International Journal of Robotics and Automation, Vol. 40, No. 10, 2025
An Open Access Paper

DYNAMIC COMMUNITY DETECTION-DRIVEN FRAMEWORK FOR COLLABORATIVE PLANNING AND ADAPTIVE CONTROL OF UAV SWARMS

Haonan Liu,*,**,*** Xiaoyu Li,*** Li Jin,*** Wen Shi,*** Yang Bai,***
Linhao Zhang,*,**,*** and Hongqi Wang***

Abstract

UAVs (unmanned aerial vehicle) are widely employed in disaster relief, agricultural monitoring, logistics distribution, and military reconnaissance. With the expansion of UAV network scales, there are heightened demands for task allocation, path optimisation, and system robustness. The intrinsic community structures within UAV networks play a vital role in enhancing task efficiency and network performance. In order to use the potential community of UAVs to assist in scheduling UAV network scheduling, we propose a novel semi-supervised community construction method based on iterative cross-layer graph contrastive learning. This method integrates graph-level and node-level information to identify key nodes and community structures, thereby optimising task allocation, path planning, and system performance. Experimental results on various datasets demonstrate the methods efficiency in handling large-scale UAV network data and meeting real-time and robustness requirements. It holds significant potential in the realms of intelligent UAV collaboration and automated control.

Key Words

Cross-layer graph contrastive learning, multi-UAV collaborative networks, graph neural networks, dynamic community detection

* Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100190, China; e-mail: liuhaonan21@mails.ucas.ac.cn; zhanglinhao20@mails.ucas.ac.cn
** School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100190, China
*** Key Laboratory of Target Cognition and Application Technology (TCAT), Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100190, China; e-mail: lixy01@aircas.ac.cn; jinlimails@gmail.com; shiwen@aircas.ac.cn; baiyang@aircas.ac.cn; 2705346188@qq.com
Corresponding author: Wen Shi

Recommended by Shichao Liu.
(DOI: 10.2316/J.2025.206-1308)

1. Introduction

With the exponential growth of unmanned aerial vehicle (UAV) technology, multi-UAV networks have become increasingly prevalent in critical applications such as disaster response systems, precision agriculture monitoring, autonomous logistics, and intelligent surveillance [1]. These networks rely on collaborative task execution among heterogeneous UAV swarms to achieve complex missions including target tracking, environmental sensing, and payload delivery. However, scaling such systems introduces significant challenges in dynamic task allocation, real-time path optimisation, and fault-tolerant operation [2]. A key observation is that UAV collaboration patterns often form hierarchical community structures characterised by dense intra-group interactions and sparse inter-group connections, which are shaped by topological proximity, communication bandwidth constraints, and mission objectives [3], [4]. Identifying these latent community structures is essential for optimising system performance through localised task scheduling, energy-efficient routing, and adaptive resource management.

Community detection, a fundamental problem in complex network analysis, aims to partition graphs into cohesive subgraphs with high internal connectivity [5]. In the context of UAV networks, this process enables the discovery of functional clusters that share common operational requirements. For instance, geographically proximate UAVs forming a community can be assigned interdependent tasks to minimise communication latency and energy expenditure. Moreover, detecting central nodes within communities allows for priority-based resource allocation, while dynamic community adaptation enhances system resilience against node failures or environmental changes [6]. These capabilities are particularly critical for real-time systems where mission success depends on efficient coordination.

Existing community detection algorithms face significant limitations when applied to UAV networks.

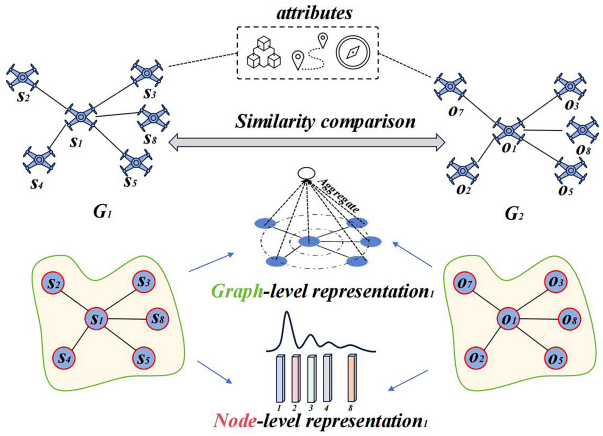

Figure 1. Similarity comparison of graphs. Graph-level representation and node-level representation are the features of two levels.

Traditional methods such as vGraph lack scalability and fail to capture dynamic topological changes inherent in mobile networks [7]. Graph embedding techniques like DeepWalk, while theoretically sound, struggle with temporal variations and require prohibitive computational resources for large-scale deployments [8], [9]. Furthermore, most approaches ignore the hierarchical nature of UAV networks [10], where both global topological properties and local node features contribute to community formation. Addressing these challenges requires a novel approach that integrates multi-scale graph representations with adaptive learning mechanisms.

To bridge this gap, we propose a semi-supervised community construction method based on iterative cross-layer graph contrastive learning (CGNet), as shown in Fig. 1, aiming to provide efficient support for multi-UAV collaboration and automated system optimisation. By combining graph-level and node-level information and designing a cross-layer contrastive loss function, the proposed method can efficiently identify key nodes and community structures in UAV networks, thereby supporting task allocation, path planning, and system optimisation.

The contributions of this paper are summarised as follows.

(1) The research presented in this paper provide a new solution for intelligent UAV collaboration and automated control through the detection and construction of UAV community networks, with broad application potential in scenarios such as disaster relief and logistics distribution.

(2) The designed cross-layer contrastive learning framework, combining graph-level and node-level information, enhances detection accuracy while continuously optimising model performance through an iterative mechanism to adapt to the dynamic changes in UAV networks.

(3) Experiments on multiple public datasets demonstrate that the proposed method exhibits significant advantages in handling large-scale network data and meets the real-time and robustness requirements of drone systems.

2. Related Work

2.1 Semi-supervised Community Detection

Community detection aims to find similar nodes from the network and group them into a class of clusters, which includes traditional community detection (TCD) and semi-supervised community detection (SSCD). TCD finds all possible communities in the network [8], while SSCD finds the remaining communities in the network given a small number of query communities. Different from the TCD task [11], [12], SSCD focuses on targeted mining communities with specific structures and properties rather than identifying the communities exhaustedly [6], [8], [9]. For example, some work [6] scored the nodes in the network and selected the nodes with high scores as the seeds. Some work [11] proposed a community-based method to select subgraphs rather than seed nodes. However, they all treated the seed/subgraph selection and rewriting processes as two separate processes, failing to optimise them jointly and thus achieving unsatisfactory results.

2.2 Graph Supervised Learning

Graph supervised learning (GSL) has been applied in many fields [14], [15]. Early works focus on predictive learning [15], [16]. For example, ARVGA [15] learn to predict missing edges by structural reconstruction. GraphMAE [16] uses a masking strategy and scaled cosine loss for feature reconstruction. Nowadays, more attention has been paid to contrastive-based learning. Some works concentrate on the development of graph augmentation strategies [15]–[18]. For example, SPAN [20] augments the node features matrix from a spectral perspective. Several recent studies [21]–[23] have explored negative sample-free methods. In this study, we propose a novel sGNN using an overlapping sampling strategy.

2.3 Optimal Transport

Optimal transport theory provides a method for inferring the correspondence between two distributions and has various applications in different fields. For example, [24] solves domain adaptation problems by learning the transport plan from the source domain to the target domain. [25] uses optimal transport to handle 3D shape matching and surface registration problems. Other applications include generative models [26], [27]. Some works, such as [28], have proposed an optimal transport-based method for graph-based regularised empirical distribution. In the field of community detection, we are the first to use optimal transport to model the distribution of nodes.

3. Method

3.1 Method Overview

In this section, we describe our proposed iterative CGNet model for community detection on multi-UAV collaborative networks. CGNet takes the UAV network

$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ (where $\mathcal{V}$ denotes the set of nodes, $\mathcal{E}$ is the set of edges, and $\boldsymbol{X}$ represents the node feature matrix) and query communities $\mathcal{Q} = \{Q^1, Q^2, \ldots, Q^M\}$ as inputs. It outputs the detected communities $\mathcal{D} = \{D^1, D^2, \ldots, D^N\}$. As shown in Fig. 2, the model is mainly composed of two phases: (1) candidate subgraph matching; (2) candidate subgraph rewriting. For a given query community Q^i, candidate subgraph matching aims to seek the best matching k-ego nets $\mathcal{C} = \{C^1, C^2, \ldots, C^K\}$ (a central node along with surrounding within k hops) from the network. It first gets a series of samples through a cover sampling algorithm, while node distances are then combined with graph distances to complete cross-layer comparison training. The trained sGNN embedding is then used to match candidates. Due to its fixed structure, the community detected by matcher may not be accurate, so we introduce the subgraph rewriter for further refinement. Afterwards, we optimise the matcher and rewriter iteratively by narrowing the embedding distances between the query community and generated high-quality communities (*rewriting results*) as well as corresponding candidate communities (*matching results*) with a regularised contrastive loss.

3.2 Candidate Subgraph Matching

In this subsection, we describe our candidate subgraph matching component. In detail, we develop a sGNN encoder to encode individual subgraphs and then compute the similarities between the subgraphs and query communities.

3.2.1 Self-supervised GNN

We train a self-supervised GNN to encode a graph as an embedding, capturing the similarities between graphs. We propose overlapping subgraph sampling. For a query community, we sample subgraphs with high overlapping ratio as positive samples, and sample subgraphs with low overlapping ratio as negative samples for contrastive learning. For the optimisation loss, we consider not only the global graph-level distance but also the node-level distance based on optimal transport.

(1) Overlapping Subgraph Sampling: Generally, the most basic judgement metric for two similar graphs is that they have enough overlapping nodes [27]. Specifically, to construct a sample S, we first randomly select a node n in a query community Q based on the node connectivity degree. Then, we connect the selected node to a fixed number of its neighbouring nodes (we set the fixed number as 15 via validation). For the constructed sample, we calculate overlapping ratio of selected subgraph nodes between the sample S and the query community $Q(\frac{S \cap Q}{S \cup Q})$. If the overlapping ratio is above a given threshold (set as 0.65 in this work), it is taken as a positive sample, otherwise, it is a negative sample. We continue the sample construction process until the number of positive samples as well as that of the negative samples reaches the preset number.

(2) Graph Convolution: Our sGNN utilises graph convolution to aggregate the information of neighbouring nodes in the graph [28]. The l-th layer GNN updates the node embeddings $\boldsymbol{h}$ as:

$$\boldsymbol{h}l = \mathit{GraphConv}(\mathbf{A}, \boldsymbol{h}l - 1, \mathbf{W}) \tag{1}$$

where $\boldsymbol{A}$ means the adjacent matrix, $\boldsymbol{W}$ represents the trainable parameters. $\boldsymbol{h}0 = \boldsymbol{X}$ and we use $\boldsymbol{h}l,u$ to represent node u's embedding of layer l.

(3) Optimisation Objective: With the positive and negative samples of the query communities, we train the sGNN by optimising the following objectives: global graph-level distance D_1 and node-level distance D_2 based on optimal transport. In this way, our GNN encoder can capture not only the global graph properties but also the local node properties.

Graph-level distance. We can get the graph embedding by adding a pooling layer over the nodes in the graph. Then we employ the cosine similarity to obtain the graph-level distance between the graphs as follows:

$$Z^G = \sum_{u \in G} h_{l,u} \tag{2}$$

$$D_1(Q, S) = 1 - \frac{{Z^Q}^\top Z^S}{|Z^Q|_2 |Z^S|_2} \tag{3}$$

Optimal transport based node-level distance: Optimal transport theory can be used to calculate the difference between two distributions. We use optimal transport theory [31] to compute the difference in the distribution of nodes between two graphs. In detail, we concatenate the representations of nodes (augmented feature [32], [6]) in all sGNN convolution layers as their features of node level to compute the transport cost between two graphs [33]. Formally, the minimal transport cost, also known as Wasserstein distance $\mathcal{D}_{\mathrm{W}}$, is considered as the final node-level distance

$$\begin{aligned} D_2(Q, S) &= \mathcal{D}_{\mathrm{W}}(\boldsymbol{X}_Q, \boldsymbol{X}_S) \\ &= \min_T \sum_{u \in Q} \sum_{v \in S} T_{uv} \cdot \mathit{cost}\left(\overline{h}_u, \overline{h}_v\right) \end{aligned} \tag{4}$$

where $\boldsymbol{X}_Q$ and $\boldsymbol{X}_S$, respectively, refer to the node features of query community Q and subgraph $S \cdot \boldsymbol{T}$ represents a transport matrix, where $\boldsymbol{T}_{uv}$ denotes the transportation coefficient from u to v. *cost* is the cost function that evaluates the distance between u and v , such as cosine distance. u is the concat embeddings of node u from all convolution layers

$$\overline{h}_u = [h_{0,u}| \cdots |h_{i,u}| \cdots |h_{l,u}] \tag{5}$$

Then we sum the graph-level and node-level distances, and apply the cross-level loss L_{con} for optimisation:

$$\mathrm{Dis}(Q, S) = D_1(Q, S) + \gamma D_2(Q, S) \tag{6}$$

$$L_{\mathrm{con}} = \sum_{\mathrm{Pos}} \mathrm{Dis}(Q, Q') + \sum_{\mathrm{Neg}} \max\{0, m - \mathit{Dis}(Q, Q')\} \tag{7}$$

where *Pos* denotes the set of positive samples where community Q is a query community and Q' is a sample subgraph, *Neg* denotes the negative samples, and m is the margin.

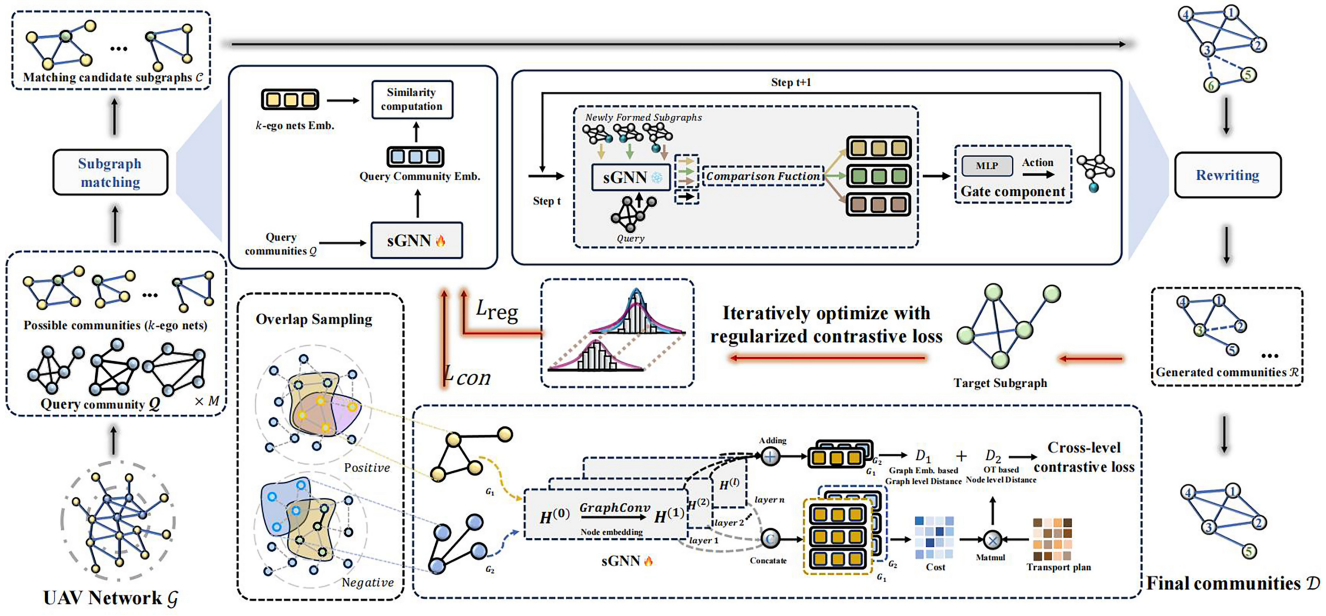

Figure 2. Overview of CGNet. Phase-1 involves selecting potentially relevant UAV communities. The query community is input, and the sGNN trained by contrastive learning is used to embed and compare different ego-nets. Phase-2 involves refining these candidate communities to improve accuracy. The gating component makes node adjustments by comparing the dynamic graph embedding. The sGNN is optimised by regular contrastive loss. Finally, the rewritten completed UAV subgraph communities are output as final results on the right side.

3.2.2 Similar Subgraph Matching

We match the query community with all the possible subgraphs from the network $\mathcal{G}$ to get the candidate matching communities $\mathcal{C} = \{C^1, C^2, \ldots, C^{M*K}\}$ based on the graph embeddings obtained by our sGNN. We consider the k-ego nets of all nodes as possible subgraphs. Subsequently, the respective embeddings of k-ego nets and query community are obtained through our sGNN. We choose K nearest neighbors for each query community $Q^i (i = [1, M])$ according to cosine similarity and get all the candidate matching communities $\mathcal{C} = \{C^1, C^2, \ldots, C^{M*K}\}$.

3.3 Subgraph Filtering

In the candidate subgraph matching, we assume a fixed subgraph structure (*i.e.*, k-ego net) for efficiency while sacrificing flexibility and accuracy. Based on the matched potential candidate subgraphs, we further refine them via candidate subgraph rewriting. Hence, a gating component is employed to automatically select nodes to rewrite the subgraph. We adjust subgraph structures by comparing the community embedding of nodes placed in the associated community against the query community and performing actions (including adding and dropping nodes) until the occurrence of a *STOP* signal.

(1) Gating Component: We consider the nodes present in the candidate community as the state s of gating component, while the nodes that can be adjusted at step t are considered as the action a of gating component, as shown in Fig. 3. At step t, the gating component chooses

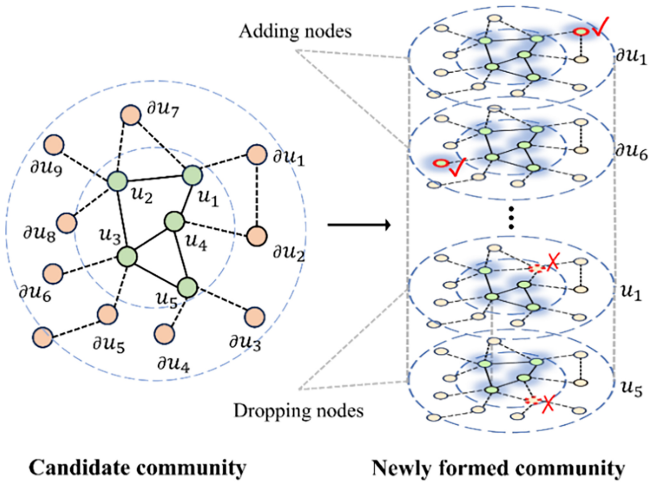

Figure 3. The overview of writing progress. Green dots indicate already existing nodes. Red nodes are the surrounding nodes immediately adjacent to them.

the proper at and transfers from st-1 to st:

$$st-1 = \mathcal{U}t-1 = \cup u \in Gt-1 u \quad (8)$$

$$at = \mathcal{U}t-1 \cup \partial\mathcal{U}t-1 = \cup u \in Gt-1 \mathcal{N}(u) nu \quad (9)$$

$\mathcal{U}t$-1 refers to the nodes currently in the subgraph, $\partial\mathcal{U}t$-1 refers to the nodes not currently in the subgraph but connected to intra-nodes of subgraph Gt-1 , which may be added to Gt-1 in the next step, and $\mathcal{N}$ (u) indicates the set of neighbouring nodes surrounding node u.

(2) Dynamic Comparison of Possible Nodes: Considering that the query community is our main optimisation

target for rewriting, we use the comparison information between the target query community and the newly-formed subgraph as the comparison factor. Specifically, we adopt the query community with the closest graph embedding distance (as described in Section 3.2) to the candidate matching community as the target query community Q. Subsequently, we build possible newly-formed subgraphs G' by adding or dropping nodes u if u belongs to $\mathcal{U}$ or $\partial\mathcal{U}$. Then, we compare the embeddings of the newly-formed subgraphs with query community Q by:

$$V_{G'=}F_c\left(Z^{G'}, Z^{Q}\right) \tag{10}$$

where $Fc(x, y) = W\gamma[x - y, x\Theta y]$ denotes a comparison function measuring the embedding closeness and relevance. $W\gamma$ is a transformation matrix. Θ is Hadamard product. $V_{G'}$ is the obtained comparison vectors, serving as the environment factors.

The comparison vector is then fed into the gating component. An MLP (Multilayer Perceptron) network together with a Softmax function will be applied for scoring the newly-formed subgraphs. The node u corresponding to the newly-formed subgraph with the highest score is selected as the action for the current step is

$$a_t^r = Softmax\left(\mathrm{MLP}_{W_\alpha}\left(V_{G'}\right)\right) \tag{11}$$

Moreover, we add a special $STOP$ node to the action set. When the $STOP$ node is selected, rewriting progress is immediately stopped and we get the generated community.

After reaching the pre-set maximum steps l or $STOP$ node, the gating component will obtain a rewritten community together with the corresponding rewriting trajectory. After each step of writing, we update the MLP parameters W_α as

$$W_{\alpha'} = W_\alpha + lr\sum_{t=0}^{T}\nabla W_\alpha \log\left(\alpha_t^r\right)\cdot r \tag{12}$$

where lr stands for the learning rate and ∇ refers to the gradient. r is the difference of each writing

$$r = F1score(G', Q)—F1score(G, Q) \tag{13}$$

where Q denotes the corresponding query community.

Finally, by rewriting all candidate subgraphs, we will generate a set of refined communities $\mathcal{R} = \{R1, R2, \ldots, RN\}$.

3.4 Iterative Optimisation

We propose a regularised contrastive loss that minimises the embedding distances between the query community Q and the generated high-quality communities R as well as corresponding candidate communities C. Specifically, referring to Student T-distribution, the distance between the generated graph R^i and any query community Q^j is converted to probability $r_{ij}(\sum_j r_{ij} = 1)$. The loss is computed as follows:

$$r_{\mathrm{ij}} = \frac{\left(1 + \left\|Z^{R^i} - Z^{Q^j}\right\|^2\right)^{-\frac{1}{2}}}{\sum_{j'}\left(1 + \left\|Z^{R^i} - Z^{Q^j}\right\|^2\right)^{-\frac{1}{2}}} \tag{14}$$

$$p_{\mathrm{ij}} = \frac{\frac{r_{\mathrm{ij}}^2}{\sum_i r_{\mathrm{ij}}}}{\sum_{j'}\frac{r_{ij'}^2}{\sum_i r_{ij'}}} \tag{15}$$

$$L_{R\to Q} = KL(P||R) = \sum_i\sum_j p_{\mathrm{ij}}\log\frac{p_{\mathrm{ij}}}{r_{\mathrm{ij}}} \tag{16}$$

where Z is the graph embedding obtained by pooling over node embeddings learned via sGNN and p_{ij} refers to high-confidence assignments. Similarly, the regularisation loss $L_{R\to Q}$ between graph C^i and Q^j can be calculated. The overall regularised contrastive loss function is :

$$L_{\mathrm{reg}} = L_{R\to Q} + L_{C\to Q} \tag{17}$$

With the regularised contrastive loss, we further optimise the sGNN. This allows the generated results of the rewriter and the matching results of the matcher more compatible with the query communities. The optimised sGNN is then re-used in the matching and rewriting processes. Through I iterations, we can finally get the unified optimal results $\mathcal{D}$ (*i.e.*, constructed communities at the I-th iteration) with a well-trained sGNN.

Table 1
Statistics of Datasets. From Left to Right: Nodes Number, Edges Number, Nodes Number of the Largest Community, Average Nodes Number of All Communities

	$\#N$	$\#E$	C_{Max}	C_{Avg}
Amazon	6926	17893	30	9.38
DBLP	37020	149501	16	8.37
LiveJournal	69860	911179	30	13.00
YouTube	216544	1393206	25	7.70
Facebook	1826	34964	33	11.20
Twitter	46930	784135	42	9.76

4. Experiments

In this section, we conduct experiments on different network datasets to validate our model's ability to mine local node subgraphs for construction. The detailed data information is presented in Table 1.

4.1 Experiment Setup

4.1.1 Self-supervised GNN

We conduct experiments on six public network datasets, including LiveJournal, Amazon, DBLP, YouTube, Twitter, and Facebook. We download them from the public

Table 2
Experimental Results, Bold Indicates Optimal Results. The Underline Is Sub-Optimal. CGNet Represents Our Original Version. CGNet-A Is an Enhanced Version Where We Have Improved the Calculation of Node Embedding Distances by Incorporating Additional Data Features, Such as Node Degree

Method	Amazon			DBLP			Live journal			YouTube		
	F1	Jaccard	ONMI	F1	Jaccard	ONMI	F1	Jaccard	ONMI	F1	Jaccard	ONMI
BigClam	0.6657	0.5637	0.0543	0.3314	0.2036	0.0764	0.3738	0.2984	0.1437	0.0929	0.0568	0.0720
ComGAN	0.6683	0.5650	0.0712	0.3352	0.2071	0.0413	0.3760	0.2913	0.0025	0.1201	0.0851	<1e-4
vGraph	0.6704	0.5604	0.0269	0.1080	0.0754	0.0138	0.0411	0.0354	<1e-4	0.1432	0.0872	<1e-4
RaidB	0.6657	0.5637	0.5435	0.3314	0.2036	0.0764	0.3738	0.2984	0.2437	0.1307	0.1022	0.0351
DSCPCD	0.6245	0.5568	0.5586	0.2692	0.1845	0.1087	0.3277	0.2229	0.1759	0.2868	0.1160	0.0690
Bespoke	0.6778	0.6229	0.6303	0.3492	0.2943	0.1515	0.3762	0.3123	0.2868	0.2745	0.1723	0.0786
Seal	0.6638	0.6265	0.6268	0.3492	0.0367	0.0006	0.3762	0.3587	0.3425	0.2091	0.1924	0.0934
SLSS	0.7263	0.6608	0.6466	0.3512	0.2734	0.1575	0.4146	0.3345	0.2697	0.3073	0.1860	0.0854
Clare	0.7541	0.6614	0.6782	0.3949	0.3012	0.2277	0.4587	0.3658	0.3216	0.3098	0.2122	0.0936
GraphLLM	0.7020	0.6372	0.6176	0.1404	0.1358	0.0952	0.2031	0.1145	0.1159	0.1698	0.1347	<1e-4
CGNet	**0.7714**	**0.6829**	**0.7102**	**0.4259**	**0.3275**	**0.2566**	**0.4926**	**0.3973**	**0.3616**	0.2945	0.2203	0.0901
CGNet-a	0.7493	0.6426	0.7076	0.4086	0.3131	0.2430	0.4554	0.3624	0.3533	**0.3145**	**0.2480**	**0.0936**

datasets website SNAP for real communities. We excluded communities whose size exceeded the 90th percentile. We extract community subgraphs that contain only the nodes in the community and their outer boundaries. This is mainly because the downloaded dataset nodes are not all labelled. Based on the literature [8], we randomly selected 1,000 communities from the full population. This is mainly a trade-off between most baseline methods and larger network sizes.

4.1.2 Self-supervised GNN

We compare our model with both TCD methods and SSCD methods, including (1) BigClam [33], (2) ComGAN [42], (3) vGraph [5], (4) RaidB [13], (5) DSCPCD [11], (6) Bespoke [7], (7) Seal [6], (8) SLSS [8], (9) Clare [9] and (10) GraphLLM [36].

Methods 1–5 are TCD baselines while Methods 6–9 are state-of-the-art SSCD baselines. Method 10 is notable for its ability to infer processing graphs from a small number of samples.

4.1.3 Evaluation Metrics

We use the most commonly used metrics $F1$ score, Jaccard score, and ONMI to evaluate the model performance [38]. Given M constructed communities R^i and N ground truth communities Q^j, we compute the performance score as follows:

$$\frac{1}{2}\left(\frac{1}{M}\sum_i \max_j \delta\left(R^i, Q^j\right) + \frac{1}{N}\sum_j \max_i \delta(R^i, Q^j)\right) \quad (18)$$

where δ can be the $F1$, Jaccard [38] and ONMI [39] functions as described in the technical content.

4.2 Performance Comparison

Table 2 and Table 3 show the outcomes of our model on popular datasets as well as the large scale graph. As we can see, our proposed CGNet generally achieves the best performance in terms of all metrics. The reason could be that these traditional approaches aim at clustering all nodes in a network rather than detecting a specific type of community. GraphLLM performs slightly inferiorly due to its limited understanding of graph structures. Moreover, it can be observed that CGNet consistently outperforms all the SSCD models by a large margin, which shows the effectiveness of our proposed method. We believe the reasons are two-fold.

1) Our proposed sGNN with contrastive loss including graph-level distance and optimal transport distance that can capture the graph properties at both graph level and node level.
2) In our gating component, the actions are decided by constantly comparing the newly-formed graph with the query community.

4.3 Semi-supervised Detection

Recall that the primary objective of this study is to detect communities of interest given a limited number of query community samples. We evaluate the detection performance of our model across various scales of query communities. As shown in Fig. 4, the performance improves consistently across all four datasets as the number of input

Table 3
Experimental Results on Large Scale Graph

Method	Facebook			Twitter		
	F1	Jaccard	ONMI	F1	Jaccard	ONMI
BigClam	0.1404	0.1097	0.0464	0.0299	0.0431	0.0264
ComGAN	0.0446	0.0139	0.0466	0.0120	0.0540	0.0233
vGraph	0.1241	0.0152	<1e-4	<1e-4	<1e-4	0.0403
RaidB	0.1435	0.1063	0.0550	0.1496	0.1019	0.0563
DSCPCD	0.1455	0.1081	0.0574	0.1370	0.1057	0.0776
Bespoke	0.1503	0.1385	0.0664	0.1587	0.1330	0.0885
Seal	0.1581	0.1403	0.1083	0.1701	0.1633	0.1080
SLSS	0.1593	0.142	0.1256	0.1705	0.1623	0.1107
Clare	0.1746	0.1477	0.1370	0.2577	0.2009	0.1163
GraphLLM	0.0837	0.1307	<1e-4	0.1532	0.0913	<1e-4
CGNet	**0.1961**	**0.1538**	**0.1771**	0.2603	0.1998	0.1187
CGNet-a	0.1957	0.1534	0.1726	**0.2709**	**0.2102**	**0.1259**

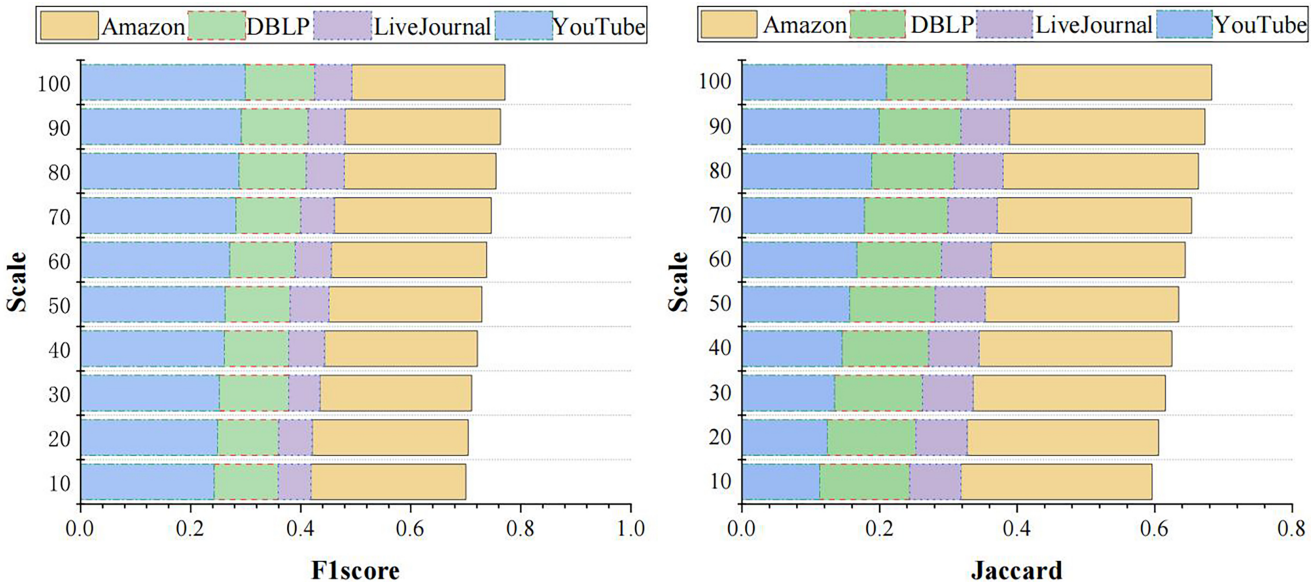

Figure 4. Effect of different number of query communities. Wider means better performance.

query communities increases. Notably, with only 10 query communities, the model achieves results comparable to what many models require with full input sets (Table 4). It indicates that our model is capable of learning sufficiently rich information from a small number of community structures through self-supervised learning.

4.4 Discussion

We conduct experiments about GNN considering the importance of GNN in our model.

4.4.1 Effect of Different GNN Convolution Methods

We test different graph neural network convolution types to achieve optimal results, including GCN, GIN, Sage, and GAT [15], [33], [9], [30]. To be fair, we keep the number of network layers uniform at 2. As shown in Table 4, our algorithm achieves high performance under most graph convolutional networks. GAT achieves the best performance in all cases. Among them, the optimal results are consistently achieved on Amazon and DBLP, which consist of fewer nodes and are suitable for the computation of GAT under the attentional mechanism computation, however, the computation of GAT is more

Table 4
Performance with Different Convolution Types

Type	Amazon			DBLP			Livejournal		
	F1	Jaccard	ONMI	F1	Jaccard	ONMI	F1	Jaccard	ONMI
GCN	0.7424	0.65	0.6703	0.3869	0.2902	0.2173	0.4888	0.3925	0.3611
GIN	0.7514	0.6565	0.6721	0.3867	0.2877	0.2133	**0.4949**	0.3966	0.3614
SAGE	0.7442	0.6514	0.6688	0.3807	0.2857	0.2018	0.4795	0.3835	0.3411
GAT	**0.7714**	**0.6829**	**0.7102**	**0.4259**	**0.3275**	**0.2566**	0.4926	**0.3973**	**0.3616**

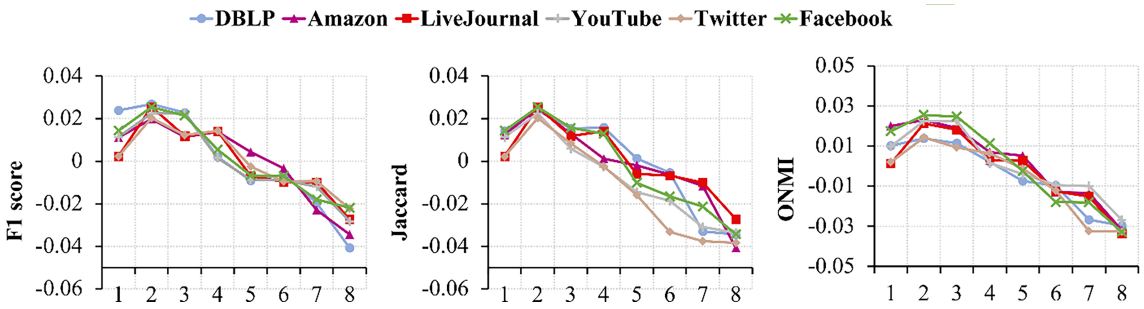

Figure 5. Effect of different GNN layer numbers.

Table 5
Ablation Study of Modules on $F1$ Score

Method	Amazon	DBLP	Livejournal	YouTube	Twitter	Facebook
CGNet	**0.7714**	**0.4259**	**0.4926**	**0.3145**	**0.1961**	**0.2709**
- w/o Comparing	0.6344	0.3115	0.318	0.2931	0.1487	0.2347
- w/o OT.Dis	0.6047	0.303	0.3154	0.2477	0.1184	0.1257
- w/o Graph.Dis	0.7092	0.3187	0.3221	0.2964	0.1243	0.2140
- w/o sGNN	0.5232	0.2526	0.2782	0.2432	0.1031	0.1175

complicated under the mega node data of Livejournal, so the final performance is affected. Considering the node size of UAV network, GAT can be reasonably adapted to the performance calculation under medium to large scale. So we finally adopt GAT as the main method.

4.4.2 Effect of Different GNN Layer Numbers

Following existing methods [43]–[45], we examine the influence of GNN layer numbers on performance. The comparison with different choices of layer number is depicted in Fig. 5. The trend on different datasets is similar. Therefore, we set 2 for all datasets during experiments.

4.5 Efficiency Study

We evaluate the efficiency of CGNet by directly comparing its inference time with that of all baselines, as existing works [46]–[49]. In the evaluation, all parameters of the baselines were set according to their original papers. Fig. 6 shows the performance ($F1$ score) and the runtime (in seconds). CGNet also takes only about 1,000 s to complete most of the tests. It also outperforms other faster models in terms of performance ($F1$ score).

4.6 Ablation Study

In this subsection, we conduct ablation experiments to study the effectiveness of each module in CGNet as existing research set [50]–[53]. As shown in Table 5, removing iterative optimisation between matching and rewriting (w/o Iteration) leads to a slight performance drop. Removing the environment factor of comparison with query communities (w/o Comparing) will largely impair the performance. If we remove the whole rewriting model and only keep the candidate subgraph matching module (*i.e.*, w/o Rewriting), we can also obtain excellent performance. While removing optimal transport distance (*i.e.*, w/o Rewriting & Optimal Tran.) and graph-level distance loss (*i.e.*, w/o Rewriting & Graph-level Dis.), the model performance decreases by around 3.1% to 10.7% on $F1$ score. Removing both graph level distance and optimal

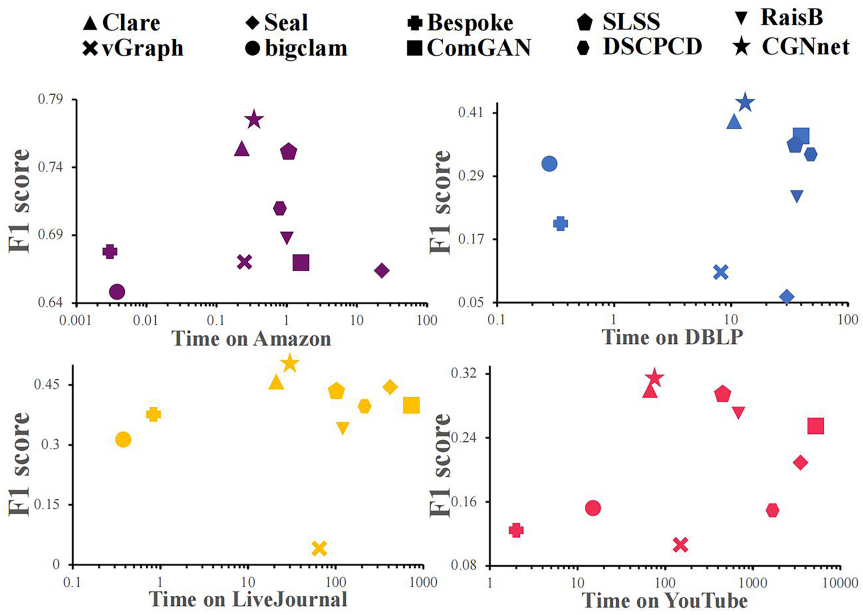

Figure 6. Efficiency comparison with all baselines. The upper left is optimal.

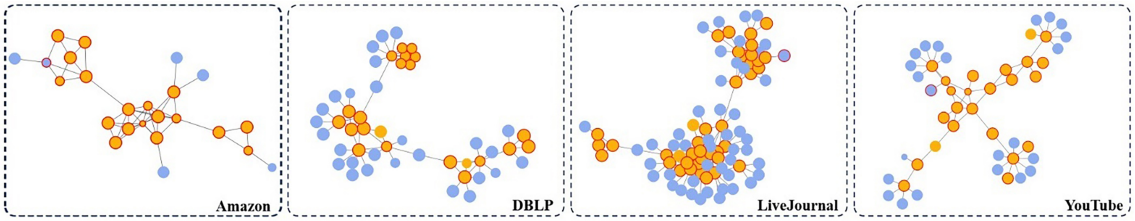

Figure 7. Case study. Yellow nodes indicate detected communities. Blue are uninterested nodes. Red represents the ground-truth.

transport distance, namely without optimizing sGNN (w/o sGNN) will lead to substantial performance drops (around 10-30%). It demonstrates the importance of sGNN for graph encoding in SSCD.

4.7 Case Study

We conduct a case study to evaluate the performance of our model following existing paper [54]–[57] As shown in Fig. 7, the model detects communities from real networks. On the Amazon dataset, the model successfully identifies the community nodes of interest from the surrounding unrelated nodes. For more complex networks such as DBLP, LiveJournal, and YouTube, the model accurately detects nodes that are close to the ground-truth communities. Even though there are unrelated nodes interspersed in the complex network, our model can still accurately detect them. This robustness can be attributed to the model's precise community matching and rewriting capabilities. It also demonstrates the model's scheduling ability in the face of complex real-time environments.

5. Conclusion

This study addresses the key challenge of community construction in dynamic UAV networks by proposing a semi-supervised detection method based on iterative cross-layer graph contrastive learning. By integrating graph-level topological features with node-level collaboration relationships, and combining dynamic contrastive loss functions with an incremental optimisation mechanism, experiments have shown that this method achieves a community detection accuracy of 77% on large-scale network datasets (an improvement of 8% compared to traditional graph embedding methods), while reducing network reconstruction latency to 1.2 s per hundred nodes. These breakthroughs provide a new paradigm for real-time decision-making in critical scenarios for UAV fleets, such as military reconnaissance and precision agriculture. The semi-supervised mechanism (requiring less than 10% labelled data) significantly enhances engineering applicability. Future research will explore distributed community optimisation under a federated learning framework to meet the collaborative control needs of ultra-large-scale heterogeneous UAV swarms.

Acknowledgement

This research was funded by the National Natural Science Foundation of China Projects NO. 62206267 and NO.62303439. This work is also supported by Grant No. 2023KJC-Y-0192.

References

[1] Z. Guoqi, B. Yu, L. Chunlei, C. Kai, and Y. Huanyin, Multitask assignment of swarming UAVs based on improved PSO, *International Journal of Robotics and Automation*, *36*(3), 2021, 188.

[2] H. T. Do, H. T. Hua, M. T. Nguyen, C. V. Nguyen, H. T. T. Nguyen, H. T. Nguyen, and N. T. Nguyen, Formation control algorithms for multiple-UAVs: A comprehensive survey, *EAI Endorsed Transactions on Industrial Networks and Intelligent Systems*, *8*(27), 2021, 170230.

[3] X. Ma, W. Dong, and B. Li, Comprehensive fault-tolerant control of leader–follower unmanned aerial vehicles (UAVs) formation, *International Journal of Robotics and Automation*, *34*(6), 2019, 195.

[4] Y. Yan, Z. Lv, J. Yuan, and S. Zhang, Obstacle avoidance for multi-UAV system with optimised artificial potential field algorithm, *International Journal of Robotics and Automation*, *36*, 2021, 7.

[5] A. Bakshi, S. Parthasarathy, and K. Srinivasan, Semi-supervised community detection using structure and size, in *Proceeding of IEEE International Conference on Data Mining*, 2018, 869–874.

[6] Y. Zhang, Y. Xiong, Y. Ye, T. Liu, W. Wang, Y. Zhu, and P. S. Yu, SEAL: Learning heuristics for community detection with generative adversarial networks, in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, 1103–1113.

[7] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang, Self-supervised learning: generative or contrastive, *IEEE Transactions on Knowledge and Data Engineering*, *35(*1), 2021, 857–876.

[8] L Ni, J Ge, Y Zhang, W Luo, and V. S. Sheng, Semi-supervised local community detection, *IEEE Transactions on Knowledge and Data Engineering*, *36,* 2033, 1–17.

[9] X. Wu, Y. Xiong, Y. Zhang, Y. Jiao, C. Shan, Y. Sun, Y. Zhu, and P. S. Yu, CLARE: A semi-supervised community detection algorithm. in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, 2059–2069.

[10] S. Min, C. Rhim, and S. Chang, An ANN-based intergrated model for autonomous UAV flight control considering external forces, *International Journal of Robotics and Automation*, *39*(5), 2024, 362–378.

[11] Y. Wang, J. Cao, Z. Bu, J. Wu, and Y. Wang, Dual structural consistency preserving community detection on social networks, *IEEE Transactions on Knowledge and Data Engineering*, *35*, 2023, 11301–11315.

[12] H. Roghani and A. Bouyer, A fast local balanced label diffusion algorithm for community detection in social networks, *IEEE Transactions on Knowledge and Data Engineering*, *35*, 2023, 5472–5484.

[13] P. G. Sun, X. Wu, Y. Quan, and Q. Miao, Rearranging 'indivisible' blocks for community detection, *IEEE Transactions on Knowledge and Data Engineering*, *35*, 2023, 6252–6263.

[14] G. A Bilodeau and R. Bergevin, Matching graphs with fuzzy attributes in machine vision, *International Journal of Robotics and Automation*, *20*(1), 2005, 50-59.

[15] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, Adversarially regularised graph autoencoder for graph embedding, in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, 2018, 2609–2615.

[16] Z. Hou, X. Liu, Y. Cen, Y. Dong, H. Yang, C. Wang, and J. Tang, GraphMAE: Self-supervised masked graph autoencoders, in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, 594–604.

[17] K. Hassani and A. H. Khasahmadi, Contrastive multi-view representation learning on graphs, in *Proceeding of International Conference on Machine Learning*, 2020, 4116–4126.

[18] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, Deep graph contrastive representation learning, 2020, *arXiv:2006.04131.*

[19] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, Graph contrastive learning with adaptive augmentation, in *Proceedings of the Web Conference,* 2021, 2069–2080.

[20] L. Lin, J. Chen, and H. Wang, Spectral augmentation for self-supervised learning on graphs. in *Proceeding of Eleventh International Conference on Learning Representations*, 2023, 1–27.

[21] S. Thakoor, C. Tallec, M. G. Azar, M. Azabou, E. L. Dyer, R. Munos, P. Velickovic, and M. Valko, Large-scale representation learning on graphs via bootstrapping, in *Proceeding of Tenth International Conference on Learning Representations*, 2022, 1–21.

[22] H. Zhang, Q. Wu, J. Yan, D. Wipf, and P. S. Yu, From canonical correlation analysis to self-supervised graph neural networks, in *Proceeding of Advances in Neural Information Processing Systems*, 2021, 76–89.

[23] N. Lee, J. Lee, and C. Park, Augmentation-free self-supervised learning on graphs, in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, 7372–7380.

[24] P. Huang, M. Xu, J. Zhu, L. Shi, F. Fang, and D. Zhao, Curriculum reinforcement learning using optimal transport via gradual domain adaptation, in *Proceeding of Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems (NeurIPS 2022)*, 2022, 1–15.

[25] L. Xu, H. Sun, and Y. Liu , Learning with batch-wise optimal transport loss for 3D shape recognition, in *Proceeding of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, 3328–3337.

[26] X. Gu, L. Yang, J. Sun, and Z. Xu, Optimal transport-guided conditional score-based diffusion model, in *Proceeding of Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems (NeurIPS 2023)*, 2023, 1–13.

[27] T. Adrai, G. Ohayon, M. Elad, and T. Michaeli, Deep optimal transport: A practical algorithm for photo-realistic image restoration, in *Proceeding of Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems (NeurIPS 2023)*, 2023, 1–15.

[28] J. Tang, W. Zhang, J. Li, K. Zhao, F. Tsung, and J. Li, Robust attributed graph alignment via joint structure learning and optimal transport, in *Proceeding of 39th IEEE International Conference on Data Engineering (ICDE 2023)*, Anaheim, CA, 2023, 1638–1651.

[29] Z. Lou, J. You, C. Wen, A. Canedo, and J. Leskovec, Neural subgraph matching, 2020, *arXiv:2007.03092.*

[30] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, How powerful are graph neural networks? in *Proceeding of 7th International Conference on Learning Representations (ICLR 2019)*, New Orleans, LA, 2019, 1–17.

[31] J. Zhang, W. Zhong, and P. Ma, A review on modern computational optimal transport methods with applications in biomedical research, 2020, *arXiv:2008.02995.*

[32] C. Cai and Y. Wang, A simple yet effective baseline for non-attributed graph classification, 2018, *arXiv:1811.03508.*

[33] L. Chen, Z. Gan, Y. Cheng, L. Li, L. Carin, and J. Liu, Graph optimal transport for cross-domain alignment, in *Proceedings of the 37th International Conference on Machine Learning*, 2020, 1542–1553.

[34] O. Shchur and S. Günnemann, Overlapping community detection with graph neural networks, 2019, *arXiv:1909.12201.*

[35] Z. Chen, H. Mao, H. Li, W. Jin, H. Wen, X. Wei, S. Wang, D. Yin, W. Fan, H. Liu, and J. Tang, Exploring the potential of large language models (LLMs) in learning on graphs, *SIGKDD Explorations*, *25*(2), 2023, 42–61.

[36] Q. Cappart, D. Chételat, E. B. Khalil, A. Lodi, C. Morris, and P. Velickovic, Combinatorial optimisation and reasoning with graph neural networks, in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI)*, 2021, 4348–4355.

[37] T. Chakraborty, A. Dalmia, A. Mukherjee, and N. Ganguly, Metrics for community analysis: A survey, *ACM Computing Surveys*, *4*, 2017, 1–37.

[38] A. F. McDaid, D. Greene, and N. Hurley, Normalised mutual information to evaluate overlapping community finding algorithms, 2017, *arXiv:1110.2515.*

[39] T. N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, 2016, *arXiv:1609.02907*.

[40] W. Hamilton, Z. Ying, and J. Leskovec, Inductive representation learning on large graphs, in *Proceeding of Advances in Neural Information Processing Systems*, 2017, 1–11.

[41] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, Graph attention networks, 2017, *arXiv:1710.10903*.

[42] Y. Jia, Q. Zhang, W. Zhang, and X. Wang, CommunityGAN: Community detection with generative adversarial nets, in *Proceeding of the World Wide Web Conference*, 2019, 784–794.

[43] H. R. Patel and V. A. Shah, Shadowed type-2 fuzzy sets in dynamic parameter adaption in cuckoo search and flower pollination algorithms for optimal design of fuzzy fault-tolerant controllers, *Mathematical and Computational Applications*, *27*(6), 2022, 89.

[44H. R. Patel and V. A. Shah, A metaheuristic approach for interval type-2 fuzzy fractional order fault-tolerant controller for a class of uncertain nonlinear system, *Automatika: Časopis za Automatiku, Mjerenje, Elektroniku, Računarstvo I Komunikacije*, *63*(4), 2022, 656–675.

[44] H. R. Patel and V. A. Shah, Type-2 fuzzy logic applications designed for active parameter adaptation in metaheuristic algorithm for fuzzy fault-tolerant controller, *International Journal of Intelligent Computing and Cybernetics*, *16*(2), 2022, 198–222.

[45] H. R. Patel, Fuzzy-based metaheuristic algorithm for optimization of fuzzy controller: fault-tolerant control application, *International Journal of Intelligent Computing and Cybernetics*, *15*(4), 2022, 599–624.

[46] H. R. Patel and V. A. Shah, Stable fuzzy controllers via LMI approach for non-linear systems described by type-2 T–S fuzzy model, *International Journal of Intelligent Computing and Cybernetics*, *14*(3), 2021, 509–531.

[47] H. R. Patel and V. A. Shah, Application of metaheuristic algorithms in interval type-2 fractional order fuzzy TID controller for nonlinear level control process under actuator and system component faults, *International Journal of Intelligent Computing and Cybernetics*, *14*(1), 2021, 33–53.

[48] H. R. Patel, S. K. Raval, and V. A. Shah, A novel design of optimal intelligent fuzzy TID controller employing GA for nonlinear level control problem subject to actuator and system component fault, *International Journal of Intelligent Computing and Cybernetics*, *14*(1), 2021, 17–32.

[49] B. Bede, M. Ceberio, M. De Cock, and V. Kreinovich, *Fuzzy Information Processing*. (Cham: Springer, 2022).

[50] H. R. Patel and V. A. Shah, Comparative analysis between two fuzzy variants of harmonic search algorithm: Fuzzy fault tolerant control application, *IFAC-PapersOnLine*, *55*(7), 2022, 507–512.

[51] H. R. Patel and V. A. Shah, General type-2 fuzzy logic systems using shadowed sets: a new paradigm towards fault-tolerant control, in *Proceeding of Australian & New Zealand Control Conference (ANZCC)*, 2021, 116–121.

[52] H. R. Patel and V. A. Shah, Fuzzy logic based metaheuristic algorithm for optimisation of type-1 fuzzy controller: Fault-tolerant control for nonlinear system with actuator fault, *IFAC-PapersOnLine*, *55*(1), 2022, 715–721.

[53] S. Raval, H. R. Patel, V. Shah, U. C. Rathore, and P. P. Kotak, Fault-tolerant control using optimised neurons in feed-forward backpropagation neural network-for MIMO uncertain system: A metaheuristic approach, in *Proceeding of International Conference on Intelligent and Fuzzy Systems*, 2023, 597–609.

[54] H. R. Patel and V. A. Shah, Decentralised stable and robust fault-tolerant PI plus fuzzy control of MIMO systems: a quadruple tank case study, *International Journal on Smart Sensing and Intelligent Systems*, *12*(1), 2019, 1.

[55] H. R. Patel, Metaheuristic optimisation algorithm for optimal design of type-2 fuzzy controller, *International Journal of Applied Evolutionary Computation (IJAEC)*, *13*(1), 2022, 1–15.

[56] H. R. Patel, Optimal intelligent fuzzy TID controller for an uncertain level process with actuator and system faults: population-based metaheuristic approach, *Franklin Open*, *4*, 2023, 100038.

[57] H. R. Patel and V. A. Shah, Simulation and comparison between fuzzy harmonic search and differential evolution algorithm: Type-2 fuzzy approach, *IFAC-PapersOnLine*, *55*(16), 2022, 412–417.

Biographies

Haonan Liu received the B.Sc. degree from Xidian University, Xi'an, China, in 2021. He is currently pursuing the Ph.D. degree with the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing, China. His research interests include graph neural network and natural language processing.

Xiaoyu Li received the B.E. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2016, and the M.E. degree from Beijing University of Posts and Telecommunications in 2019. He is currently a Research Assistant Fellow with the Aerospace Information Innovation Institute, Chinese Academy of Science, Beijing, China. His research interests include data mining, information extraction, event logic graph and natural language processing.

Li Jin received the B.S. degree from Xidian University, Xi'an, China, in 2012 and the Ph.D. degree from the Department of Computer Science and Technology, Tsinghua University, Beijing, China, in 2017. He is currently an Associate Professor with the Aerospace Information Research Institute, Chinese Academy of Sciences. His research interests include machine learning, knowledge graphs and geographic information processing.

Wen Shi received the B.Sc. and M.Sc. degrees from Beijing Jiaotong University, China, in 2013 and 2016, respectively, and the Ph.D. degree from Beijing Jiaotong University, China, in 2020. He is currently with the Aerospace Information Research Institute, Chinese Academy of Sciences as a Research Associate. His interests include artificial intelligence and virtual reality.

Yang Bai received the B.Sc. degree in surveying and prospecting technology and engineering and the Ph.D. degree in geodetection and information technology from China University of Geosciences (Beijing), Beijing, China, in 2018 and 2024, respectively. He is currently a Research Assistant with the Aerospace Information Research Institute, CAS. His research interests include remote sensing image interpretation and multimodal visual reasoning.

Linhao Zhang received the B.S. degree from Xidian University, Xi'an, China, in 2020, and the Ph.D. degree from Aerospace Information Research Institute, Chinese Academy of Sciences, in 2025. His research interests include affective computing and multimodal learning.

Hongqi Wang received the B.Sc. degree from the Changchun University of Science and Technology, Changchun, China, in 1983, the M.Sc. degree from the Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun, in 1988, and the Ph.D. degree from the Institute of Electronics, Chinese Academy of Sciences, Beijing, China, in 1994. He is currently a Professor with the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing. His research interests include computer vision and remote sensing image understanding.