent syntactic cues," in *Proc. Conf. Empir. Methods Natural Lang. Process.*, Seattle, Washington, USA, 2013, pp. 833–838.
[34] A. Bordes, N. Usunier, A. Garcia-Durán, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. Adv. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, 2013, vol. 2, pp. 2787–2795.
[35] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. AAAI Conf. Artif. Intell.*, Québec City, QC, CA, 2014, pp. 1112–1119.
[36] T. Wang, R. Liao, J. Ba, and S. Fidler, "Nervenet: Learning structured policy with graph neural networks," in *Proc. Int. Conf. Learn. Representations*, 2018.
[37] J. Schrittwieser et al., "Mastering atari, go, chess and shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, 2020.
[38] D. Silver et al., "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.
[39] T. M. Moerland, J. Broekens, A. Plaat, and C. M. Jonker, "Model-based reinforcement learning: A survey," *Found. Trends Mach. Learn.*, vol. 16, pp. 1–118, 2023.
[40] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.
[41] C. J. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, 1992.
[42] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3, pp. 229–256, 1992.
[43] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, Denver, CO, 1999, pp. 1008–1014.
[44] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, New York, NY, USA, 2016, vol. 48, pp. 1928–1937.
[45] J. X. Wang et al., "Learning to reinforcement learn," 2016, *arXiv:1611.05763*.
[46] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learn.*,Bejing, China 2014, vol. 32, pp. 387–395.
[47] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, San Juan, Puerto Rico, 2016.
[48] H. v. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proc. AAAI Conf. Artif. Intell.*, Phoenix, AZ, USA, 2016, pp. 2094–2100.
[49] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable MDPS," in *Proc. AAAI Conf. Artif. Intell.*, Austin, TX, USA, 2015.
[50] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, New York, NY, USA 2016, vol. 48, pp. 1995–2003.
[51] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–106, 2008.
[52] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2D knowledge graph embeddings," in *Proc. AAAI Conf. Artif. Intell.*, New Orleans, LA, USA, 2018, pp. 1811–1818.
[53] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon, "Representing text for joint embedding of text and knowledge bases," in *Proc. Conf. Empir. Methods Nat. Lang. Process*, Lisbon, PT, 2015, pp. 1499–1509.
[54] Y. Gao, H. Yang, P. Zhang, C. Zhou, and Y. Hu, "Graph neural architecture search," in *Proc. IJCAI Int. Joint Conf. Artif. Intell.*, vol. 20, Yokohama, Yokohama, Japan, 2021, pp. 1403–1409.
[55] K. Zhou, Q. Song, X. Huang, and X. Hu, "Auto-GNN: Neural architecture search of graph neural networks," Front. Big Data, vol. 5, 2022, Art. no. 1029307.

[56] L. Wang et al., "Learning robust representations with graph denoising policy network," in *Proc. IEEE Int. Conf. Data Min.*, Beijing, China, 2019, pp. 1378–1383.

[57] Y. Sun, S. Wang, X. Tang, T.-Y. Hsieh, and V. Honavar, "Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach," in *Proc. Web Conf.*,Taipei, China, 2020, pp. 673–683.

[58] S. Shen et al., "Learning network representation through reinforcement learning," in *IEEE Int. Conf. Acoust. Speech Signal Process Proc.*, 2020, pp. 3537–3541.

[59] J. Leskovec and J. Mcauley, "Learning to discover social circles in ego networks," in *Proc. Adv. Neural Inf. Process. Syst.*,Lake Tahoe, NV, USA, 2012, pp. 539–547.

[60] C. Wang, Y. Liu, X. Gao, and G. Chen, "A reinforcement learning model for influence maximization in social networks," in *Lect. Notes Comput. Sci.*, Taipei, China, 2021, pp. 701–709.

[61] J. J. McAuley and J. Leskovec, "From amateurs to connoisseurs: Modeling the evolution of user expertise through online reviews," in *Proc. Int. Conf. World Wide Web*, Rio de Janeiro, BR, 2013, pp. 897–908.

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

[63] 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 *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, Virtual Event, Ireland, 2020, pp. 315–324.

[64] H. Liu, S. Zhou, C. Chen, T. Gao, J. Xu, and M. Shu, "Dynamic knowledge graph reasoning based on deep reinforcement learning," *Knowl. Based Syst.*, vol. 241, 2022, Art. no. 108235.

[65] H. Wang, S. Li, R. Pan, and M. Mao, "Incorporating graph attention mechanism into knowledge graph reasoning based on deep reinforcement learning," in *Proc. Conf. Empir. Methods Natural Lang. Process. Int. Jt Conf. Natural Lang. Process.*, Hong Kong, China, 2019, pp. 2623–2631.

[66] G. Wan, S. Pan, C. Gong, C. Zhou, and G. Haffari, "Reasoning like human: Hierarchical reinforcement learning for knowledge graph reasoning," in *Proc. Int. Joint Conf. Artif. Intell.*, Yokohama, JP, 2021, pp. 1926–1932.

[67] X. Zhou, P. Wang, Q. Luo, and Z. Pan, "Multi-hop knowledge graph reasoning based on hyperbolic knowledge graph embedding and reinforcement learning," in *Proc. ACM Int. Conf. Proc. Ser.*, Virtual Event, TH, 2021, pp. 1–9.

[68] M. Zheng, Y. Zhou, and Q. Cui, "Hierarchical policy network with multi-agent for knowledge graph reasoning based on reinforcement learning," in *Proc. Int. Conf. Knowl. Sci. Eng. Manag.*, Cham, CH, 2021, pp. 445–457.

[69] Q. Wang, Y. Hao, and J. Cao, "ADRL: An attention-based deep reinforcement learning framework for knowledge graph reasoning," *Knowl. Based Syst.*, vol. 197, 2020, Art. no. 105910.

[70] X. V. Lin, R. Socher, and C. Xiong, "Multi-hop knowledge graph reasoning with reward shaping," in *Proc. Conf. Empir. Methods Natural Lang. Process.*, Brussels, BE, 2018, pp. 3243–3253.

[71] R. Das et al., "Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, Vancouver, BC, Canada, 2018.

[72] L. Cai and W. Y. Wang, "KBGAN: Adversarial learning for knowledge graph embeddings," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, New Orleans, LA, 2018, pp. 1470–1480.

[73] L. Chen, J. Cui, X. Tang, Y. Qian, Y. Li, and Y. Zhang, "RLPATH: A knowledge graph link prediction method using reinforcement learning based attentive relation path searching and representation learning," *Appl. Intell.*, vol. 52, no. 4, pp. 4715–4726, 2022.

[74] P. Tiwari, H. Zhu, and H. M. Pandey, "Dapath: Distance-aware knowledge graph reasoning based on deep reinforcement learning," *Neural Netw.*, vol. 135, pp. 1–12, 2021.

[75] S. Li, H. Wang, R. Pan, and M. Mao, "Memorypath: A deep reinforcement learning framework for incorporating memory component into knowledge graph reasoning," *Neurocomputing*, vol. 419, pp. 273–286, 2021.

[76] Z. Li, X. Jin, S. Guan, Y. Wang, and X. Cheng, "Path reasoning over knowledge graph: A multi-agent and reinforcement learning based method," in *Proc. IEEE Int. Conf. Data Mining Workshops*,Los Alamitos, CA, USA, 2018, pp. 929–936.

[77] A. Mukherjee, V. Venkataraman, B. Liu, and N. Glance, "What yelp fake review filter might be doing?," in *Proc. AAAI Conf. Artif. Intell.*, Cambridge, MA, USA, 2013, vol. 7, pp. 409–418.

[78] Y. Deng, Y. Li, F. Sun, B. Ding, and W. Lam, "Unified conversational recommendation policy learning via graph-based reinforcement learning," in *Proc. ACM SIGIR Int. Conf. Res. Develop. Inf. Retrieval*, Virtual Event, CA, 2021, pp. 1431–1441.

[79] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowl. Inf. Syst.*, vol. 42, no. 1, pp. 181–213, 2015.

[80] A. K. Debnath, R. L. L. d. Compadre, G. Debnath, A. J. Shusterman, and C. Hansch, "Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity," *J. Med. Chem.*, vol. 34, no. 2, pp. 786–797, 1991.

[81] H. Yuan, H. Yu, J. Wang, K. Li, and S. Ji, "On explainability of graph neural networks via subgraph explorations," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 12241–12252.

[82] Y. Luo et al., "Automated data augmentations for graph classification," in *Proc. Int. Conf. Learn. Representations*, Kigali, Rwanda, 2023.

[83] H. Toivonen, A. Srinivasan, R. D. King, S. Kramer, and C. Helma, "Statistical evaluation of the predictive toxicology challenge 2000-2001," *Bioinformatics*, vol. 19, no. 10, pp. 1183–1193, 2003.

[84] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. Vishwanathan, A. J. Smola, and H.-P. Kriegel, "Protein function prediction via graph kernels," *Bioinformatics*, vol. 21, pp. i47–i56, 2005.

[85] Z. Wu et al., "Moleculenet: A benchmark for molecular machine learning," *Chem. Sci.*, vol. 9, no. 2, pp. 513–530, 2018.

[86] N. Wale, I. A. Watson, and G. Karypis, "Comparison of descriptor spaces for chemical compound retrieval and classification," *Knowl. Inf. Syst.*, vol. 14, no. 3, pp. 347–375, 2008.

[87] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, Toulon, France, 2017.

[88] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, Vancouver, BC, Canada, 2018.

[89] H. Gao, Z. Wang, and S. Ji, "Large-scale learnable graph convolutional networks," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, London, U.K., 2018, pp. 1416–1424.

[90] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *Proc. Int. Conf. Learn. Representations*, New Orleans, LA, USA, 2019.

[91] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proc. AAAI Conf. Artif. Intell.*, Austin, TX, USA, 2015, pp. 2181–2187.

[92] W.-t. Yih, K. Toutanova, J. C. Platt, and C. Meek, "Learning discriminative projections for text similarity measures," in *Proc. Conf. Comput. Natural Lang. Learn.*, Portland, OR, USA, 2011, pp. 247–256.

[93] S. Hong, D. Yoon, and K.-E. Kim, "Structure-aware transformer policy for inhomogeneousmulti-task reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2022.

[94] J. Chen, T. Ma, and C. Xiao, "FASTGCN: Fast learning with graph convolutional networks via importance sampling," in *Proc. Int. Conf. Learn. Representations*, Vancouver, BC, Canada, 2018.

[95] Z. Li, Y. Sun, S. Tang, C. Zhang, and H. Ma, "Reinforcement learning with dual attention guided graph convolution for relation extraction," in *Proc. Int. Conf. Pattern Recognit.*, Milan, IT, 2021, pp. 946–953.

[96] X. Fu et al., "ACE-HGNN: Adaptive curvature exploration hyperbolic graph neural network," in *Proc. IEEE Int. Conf. Data Min.*, 2021, pp. 111–120.

[97] J. B. Lee, R. Rossi, and X. Kong, "Graph classification using structural attention," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, London, U.K., 2018, pp. 1666–1674.

[98] J. Xu et al., "NetRL: Task-aware network denoising via deep reinforcement learning," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 810–823, Jan. 2023.

[99] X. Wang et al., "Heterogeneous graph attention network," in *Proc. Int. Conf. World Wide Web*, San Francisco, CA, USA, 2019, pp. 2022–2032.

[100] C. Yang, Y. Feng, P. Li, Y. Shi, and J. Han, "Meta-graph based HIN spectral embedding: Methods, analyses, and insights," in *Proc. IEEE Int. Conf. Data Min.*, Singapore, 2018, pp. 657–666.

[101] Z. Zhong, C.-T. Li, and J. Pang, "Personalised meta-path generation for heterogeneous graph neural networks," *Data. Min. Knowl. Discov.*, vol. 36, no. 6, pp. 2299–2333, 2022.

[102] Y. Ma, S. Wang, T. Derr, L. Wu, and J. Tang, "Graph adversarial attack via rewiring," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, Virtual Event, Singapore, 2021, pp. 1161–1169.

[103] Y. Lyu, X. Yang, J. Liu, S. Xie, P. S. Yu, and X. Zhang, "Interpretable and effective reinforcement learning for attacking against graph-based rumor detection," in *Proc. Int. Jt. Conf. Neural Networks*, Padua, Italy, 2023.
[104] S. Zhu, I. Ng, and Z. Chen, "Causal discovery with reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, Addis Ababa, Ethiopia, 2020.
[105] X. Wang et al., "Ordering-based causal discovery with reinforcement learning," in *Proc. IJCAI Int. Joint Conf. Artif. Intell.*, Vienna, Austria, 2021, pp. 3566–3573.
[106] Y. Sun, K. Zhang, and C. Sun, "Model-based transfer reinforcement learning based on graphical model representations," *IEEE Trans. Neural. Netw. Learn. Syst.*, vol. 34, no. 2, pp. 1035–1048, Feb. 2023.
[107] L. Chen, B. Tan, S. Long, and K. Yu, "Structured dialogue policy with graph neural networks," in *Proc. Int. Conf. Comput. Linguist.*, Santa Fe, NM, USA, 2018, pp. 1257–1268.
[108] Y. Chen, L. Wu, and M. J. Zaki, "Reinforcement learning based graph-to-sequence model for natural question generation," in *Proc. Int. Conf. Learn. Representations*, Addis Ababa, Ethiopia, 2020.
[109] S. Zhou et al., "Interactive recommender system via knowledge graph-enhanced reinforcement learning," in *Proc. ACM SIGIR Int. Conf. Res. Dev. Inf. Retr.*, Virtual Event, China, 2020, pp. 179–188.
[110] A. Zhu, D. Ouyang, S. Liang, and J. Shao, "Step by step: A hierarchical framework for multi-hop knowledge graph reasoning with reinforcement learning," *Knowl. Based Syst.*, vol. 248, 2022, Art. no. 108843.
[111] H. Sun, J. Zhong, Y. Ma, Z. Han, and K. He, "Timetraveler: Reinforcement learning for temporal knowledge graph forecasting," in *Proc. Conf. Empir. Methods Nat. Lang. Process.*, Punta Cana, Dominican Republic, 2021, pp. 8306–8319.
[112] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, Denver, CO, 1999, pp. 1057–1063.
[113] X. Zhao, L. Zhang, Z. Ding, L. Xia, J. Tang, and D. Yin, "Recommendations with negative feedback via pairwise deep reinforcement learning," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, London, U.K., 2018, pp. 1040–1048.
[114] P. Wang, K. Liu, L. Jiang, X. Li, and Y. Fu, "Incremental mobile user profiling: Reinforcement learning with spatial knowledge graph for modeling event streams," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, Virtual Event, CA, USA, 2020, pp. 853–861.
[115] H. Yuan, H. Yu, S. Gui, and S. Ji, "Explainability in graph neural networks: A taxonomic survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 5, pp. 5782–5799, 2023.
[116] M. Yousefi, N. Mtetwa, Y. Zhang, and H. Tianfield, "A reinforcement learning approach for attack graph analysis," in *Proc. IEEE Int. Conf. Trust. Secur. Priv. Comput. Commun. Int. Conf. Big Data Sci. Eng.*, New York, NY, USA, 2018, pp. 212–217.
[117] R. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, "Gnnexplainer: Generating explanations for graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst*., Vancouver, Canada, vol. 32, 2019.
[118] D. Luo et al., "Parameterized explainer for graph neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, BC, CA, 2020, vol. 33, pp. 19620–19631.
[119] M. Vu and M. T. Thai, "PGM-explainer: Probabilistic graphical model explanations for graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, vol. 33, pp. 12225–12235.
[120] J. J. Damanik, H. Kasan, and H.-L. Choi, "Solving delivery assignment in hybrid-transit network using multi-agent reinforcement learning," in *Proc. Int. Conf. Robot Intell. Technol. Appl.*, Brisbane, AU, 2022, pp. 485–497.
[121] H. Peng et al., "Dynamic graph convolutional network for long-term traffic flow prediction with reinforcement learning," *Inf. Sci.*, vol. 578, pp. 401–416, 2021.
[122] J. You, B. Liu, R. Ying, V. Pande, and J. Leskovec, "Graph convolutional policy network for goal-directed molecular graph generation," in *Proc. Adv. Neural Inf. Process. Syst.*, Montréal, CA, 2018, pp. 6412–6422.
[123] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[124] F.-X. Devailly, D. Larocque, and L. Charlin, "IG-RL: Inductive graph reinforcement learning for massive-scale traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 7496–7507, Jul. 2022.
[125] Z. Yu and M. Hu, "Deep reinforcement learning with graph representation for vehicle repositioning," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 13094–13107, Aug. 2022.
[126] S. Chen, J. Dong, P. Ha, Y. Li, and S. Labi, "Graph neural network and reinforcement learning for multi-agent cooperative control of connected autonomous vehicles," *Comput-Aided Civ. Inf.*, vol. 36, no. 7, pp. 838–857, 2021.
[127] H. Zuo, H. Si, N. Ding, X. Wang, and G. Tan, "Coordinated learning for lane changing based on coordination graph and reinforcement learning," in *Proc. Artif. Intell.*, China, Singapore, 2020, pp. 599–607.
[128] Z. Zeng, "Graphlight: Graph-based reinforcement learning for traffic signal control," in *Proc. Int. Conf. Comput. Commun. Syst.*, Chengdu, China, 2021, pp. 645–650.
[129] Y. Wang, T. Xu, X. Niu, C. Tan, E. Chen, and H. Xiong, "STMARL: A spatio-temporal multi-agent reinforcement learning approach for cooperative traffic light control," *IEEE Trans. Mob. Comput.*, vol. 21, no. 6, pp. 2228–2242, Jun. 2022.
[130] P. Almasan, J. Suárez-Varela, K. Rusek, P. Barlet-Ros, and A. Cabellos-Aparicio, "Deep reinforcement learning meets graph neural networks: Exploring a routing optimization use case," *Comput. Commun.*, vol. 196, pp. 184–194,2022.
[131] K. Nakashima, S. Kamiya, K. Ohtsu, K. Yamamoto, T. Nishio, and M. Morikura, "Deep reinforcement learning-based channel allocation for wireless LANs with graph convolutional networks," *IEEE Access*, vol. 8, pp. 31823–31834, 2020.
[132] E. Meirom, H. Maron, S. Mannor, and G. Chechik, "Controlling graph dynamics with reinforcement learning and graph neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2021, vol. 139, pp. 7565–7577.
[133] B. Wang, Y. Sun, T. Q. Duong, L. D. Nguyen, and L. Hanzo, "Risk-aware identification of highly suspected COVID-19 cases in social IoT: A joint graph theory and reinforcement learning approach," *IEEE Access*, vol. 8, pp. 115655–115661, 2020.
[134] C. Fan, L. Zeng, Y. Sun, and Y.-Y. Liu, "Finding key players in complex networks through deep reinforcement learning," *Nature Mach. Intell.*, vol. 2, no. 6, pp. 317–324, 2020.
[135] L. Ma et al., "Influence maximization in complex networks by using evolutionary deep reinforcement learning," *IEEE Trans. Emerg. Top. Comput. Intell.*, early access, Jan. 13, 2022, doi: 10.1109/TETCI.2021.3136643.
[136] D. Yan, W. Xie, Y. Zhang, Q. He, and Y. Yang, "Hypernetwork dismantling via deep reinforcement learning," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 5, pp. 3302–3315, 2022.
[137] C. K. Joshi, Q. Cappart, L.-M. Rousseau, and T. Laurent, "Learning TSP requires rethinking generalization," in *Proc. Int. Proc. Inform.*, Virtual, FR, 2021, vol. 210, pp. 33:1–33:21.
[138] H. Dai, E. B. Khalil, Y. Zhang, B. Dilkina, and L. Song, "Learning combinatorial optimization algorithms over graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2018, pp. 6351–6361.
[139] S. Toyer, F. Trevizan, S. Thiebaux, and L. Xie, "Action schema networks: Generalised policies with deep learning," in *Proc. AAAI Conf. Artif. Intell.*, New Orleans, LA, USA, 2018, pp. 6294–6301.
[140] X. Zhao et al., "Deep reinforcement learning guided graph neural networks for brain network analysis," *Neural Netw.*, pp. 56–67, vol. 154, 2022.
[141] Y. Jia, Z. Tan, and J. Zhang, "DKDR: An approach of knowledge graph and deep reinforcement learning for disease diagnosis," in *Proc. IEEE Intl. Conf. Parallel Distrib. Process. Appl. Big Data Cloud Comput. Sustain. Comput. Commun. Social Comput. Netw.*, Los Alamitos, CA, USA, 2019, pp. 1303–1308.
[142] Y. Khemchandani et al., "Deepgraphmolgen, a multi-objective, computational strategy for generating molecules with desirable properties: A graph convolution and reinforcement learning approach," *J. Cheminformatics*, vol. 12, no. 1, pp. 1–17, 2020.
[143] W. Jin, R. Barzilay, and T. Jaakkola, "Multi-objective molecule generation using interpretable substructures," in *Proc. Int. Conf. Mach. Learn.*, vol. 119, 2020, pp. 4849–4859.
[144] N. D. Cao and T. Kipf, "MOLGAN: An implicit generative model for small molecular graphs," 2018, *arXiv:1805.11973*.
[145] P. Sun, J. Lan, J. Li, Z. Guo, and Y. Hu, "Combining deep reinforcement learning with graph neural networks for optimal VNF placement," *IEEE Commun. Lett.*, vol. 25, no. 1, pp. 176–180, Jan. 2021.
[146] H. Wang et al., "GCN-RL circuit designer: Transferable transistor sizing with graph neural networks and reinforcement learning," in *Proc. ACM/IEEE Des. Automat. Conf.*, Virtual Event USA, 2020, pp. 1–6.

[147] H. Dai, Y. Li, C. Wang, R. Singh, P.-S. Huang, and P. Kohli, "Learning transferable graph exploration," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, Canada, vol. 32, 2019.
[148] P. Peng et al., "Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games," 2017, *arXiv:1703.10069*.
[149] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Barcelona, Spain, vol. 29, 2016, pp. 2145–2153.
[150] S. Sukhbaatar and R. Fergus, "Learning multiagent communication with backpropagation," in *Proc. Adv. Neural Inf. Process. Syst.*, Barcelona, Spain, vol. 29, 2016, pp. 2252–2260.
[151] D. Chen, M. Nie, J. Yan, D. Wang, and Q. Gan, "Network representation learning algorithm based on complete subgraph folding," *Mathematics*, vol. 10, no. 4, 2022, Art. no. 581.
[152] W. Zhao, Y. Li, T. Fan, and F. Wu, "A novel embedding learning framework for relation completion and recommendation based on graph neural network and multi-task learning," *Soft Comput.*, pp. 1–13, 2022.
[153] A. Jain, N. Kosaka, K.-M. Kim, and J. J. Lim, "Know your action set: Learning action relations for reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2022.
[154] U. Gunarathna, R. Borovica-Gajic, S. Karunasekara, and E. Tanin, "Solving dynamic graph problems with multi-attention deep reinforcement learning," 2022, *arXiv:2201.04895*.
[155] P. Ammanabrolu and M. Hausknecht, "Graph constrained reinforcement learning for natural language action spaces," in *Proc. Int. Conf. Learn. Representations*, Addis Ababa, Ethiopia, 2020.
[156] G. S. Ramachandran, I. Brugere, L. R. Varshney, and C. Xiong, "GAEA: Graph augmentation for equitable access via reinforcement learning," in *Proc. AAAI/ACM Conf. AI, Ethics, Soc.*, Virtual Event, USA, 2021, pp. 884–894.
[157] Q. Ma, S. Ge, D. He, D. Thaker, and I. Drori, "Combinatorial optimization by graph pointer networks and hierarchical reinforcement learning," 2019, *arXiv:1911.04936*.
[158] Y. Xian, Z. Fu, S. Muthukrishnan, G. d. Melo, and Y. Zhang, "Reinforcement knowledge graph reasoning for explainable recommendation," in *Proc. ACM SIGIR Int. Conf. Res. Dev. Inf. Retr.*, Paris, FR, 2019, pp. 285–294.
[159] J. Jiang, C. Dun, T. Huang, and Z. Lu, "Graph convolutional reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, Addis Ababa, Ethiopia, 2020.
[160] P. Ammanabrolu and M. Riedl, "Playing text-adventure games with graph-based deep reinforcement learning," in *Proc. Conf. N. Am Chapter Assoc. Comput. Linguist.: Hum. Lang. Technol.*, Minneapolis, MN, USA, 2019, pp. 3557–3565.

Mingshuo Nie (Graduate Student Member, IEEE) received the master's degree in software engineering in 2021 from Software College, Northeastern University, Shenyang, China, where he is currently working toward the Ph.D. degree majoring in software engineering. His research interests include graph reinforcement learning, graph mining, link prediction, and graph neural networks.

Dongming Chen (Member, IEEE) received the Ph.D. degree in computer system architecture from Northeastern University, Shenyang, China, in 2006. He is currently a Professor with Software College, Northeastern University, China. He is a Member of IEEE Computer Society, Senior member of China Computer Federation (CCF), and Senior member of China Institute of Communications (CIC). His research interests include deep reinforcement learning, social network, and Big Data analysis.

Dongqi Wang received the Ph.D. degree in computer system architecture from Northeastern University, Shenyang, China, in 2011. He is an Associate Professor with Software College, Northeastern University. His research interests include complex networks and machine learning. He is also a Member of CCF.