Multitask Node Combinatorial Optimization in Complex Networks via Mixture-of-Experts

Xiaojie Yang, Zhijie Cao, Yeming Yang, Lingjie Li, *Member, IEEE*, Lijia Ma†, *Member, IEEE*

***Abstract*—The node combinatorial optimization (NCO) problem in complex networks tries to select a set of critical nodes to achieve objectives such as influence maximization, robustness enhancement, and edge coverage, etc. Owing to its significant applications in social network analysis, infrastructure protection, and system design, NCO has attracted extensive research attention. Recently, evolutionary deep reinforcement learning (EDRL) emerges as a promising approach by reformulating discrete NCO into continuous parameter optimization of deep Q networks. However, existing EDRL studies primarily focus on single-task scenarios, while how to efficiently solve multiple NCO problems simultaneously remains a significant challenge. To address this issue, we introduce a multi-gate mixture-of-experts (MMoE)-based multifactorial EDRL algorithm (called MMoE-EDRL) that employs a MMoE-based deep Q network (called MDQN) to model the multitask NCO problem as a weight optimization task. Specifically, the MDQN employs shared expert layers to learn general node representations across tasks and equips each task with an independent gating network to explicitly capture task similarities and differences. Subsequently, this MDQN is evolved by integrating EDRL with the multifactorial evolutionary algorithm (MFEA) for effective multitask learning (MTL). Experimental results across six real-world networks demonstrate the efficiency of MMoE-EDRL over state-of-the-art methods in addressing the NCO problems. This work provides an efficient solution for multitask node combinatorial optimization in complex networks.**

***Index Terms*—Complex Networks, Node Combinatorial Optimization, Evolutionary Deep Reinforcement Learning, Multi-gate Mixture-of-Experts, Multitask Learning**

I. Introduction

The study of complex networks involves a large number of interconnected nodes and their interaction patterns, widely existing in natural and social systems such as the internet, social networks, and metabolic networks [1]. Node combinatorial optimization (NCO) is a key focus in complex network research, addressing how to select critical node sets within a complex network to achieve specific objectives under constraints, such as influence maximization (IM) [2], network robustness analysis (NRA) [3], minimum vertex coverage (MVC) [4], minimum dominating set (MDS) [5], and maximum independent set (MIS) [6]. This class of problems is typically NP-hard, making exact solutions computationally prohibitive [7].

Traditional approaches to NCO problems include approximation and metaheuristic algorithms. Approximation algorithms (such as Greedy [8] and CELF++ [9]) usually operate by iteratively selecting the node that brings the greatest marginal gain. These approximation algorithms can provide theoretical performance guarantees, but their computational cost is often prohibitively high on large graphs. Metaheuristic algorithms (such as CMA-IM [10] and MA-IM [11]) search the solution space through mechanisms inspired by natural or physical processes. Metaheuristic algorithms offer greater flexibility and can often obtain high-quality solutions through search, yet their efficiency and convergence behavior tend to be unstable in complex solution spaces. In contrast, deep reinforcement learning (DRL)-based approaches like S2V-DQN [12] leverage deep Q-learning to directly infer solution policies, offering a data-driven alternative to manually designed heuristics.

In recent years, the intersection of evolutionary algorithm (EA) and DRL has given rise to evolutionary deep reinforcement learning (EDRL) [13]. Within the EDRL framework, prior research introduced EDRL-IM [14], an algorithm designed for NCO problems. EDRL-IM models the influence maximization problem as a continuous optimization of deep Q network (DQN) weights and employs a dynamic markov node selection strategy. It searches for the optimal seed set by co-optimizing the DQN parameters using an EA for exploration and a DRL component for exploitation. Research indicates that many NCO problems can be reduced to the NP-hard set cover problem, theoretically revealing inherent similarities among different NCO tasks [15]. However, most existing EDRL methods for solving NCO problems focus solely on single-task optimization, overlooking the potential synergies between tasks.

Multitask learning (MTL) can solve multiple related tasks simultaneously and enhance overall performance and convergence speed through knowledge transfer [16]. Based on this integration of MTL into the EDRL framework, subsequent research introduced MF-EDRL [15]. This algorithm combines the multifactorial evolutionary algorithm (MFEA) with EDRL and adopts a multi-head DQN architecture. In this design, a shared base network learns common features while task-

This work was supported by the National Natural Science Foundation of China under Grant 62173236 and Grant 62506238, the Shenzhen Natural Science Foundation under Grant JCYJ20240813141416022, the Tencent "Rhinoceros Bird" Scientific Research Foundation for Young Teachers of Shenzhen University, the Scientific Research Capacity Enhancement Program for Key Construction Disciplines in Guangdong Province under Grant 2024ZDJS063, and the Shenzhen Technology University School-level Research Project under Grant 20251061020002.

X.J. Yang, Z.J. Cao, Y.M. Yang, and L.J. Ma are affiliated with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China (email: 2400101067@mails.szu.edu.cn; 2500101011@mails.szu.edu.cn; yangyeming2021@email.szu.edu.cn; ljma1990@szu.edu.cn).

L.J. Li is affiliated with School of Artificial Intelligence, Shenzhen Technology University, Shenzhen 518118, China (email: lilingjie@sztu.edu.cn).

†: Corresponding Author.

specific output layers capture individual task characteristics, enabling the cooperative evolution of solutions for multiple tasks within a single population. However, this parameter-sharing mechanism may lead to parameter conflict. When task objectives differ significantly, parameters optimized for different tasks may conflict with each other, thereby limiting the potential for positive knowledge transfer [17]. Consequently, how to effectively integrate MTL into the EDRL framework to fully leverage task similarities in NCO problems remains a challenging area.

Mixture-of-experts (MoE) [18] is an ensemble paradigm that decomposes a monolithic network into a set of specialized sub-networks (experts) and employs a gating function to activate only the most relevant experts for each input. Sparse MoE [19] further introduces top-k sparsity and a differentiable gate to achieve conditional computation with negligible extra cost. In recent years, due to its property of dynamically combining different expert networks, the MoE model is considered to have inherent task adaptation potential in multi-task learning. It has been successfully applied to multi-task scenarios such as dense prediction [20] [21], single-cell multi-omics analysis [22], and large-scale recommendation systems [23]. Specifically, the gating mechanism of MoE allows the model to dynamically select and combine the most relevant experts for different data patterns, providing an effective mechanism to balance parameter sharing and specialization in multi-task learning. A prior research lays the groundwork for applying MoE to multitask problems by equipping each task with an independent gating network [17].

In this paper, we propose the multi-gate mixture-of-experts (MMoE)-based multifactorial EDRL algorithm (called MMoE-EDRL) inspired by the advantages of MoE in multitask learning, aiming to overcome the negative transfer problem caused by parameter conflict in existing multitask EDRL studies. Specifically, MMoE-EDRL models the solution to each NCO task as an MMoE-based DQN (called MDQN), whose parameters are optimized within a unified evolutionary multitasking framework. The main contributions of this paper are as follows:

- We formulate multitask NCO problems as parameter optimization of MDQNs, where each MDQN approximates an action-value function and obtains a set of key nodes through a Markov decision process (MDP).
- We propose MMoE-EDRL, a novel multitask EDRL algorithm. It enables implicit knowledge transfer across tasks by evolving a population of MDQNs, each designed with an MMoE structure. Within this structure, a shared expert pool captures common knowledge while task-specific gating networks explicitly mitigate negative transfer caused by parameter conflict.
- We conduct extensive experiments on six real-world networks and demonstrate the clear superiority of MMoE-EDRL over alternative state-of-the-art methods.

The remainder of this paper is organized as follows. Section II introduces the preliminaries. Section III presents the MDQN and the proposed MMoE-EDRL algorithm in detail. Section IV reports the experimental results. Section V concludes the paper.

Upon acceptance of this paper, we will make our data and code publicly available.

II. PRELIMINARIES

A. Node Combinatorial Optimization Problems

NCO problems represent a class of NP-hard combinatorial challenges defined on graphs, where the goal is to select a subset of nodes that optimizes a specific network property [24]. Formally, given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, an NCO problem aims to find a node set $\mathcal{S} \subseteq \mathcal{V}$ that minimizes or maximizes an objective function $f(\mathcal{S})$, typically subject to a problem-specific constraint. In this paper, we focus on five fundamental and representative NCO problems, which have distinct objectives and constraints as follows:

- IM: Under the fast approximation IC model, select a seed set $\mathcal{S}$ with $|\mathcal{S}| = k$ to maximize the expected number of activated nodes $\sigma(\mathcal{S})$ [14]. The objective is $f_{\text{IM}}(\mathcal{S}) = \sigma(\mathcal{S})$.
- NRA: Identify a set $\mathcal{S}$ (with $|\mathcal{S}| = k$) whose removal minimizes the size of the largest remaining connected component $C(\mathcal{G} \setminus \mathcal{S})$ [3]. The objective is transformed into maximization: $f_{\text{NRA}}(\mathcal{S}) = |\mathcal{V}| - C(\mathcal{G} \setminus \mathcal{S})$.
- MVC: Find the smallest set $\mathcal{S}$ such that every edge in $\mathcal{E}$ is incident to at least one node in $\mathcal{S}$. The objective is transformed into maximization: $f_{\text{MVC}}(\mathcal{S}) = |\mathcal{V}| - |\mathcal{S}|$.
- MDS: Find the smallest set $\mathcal{S}$ such that every node in $\mathcal{V}$ is either in $\mathcal{S}$ or adjacent to a node in $\mathcal{S}$. The objective is transformed into maximization: $f_{\text{MDS}}(\mathcal{S}) = |\mathcal{V}| - |\mathcal{S}|$.
- MIS: Find the largest set $\mathcal{S}$ such that no two nodes in $\mathcal{S}$ are adjacent. The objective is $f_{\text{MIS}}(\mathcal{S}) = |\mathcal{S}|$.

B. Multifactorial Evolutionary Algorithm

MFEA [25] is a leading algorithm for evolutionary multitasking. It encodes solutions from different tasks in a unified search space, allowing a single population to address multiple tasks concurrently. The algorithm operates through two key mechanisms. First, assortative mating guides reproduction such that individuals sharing the same skill factor are more likely to mate, fostering task-specific refinement, while cross-task mating occurs with a probability determined by the random mating probability (rmp) parameter, which controls knowledge transfer between tasks. Second, vertical cultural transmission ensures that each offspring inherits the skill factor of one parent and is evaluated solely on that corresponding task. This framework enables efficient implicit knowledge transfer while avoiding the computational cost of evaluating every individual across all tasks.

III. PROPOSED METHOD

A. MMoE Module

The core of the proposed approach is the MMoE module, which employs the MDQN to approximate the action-value function $\mathbf{Q}(S_t^j, a^j)$ for each task j. This function evaluates

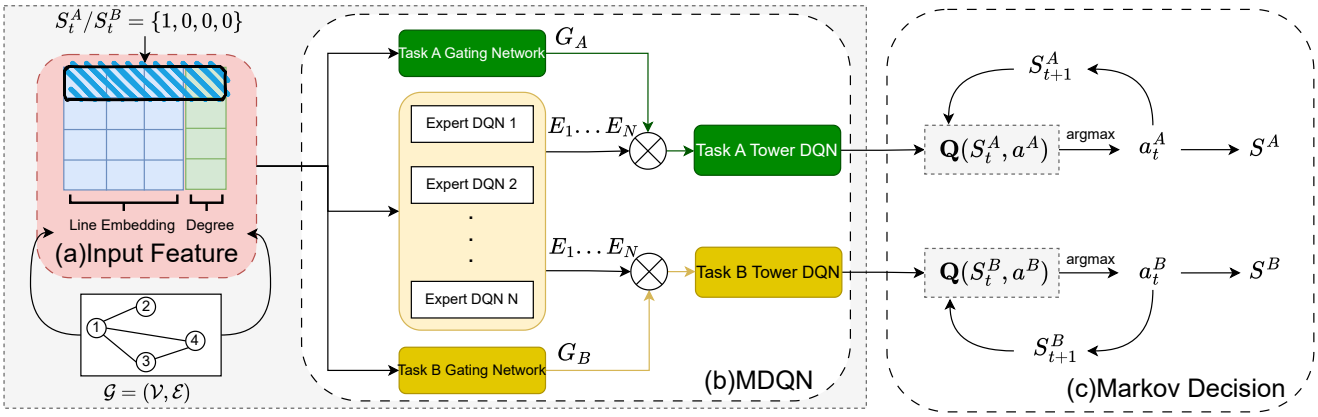

Fig. 1. The overview of the MMoE module for two NCO problems. The figure illustrates the three-stage pipeline: (a) Input Feature Preparation, (b) the MDQN representation with shared experts and task-specific gates/towers, and (c) the Markov Node Selection process.

the expected long-term utility of selecting a node a^j given the current task state S_t^j. Unlike a conventional DQN, the MDQN employs a shared pool of expert networks to learn general node representations across tasks, while each task is equipped with an independent gating network to dynamically combine these experts, thereby explicitly capturing task similarities and differences. This design provides a flexible parameter-sharing mechanism that mitigates inter-task conflict. The MMoE module comprises three sequential stages: input feature preparation, MDQN representation, and the Markov node selection process. Figure 1 provides an overview of the proposed MMoE module, illustrating its three-stage architecture.

1) Input Feature Preparation: The MDQN approximates the action-value function $\mathbf{Q}(S_t^j, a^j)$. Its input is the node selection state S_t^j for task j at step t, represented as a binary vector of selected seed nodes. Incorporating this state prevents node reselection. To assess the action-value of selecting a particular node, the module must also account for the network structure. Therefore, the MDQN's input features are constructed by fusing this state information with the network's topological features in two steps.

First, we use the Line method [26] to learn low-dimensional node embeddings, capturing the network topological information. The process is defined as follows:

$$\mathrm{L} = \mathbf{Line}(\mathrm{A}, d), \tag{1}$$

where $\mathbf{Line}(\cdot)$ denotes the Line embedding function, A is the network adjacency matrix, d is the embedding dimension, $n = |\mathcal{V}|$, and $\mathrm{L} \in \mathbb{R}^{n \times d}$ is the resulting node embedding matrix. We concatenate L with node degree features D to form an enriched structural feature matrix $\mathrm{L}^D = [\mathrm{L}\ D]$.

Second, a masking operation integrates the current selection state S_t^j into L^D. This operation nullifies the features of already-selected nodes to exclude them from further consideration.The masked operation is computed as follows:

$$\mathrm{L}_t^j = \mathbf{\Phi}(\mathrm{L}^D, S_t^j) = \mathrm{L}^D \odot (1 - s_t^j), \tag{2}$$

where s_t^j is the binary vector for state S_t^j, $\odot$ denotes element-wise multiplication broadcast across the feature dimension, and L_t^j is the masked feature matrix containing only information from unselected nodes. This matrix L_t^j, which jointly encodes the network structure and current decision state, serves as the input to the subsequent MDQN in the MMoE module.

2) MDQN Representation: The MDQN representation transforms the masked features into the final action-value scores through three components: shared expert DQNs, task-specific gating network, and task-specific tower DQN.

Shared Expert DQNs: A set of N shared expert networks $\{E_n\}_{n=1}^N$ processes the input. Each expert extracts a distinct pattern from the features. The output of the n-th expert is given as follows:

$$E_n = \mathbf{Relu}\left(\mathrm{L}_t^j \cdot W_{n1}\right) \cdot W_{n2}, \quad \text{for } n = 1, 2, ..., N, \tag{3}$$

where W_{n1} and W_{n2} are the weight matrices of expert E_n, and $\mathbf{Relu}(\cdot)$ denotes the rectified linear unit activation function.

Task-specific Gating Network: For each task j, a gating network computes adaptive weights to combine the expert outputs. The gating weights are calculated as follows:

$$G_j = \mathbf{Softmax}\left(\mathbf{Relu}\left(L_t^j \cdot V_{j1}\right) \cdot V_{j2}\right), \tag{4}$$

where V_{j1} and V_{j2} are the parameters of the gating network. The output $G_j = [g_{j1}, ..., g_{j\mathrm{N}}]$ is a probability distribution over the N experts, satisfying $\sum_{n=1}^{\mathrm{N}} g_{jn} = 1$.

Task-specific Tower DQN: The final action-value scores for task j are produced by its dedicated tower network. First, the expert outputs are aggregated using the gating weights, and then the mixture is processed by the tower network. This is formulated in a single step as follows:

$$\mathbf{Q}(S_t^j, a^j) = \mathbf{Relu}\left(\left(\sum_{n=1}^{\mathrm{N}} g_{jn} \cdot E_n\right) \cdot U_{j1}\right) \cdot U_{j2}, \tag{5}$$

where U_{j1} and U_{j2} are the weight matrices of the tower network. The inner sum $\sum_{n=1}^{\mathrm{N}} g_{jn} \cdot E_n$ represents the weighted combination of expert features, which is then non-linearly transformed by the tower network to yield the final Q-values.

3) Markov Node Selection: The MMoE module is integrated into a MDP to sequentially select the set of key nodes for each NCO task. At each step t of task j, the process begins with the current state S_t^j. This state is first processed through the input feature preparation stage (Section III-A1) to yield the masked feature matrix L_t^j. Subsequently, this matrix is fed into the MDQN representation (Section III-A2), which computes the action-value $\mathbf{Q}(S_t^j, a^j)$ for every candidate node a^j. The action (node to select) is determined by choosing the one with the highest estimated value. This process can be summarized as follows:

$$\begin{aligned} \mathbf{Q}(S_t^j, a^j) &= \mathbf{MDQN}(S_t^j; \Theta), \\ a_t^j &= \arg\max_{a^j} \mathbf{Q}(S_t^j, a^j), \end{aligned} \tag{6}$$

where Θ collectively denotes all trainable parameters of the MDQN. Executing the selected action a_t^j updates the state to S_{t+1}^j. This evaluate-select-update loop is repeated until the constraints of the NCO problem are violated.

Therefore, the process of solving a multitask NCO problem is effectively reformulated as the optimization of the continuous parameters Θ of the MDQN. The goal of this optimization is to find Θ that, when used within the described MDP, yields a node set $\mathcal{S}$ that maximize the objective functions $f_j(\mathcal{S})$ (detailed in Section II-A) for all tasks j simultaneously.

B. MMoE-EDRL

Based on the established MMoE Module, we propose MMoE-EDRL, an EDRL algorithm that leverages multitask optimization to evolve a population of MDQNs for solving multiple NCO tasks simultaneously. By utilizing MDQN, MMoE-EDRL facilitates efficient knowledge transfer across tasks while mitigating negative interference.

Algorithm 1 outlines the complete procedure of MMoE-EDRL. The algorithm begins by initializing a population of N individuals, each encoding a complete MDQN parameter set (line 1). Each individual is then evaluated on all K tasks, and the task on which it performs best is assigned as its skill factor τ (lines 2-5). In each generation, offspring are generated through assortative mating (lines 8-17). For each pair of parent individuals p_a and p_b, recombination occurs if the two parents are specialized in the same task or if the random mating condition (governed by the rmp parameter) is satisfied. When this condition is satisfied, a parameter-wise crossover is applied, ensuring that only corresponding parameters are exchanged (e.g., expert weights with expert weights), thereby preserving the functional semantics of the MDQN architecture. Otherwise, only mutation is applied to each parent independently. Each offspring then imitates the skill factor of one parent and is evaluated only on that imitated task . Finally, the parent and offspring populations are merged, and the N fittest individuals are selected to form the next generation (line 18-19). This evolutionary cycle repeats until a predefined termination criterion is met. The algorithm ultimately outputs the best-evolved MDQN parameters for each task, which represent high-quality policies for solving the corresponding NCO problems.

Algorithm 1 Basic Structure of MMoE-EDRL

Input: K NCO tasks, population size N, random mating probability rmp
Output: Best MDQN for each task

1: Initialize population P of N individuals, each p_i being a parameter vector Θ encoding an MDQN.
2: **for** each $p_i \in P$ **do**
3: Evaluate p_i on all K NCO tasks.
4: Assign skill factor τ_{p_i}.
5: **end for**
6: **repeat**
7: Initialize an offspring population $C \leftarrow \varnothing$.
8: **for** each mating pair (p_a, p_b) from P **do**
9: **if** $\tau_{p_a} = \tau_{p_b}$ **or** $\mathrm{rand}() < rmp$ **then**
10: Apply crossover operator to produce offspring o_1, o_2. // See Algorithm 2
11: **else**
12: Apply mutation operator to produce offspring o_1, o_2. // See Algorithm 3
13: **end if**
14: Assign o_1, o_2 skill factors from one parent.
15: Evaluate o_1, o_2 on their imitated tasks only.
16: Add o_1, o_2 to C.
17: **end for**
18: Merge P and C, and compute scalar fitness for all individuals.
19: Select the top N individuals from the merged set as next P.
20: **until** termination criterion met
21: **return** Best MDQN for each task.

In the following subsections, the core mechanisms of MMoE-EDRL are detailed.

1) Individual Representation: In MMoE-EDRL, an individual p_i in the population is encoded as the complete set of trainable parameters of an MDQN. Formally, an individual p_i specialized in task j ($\tau_{p_i} = j$) is defined by the parameter vector as follows:

$$p_i = \left[\{W_{n1}, W_{n2}\}_{n=1}^{\mathrm{N}},\ V_{j1}, V_{j2},\ U_{j1}, U_{j2} \right], \tag{7}$$

where $\{W_{n1}, W_{n2}\}_{n=1}^{\mathrm{N}}$ denote the weight matrices of the N shared expert networks that capture general node representations across tasks; V_{j1}, V_{j2} are the parameters of the task-specific gating network for task j, which adaptively combines the expert outputs; and U_{j1}, U_{j2} correspond to the weights of the task-specific tower network that produces the final action-value scores for task j.

2) Crossover and Mutation Operators: The evolutionary search in MMoE-EDRL is guided by specialized crossover and

Algorithm 2 Crossover

Input: Two parent individuals p_a and p_b, each structured as $[\{W_{n1}, W_{n2}\}_{n=1}^{\mathrm{N}}, V_{j1}, V_{j2}, U_{j1}, U_{j2}]$; crossover probability p_c
Output: Two offspring individuals p'_a and p'_b

Copy p_a to p'_a and p_b to p'_b.
for each parameter x_i in p'_a and y_i in p'_b **do**
  if $\mathrm{rand}() < p_c$ **then**
    Swap x_i and y_i.
  end if
end for
return p'_a, p'_b

Algorithm 3 Mutation

Input: A parent individual p_i, structured as $[\{W_{n1}, W_{n2}\}_{n=1}^{\mathrm{N}}, V_{j1}, V_{j2}, U_{j1}, U_{j2}]$; mutation probability p_m
Output: Mutated offspring individual p'_i

Copy p_i to p'_i.
for each parameter x_i in p'_i **do**
  if $\mathrm{rand}() < p_m$ **then**
    $x_i \leftarrow$ sample from $\mathcal{U}(-1, 1)$
  end if
end for
return p'_i

mutation operators designed to respect the semantic structure of the MDQN representation.

Crossover operator (Algorithm 2) is triggered when two parents are eligible to mate. It performs exchanges at the level of individual parameter, such as weight matrices within expert networks. For each aligned parameter pair (x_i, y_i) drawn from parents p_a and p_b, crossover is restricted strictly between x_i and y_i. This constraint ensures that expert parameters recombine only with expert parameters, and task-specific parameters only with their counterparts from the same component, thereby maintaining the functional roles of the MDQN modules and enabling meaningful, component-aware knowledge transfer. Mutation operator (Algorithm 3) serves as a fundamental variation operator, applied either after crossover or when cross-task mating is disallowed. Each parameter x_i in an individual is replaced with probability p_m by a random value drawn uniformly from $[-1, 1]$. This operation introduces controlled stochasticity into the search process, fostering exploration of the policy parameter space and aiding escape from local optima.

3) Individual Evaluation: Each individual p_i in MMoE-EDRL is evaluated to fulfill the requirement of the evolutionary iteration. The evaluation proceeds in two concrete steps. First, the parameter vector Θ of the individual p_i is decoded to instantiate its MDQN module, which then acts as the policy within the MDP defined in Section III-A3. Executing this policy yields a sequentially selected set of key nodes $\mathcal{S}$. Second, according to the skill factor of p_i $\tau_{p_i} = j$, the quality of the solution set $\mathcal{S}$ is measured using the corresponding task-specific objective function f_j defined for the NCO problems in Section II-A. The resulting value $f_j(\mathcal{S})$ provides the performance metric for p_i and is used to drive the selection process in the evolutionary loop.

IV. COMPUTATIONAL EXPERIMENTS

A. Experimental Settings

1) Experimental Networks: To evaluate algorithm performance across the five NCO tasks, we test all algorithms on six real-world networks (such as Polbooks, Elegans, Email, Power, Wiki, and Pgp [27]) from social, biological, and infrastructural domains, reflecting diverse real-world communication patterns. Detailed statistics are provided in Table I.

TABLE I
PROPERTIES OF THE SIX REAL-WORLD NETWORKS USED IN OUR EXPERIMENTS.

Network	Nodes	Edges	Avg. Degree
polbooks	105	441	8.40
elegans	453	2,025	8.94
email	1,133	5,451	9.62
power	4,941	6,594	2.67
wiki	7,115	103,689	29.15
pgp	10,680	20,340	4.54

2) Baseline Algorithms and Parameter Setting: Seven representative algorithms are selected as baseline methods, including two greedy-based heuristics (Degree and PageRank [28]), two metaheuristic algorithms (MA-IM [11] and CMA-IM [10]), one DRL-based algorithm (S2V-DQN [12]), and two state-of-the-art EDRL algorithms (EDRL-IM [14] and MF-EDRL [15]). The comparison with greedy, metaheuristic, and DRL algorithms aims to demonstrate the superiority of the evolutionary reinforcement learning paradigm for NCO tasks. Meanwhile, the comparison with the single-task solver EDRL-IM is to validate the effectiveness of the multi-task learning paradigm. Furthermore, the comparison with the multitask baseline MF-EDRL is designed to demonstrate the superior performance of our method, as MMoE-EDRL can effectively mitigate negative transfer induced by parameter conflicts via its MDQN architecture. For all comparison algorithms, we use the parameter settings specified in their original papers. To ensure a fair comparison, for all EA-based algorithms among the baselines, the population size and the number of generations are set to 100 and 200, respectively. For our method, to align with MF-EDRL for a direct comparison, we set the number of evaluations to be the same, with a population size of 100, rmp=0.3, and crossover and mutation probabilities of 0.8 and 0.2, respectively. The MDQN in our algorithm is configured with 4 expert networks. For the IM and NRA tasks, the seed/attack set sizes are set to 10 and $\lceil 0.01 \times n \rceil$ (where n is the number of nodes), respectively.

3) Criteria: For each NCO task, we use its corresponding objective function value in Section II-A to evaluate algorithm performance, with higher values indicating better results. The

TABLE II

COMPARATIVE PERFORMANCE OF MMOE-EDRL AND BASELINE METHODS ACROSS SIX REAL-WORLD NETWORKS. IN THE RESULTS, '+', '-' AND '=' INDICATE THAT THE COMPARISON ALGORITHM PERFORMS BETTER THAN, WORSE THAN, AND SIMILAR TO MMOE-EDRL, RESPECTIVELY. THE BEST RESULTS ARE MARKED IN BOLDFACE.

Task	Network dataset	Degree	PageRank	MA-IM	CMA-IM	S2V-DQN	EDRL-IM	MF-EDRL	MMoE-EDRL
IM	Polbooks	36.22(-)	36.17(-)	35.69(-)	35.21(-)	36.56(-)	36.84(-)	36.91(-)	**37.13**
	Elegans	78.79(-)	28.56(-)	70.56(-)	82.02(-)	43.48(-)	**82.85**(=)	82.70(-)	**82.85**
	Email	131.3(-)	129.6(-)	127.6(-)	134.2(-)	131.3(-)	134.1(-)	134.5(-)	**134.75**
	Power	20.57(-)	10.20(-)	15.78(-)	20.04(-)	11.47(-)	20.57(-)	**20.66**(=)	**20.66**
	Wiki	**2872**(=)	252.7(-)	1372(-)	2825(-)	785.8(-)	**2872**(=)	**2872**(=)	**2872**
	Pgp	164.6(-)	12.22(-)	90.24(-)	154.9(-)	44.79(-)	169.8(-)	172.3(-)	**174.17**
NRA	Polbooks	65(-)	60(-)	68(-)	58(-)	34(-)	67(-)	69(-)	**70**
	Elegans	432(-)	434(-)	409(-)	254(-)	257(-)	**439**(=)	**439**(=)	**439**
	Email	493(-)	488(-)	423(-)	338(-)	399(-)	525(-)	539(-)	**544**
	Power	1328(-)	1095(-)	487(-)	748(-)	1109(-)	2011(-)	2948(-)	**3776**
	Wiki	6785(-)	7054(-)	3348(-)	3253(-)	6797(-)	**7081**(+)	7050(-)	7065
	Pgp	10453(-)	10289(-)	3351(-)	5412(-)	6201(-)	**10598**(+)	10563(-)	10585
MVC	Polbooks	41(-)	42(-)	/(-)	/(-)	38(-)	**43**(=)	**43**(=)	**43**
	Elegans	195(-)	193(-)	/(-)	/(-)	187(-)	199(-)	201(-)	**202**
	Email	520(-)	520(-)	/(-)	/(-)	504(-)	525(-)	**529**(+)	528
	Power	2654(-)	2672(-)	/(-)	/(-)	2491(-)	2676(-)	2687(-)	**2689**
	Wiki	**4874**(+)	4759(-)	/(-)	/(-)	4838(-)	/(-)	4812(-)	4850
	Pgp	6305(-)	6314(-)	/(-)	/(-)	5888(-)	/(-)	**6323**(+)	6322
MDS	Polbooks	84(-)	81(-)	/(-)	/(-)	80(-)	87(-)	89(-)	**90**
	Elegans	300(-)	361(-)	/(-)	/(-)	312(-)	**365**(=)	364(-)	**365**
	Email	757(-)	777(-)	/(-)	/(-)	735(-)	792(-)	787(-)	**806**
	Power	3111(-)	3136(-)	/(-)	/(-)	2758(-)	3146(-)	3152(-)	**3162**
	Wiki	2475(-)	2953(-)	/(-)	/(-)	3130(-)	**3784**(+)	3605(-)	3755
	Pgp	6598(-)	**6714**(+)	/(-)	/(-)	5423(-)	6622(-)	6637(-)	6654
MIS	Polbooks	40(-)	42(-)	/(-)	/(-)	34(-)	**43**(=)	**43**(=)	**43**
	Elegans	196(-)	194(-)	/(-)	/(-)	189(-)	201(-)	201(-)	**202**
	Email	519(-)	520(-)	/(-)	/(-)	494(-)	528(-)	529(-)	**532**
	Power	2672(-)	2675(-)	/(-)	/(-)	2319(-)	2681(-)	**2691**(+)	2690
	Wiki	4759(-)	4753(-)	/(-)	/(-)	4782(-)	**4824**(+)	4794(-)	4821
	Pgp	6311(-)	6315(-)	/(-)	/(-)	5727(-)	6323(-)	6323(-)	**6325**
best/all	/	2/30	1/30	0/30	0/30	0/30	10/30	8/30	21/30
+/-/=	/	1/28/1	1/29/0	0/30/0	0/30/0	0/30/0	4/20/6	3/22/5	/

Wilcoxon rank-sum test at a 5% significance level is employed to assess statistical significance, denoted by '+', '-', and '=' indicating the baseline performs significantly better, worse, or similarly to our method, respectively. Furthermore, an overall performance ranking is calculated across all test instances to provide a comprehensive comparison.

B. Experimental Results

Table II summarizes the average performance of all algorithms over 30 independent runs on six real-world networks, with the best results highlighted in bold.

From Table II, we can make the following experimental observations. First, classical heuristics (Degree, PageRank [28]) deliver stable results across all tasks but exhibit a clear performance ceiling, indicating their inability to model complex network patterns despite theoretical soundness. In contrast, metaheuristic algorithms (MA-IM [11], CMA-IM [10]) show effectiveness only on IM and NRA tasks, a limitation stemming from their heavy reliance on problem-specific design. Meanwhile, the DRL-based method S2V-DQN [12] is competitive on smaller networks but falters on larger, complex ones, as its gradient-based optimizer is prone to local optima. This limitation is directly addressed by the EDRL paradigm. All EDRL-based algorithms (EDRL-IM [14], MF-EDRL [15], and MMoE-EDRL) consistently outperform the pure DRL baseline S2V-DQN, validating the critical role of evolutionary search for global exploration. Furthermore, the multitask baseline MF-EDRL generally surpasses the single-task baseline EDRL-IM, demonstrating the benefit of MTL. Most notably, MMoE-EDRL demonstrates clear advantages over the multitask baseline MF-EDRL. It performs better in 22 instances, ties in 5, and is slightly worse in only 3 (each

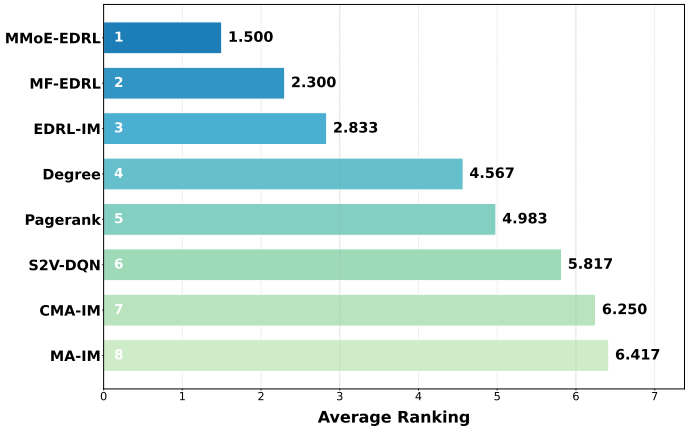

Fig. 2. The average performance ranking of all algorithms across all real-world networks, where lower values indicate better performance.

by a margin of one node). Crucially, this performance gain increases notably on larger and more structurally complex networks. These indicate that the MDQN architecture specifically addresses the severe parameter conflict in fully-shared models.

Overall, MMoE-EDRL achieves the best results in 21 out of 30 test instances, whereas the best-performing baseline obtains the top result in at most 10 instances. The Wilcoxon rank-sum test (with a significance level of 5%) further confirms the statistical advantage of MMoE-EDRL.

To provide an overall performance perspective across all tasks and networks, we present the average ranking of each algorithm in Figure 2. MMoE-EDRL attains the best average ranking of 1.5, which is significantly better than that of MF-EDRL (2.3) and all other single-task baselines.

In summary, these results validate the superiority of MMoE-EDRL. They demonstrate not only the effectiveness of the multi-task EDRL paradigm but, more importantly, the pivotal role of the MDQN architecture in mitigating parameter conflict, confirming the efficacy of integrating MoE with EDRL.

V. CONCLUSION

Evolutionary Deep Reinforcement Learning (EDRL) has shown promising results in solving Node Combinatorial Optimization (NCO) problems. However, existing multitask EDRL approaches can suffer from parameter conflict, which hinders effective knowledge transfer and limits overall performance. To address this issue, this study leverages the advantage of Mixture-of-Experts (MoE) in multitask optimization. Specifically, we reformulate multitask NCO as a parameter optimization problem based on multi-gate mixture-of-experts (MMoE)-based DQN (called MDQN) and propose the MMoE-based multifactorial EDRL algorithm (called MMoE-EDRL). This algorithm mitigates the negative transfer caused by parameter conflicts in multi-task models that employ fully-shared base layers. Our experimental results demonstrate that MMoE-EDRL outperforms other baseline algorithms on six real-world networks. Future work will focus on extending this approach to dynamic network scenarios.

REFERENCES

[1] S. H. Strogatz, "Exploring complex networks," *Nature*, vol. 410, no. 6825, pp. 268–276, 2001.

[2] Y. Li, J. Fan, Y. Wang, and K.-L. Tan, "Influence maximization on social graphs: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 10, pp. 1852–1872, 2018.

[3] J. Mao, D. Zou, L. Sheng, S. Liu, C. Gao, Y. Wang, and Y. Li, "Identify critical nodes in complex network with large language models," *arXiv Preprint arXiv:2403.03962*, 2024.

[4] R. Sun, P. Liu, Y. Wang, Z. Liu, L. Du, and J. Gao, "Infvc: An inference-enhanced local search algorithm for the minimum vertex cover problem in massive graphs," in *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, 2025, pp. 8977–8986.

[5] F. Grandoni, "A note on the complexity of minimum dominating set," *Journal of Discrete Algorithms*, vol. 4, no. 2, pp. 209–214, 2006.

[6] D. V. Andrade, M. G. Resende, and R. F. Werneck, "Fast local search for the maximum independent set problem," *Journal of Heuristics*, vol. 18, no. 4, pp. 525–547, 2012.

[7] Y. Bengio, A. Lodi, and A. Prouvost, "Machine learning for combinatorial optimization: a methodological tour d'horizon," *European Journal of Operational Research*, vol. 290, no. 2, pp. 405–421, 2021.

[8] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003, pp. 137–146.

[9] A. Goyal, W. Lu, and L. V. Lakshmanan, "Celf++ optimizing the greedy algorithm for influence maximization in social networks," in *Proceedings of the 20th International Conference Companion on World Wide Web*, 2011, pp. 47–48.

[10] M. Gong, C. Song, C. Duan, L. Ma, and B. Shen, "An efficient memetic algorithm for influence maximization in social networks," *IEEE Computational Intelligence Magazine*, vol. 11, no. 3, pp. 22–33, 2016.

[11] S. Wang, J. Liu, and Y. Jin, "Finding influential nodes in multiplex networks using a memetic algorithm," *IEEE Transactions on Cybernetics*, vol. 51, no. 2, pp. 900–912, 2019.

[12] E. Khalil, H. Dai, Y. Zhang, B. Dilkina, and L. Song, "Learning combinatorial optimization algorithms over graphs," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[13] Q. Zhu, X. Wu, Q. Lin, L. Ma, J. Li, Z. Ming, and J. Chen, "A survey on evolutionary reinforcement learning algorithms," *Neurocomputing*, vol. 556, p. 126628, 2023.

[14] L. Ma, Z. Shao, X. Li, Q. Lin, J. Li, V. C. Leung, and A. K. Nandi, "Influence maximization in complex networks by using evolutionary deep reinforcement learning," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 7, no. 4, pp. 995–1009, 2022.

[15] L. Ma, L. Xu, X. Fan, L. Li, Q. Lin, J. Li, and M. Gong, "Multifactorial evolutionary deep reinforcement learning for multitask node combinatorial optimization in complex networks," *Information Sciences*, vol. 702, p. 121913, 2025.

[16] Y. Zhang and Q. Yang, "A survey on multi-task learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 12, pp. 5586–5609, 2021.

[17] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, and E. H. Chi, "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1930–1939.

[18] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[19] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, "The sparsely-gated mixture-of-experts layer," *Outrageously Large Neural Networks*, vol. 2, 2017.

[20] Y. Yang, P.-T. Jiang, Q. Hou, H. Zhang, J. Chen, and B. Li, "Multi-task dense prediction via mixture of low-rank experts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 27 927–27 937.

[21] H. Ye and D. Xu, "Taskexpert: Dynamically assembling multi-task representations with memorial mixture-of-experts," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 21 828–21 837.

[22] S. Yun, J. Peng, N. Lee, Y. Zhang, C. Park, Z. Liu, and T. Chen, "scmoe: single-cell multi-modal multi-task learning via sparse mixture-of-experts," *bioRxiv*, pp. 2024–11, 2024.

[23] Z. Zhang, S. Liu, J. Yu, Q. Cai, X. Zhao, C. Zhang, Z. Liu, Q. Liu, H. Zhao, L. Hu *et al.*, "M3oe: Multi-domain multi-task mixture-of-experts recommendation framework," in *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024, pp. 893–902.

[24] B. Korte and J. Vygen, *Combinatorial optimization: theory and algorithms*. Springer, 2008.

[25] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: Toward evolutionary multitasking," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 343–357, 2015.

[26] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 1067–1077.

[27] L. Ma, M. Gong, J. Liu, Q. Cai, and L. Jiao, "Multi-level learning based memetic algorithm for community detection," *Applied Soft Computing*, vol. 19, pp. 121–133, 2014.

[28] S. Brin, "The pagerank citation ranking: bringing order to the web," *Proceedings of ASIS*, vol. 98, pp. 161–172, 1998.