Neural Processing Letters (2021) 53:4073–4096
https://doi.org/10.1007/s11063-021-10583-x

Identification of Influential Nodes for Drone Swarm Based on Graph Neural Networks

Qiang Wang[1] · Dongye Zhuang[1] · Haibin Xie[1]

Accepted: 5 July 2021 / Published online: 22 July 2021

Abstract
Accurately tracking the influential nodes in swarm is the key to improve the control or defense against the drone combat. But the identification of influential nodes is intractable due to hardly capturing the pattern information of the running drones based on the traditional Graph Theory. To tackle this issue, in this paper, we propose a novel framework to identify the influential nodes, which includes Latent Interaction Graph Extraction (LIGE), Degree Centrality based Influential Inference (DCII) and Formation Stability Coefficient based Re-ranking (FSCR). LIGE is employed to reconstruct the latent interaction graph from the dynamic trajectory data by exploiting the encoder in Neural Relational Inference. DCII invokes the traditional graph model and measures the significance of nodes. In the re-ranking module FSCR, we introduce a novel metric, the formation stability coefficient, to measure swarm formation stability and re-ranking the candidates. The experimental results on the Drone swarm datasets demonstrate that our framework can effectively identify the influential nodes of drone swarm. The extracting time in LIGE is relatively long, which makes it difficult to recognize the core node within the limit time. By optimizing the encoder structure, the extraction time can be reduced to at least twice the original.

Keywords Drone swarm · Influential nodes · Pattern information · Neural relational inference · Formation stability coefficient

1 Introduction

In recent years, the continuous improvement of the intelligence level of drone swarm [1,2] has brought great challenges and difficulties to the threat analysis from the low-level discrete data(containing the location and velocity information) obtained by the radar, laser, etc. Inferring the pattern information between running drones and mining semantic information from low-level discrete data, such as the identification of influential nodes in drone swarm, contribute to understand the combat formation and combat intent of drone swarm [3].

✉ Dongye Zhuang
zhuangdongye@nudt.edu.cn

[1] College of Intelligence Science and Technology, National University of Defense Technology, Changsha 410073, China

Drone swarm system structure [4–6] is a distributed architecture based on local perception or communication. When the swarm formation composed of multiple running drones is performing a mission, they need to fly in accordance with the set formation based on the control method. During the past ten years, the main control methods include the behavior-based decentralized control [7], the virtual structure control [8], the artificial potential field [9] and the Leader-Follower [10,11]. These methods can take full advantages of the information-rich drone guidance to fulfill swarm control. In the prior art, the drone embracing rich information is directly regarded as the leader or core node [12,13]. Besides, the core node acts a pivotal part and interacts with the other ones in drone swarm. However, these methods can not deal with the issue of how to identify who is the leader or the core node without direct communication between drones. Recognizing the core node can significantly boost the performance of locating and tracking it within the limited time. Once the core node is removed, the adaptive capacity and robustness of the entire swarm will be significantly reduced [14]. Therefore, identifying the core node is crucial to tackle the issue of defense against the drone swarm combat [15] on a basis of an understanding of the combat formation and combat intent.

Drone swarm can be treated as interacting system networks [16] on account of the similar behavior and dynamics. For each interacting system, its dynamical model can be embodied by a graph [17]. The identification of influential nodes is intractable in the previous applications due to the problem of capturing the pattern information in the graph. To tackle this issue, Graph Theory is generally considered as the modeling tool to collect and describe the pattern information of running drones. Several measure methods based on traditional Graph Theory have been put forward to identify the influential nodes in graph networks [18–21] including Degree Centrality (DC), Closeness Centrality (CC), Betweenness Centrality (BC), and Eigenvector Centrality (EC). These methods are applicable to identify the influential nodes in the graph networks when the graph structure is known and static. In this paper, we consider the circumstance that the graph structure and the pattern information between drones are unknown. The traditional Graph Theory cannot handle the dynamic and uncertain flying trajectory data. For the sake of solving the problem of capturing the pattern information of the running drones, state-of-the-art methods are mostly based on the Graph Neural Networks (GNNs) to extract the latent interaction graph, which describes the pattern information of running drones.

1.1 Graph Neural Networks (GNNs)

The scheme we focus on is to exploit the Graph Neural Networks (GNNs) to extract the latent interaction graph from the dynamic flying trajectory data. GNNs [22–24] are a class of Neural Network model that can process the dynamic graph structure data. The architecture of GNNs [25] is constructed from the graph $G = (V, E)$ with a set of nodes V and edges E. The approaches to deal with the task of the latent interaction graph extraction are closely associated with the methods of graph generation [26].

Kipf et al. [27] propose the Variable Graph Auto-Encoder (VGAE) model. The VGAE is a probabilistic latent variable model for graph generation, which can date back to the random graph model [28] with fixed probability edges. In addition, the VGAE can tackle the problem of edge probabilities by employing the method of the neural network-based model. The VGAE model is limited as a conditional auto-encoder with the encoder and decoder conditioned on the node feature. The input of VGAE model is the eigenmatrix $\mathbf{X}$ of the node and the adjacency matrix $\mathbf{A}$ of the graph. VGAE uses a Graph Convolutional Networks to receive a set of node features (the $\mathbf{X}$ and $\mathbf{A}$) and output a set of updated node representations.

In addition, VGAE utilizes the decoder to predict the A to generate the graph from the a set of updated node representations.

Apart from the VGAE, two other graph generation models based on Deep Neural Networks have been proposed. Johnson et al. [29] put forward a graph generation model that obtains intermediate node representations and can predict one or more nodes and edges in each step of the recursive generation process. Gomez-Bombarelli et al. [30] introduce a graph generation model for molecular based on the Variable Auto-Encoder, exploiting the recursive neural networks to operate on sequence-based molecule representations.

The above methods implement promising performance on the graph generation with prior knowledge of the adjacency matrix $\mathbf{A}$. In this paper, we consider the circumstance that the adjacency matrix $\mathbf{A}$ is unknown. Therefore, we exploit the encoder in Neural Relational Inference (NRI) [31] to extract the latent interaction graph, which is an unsupervised graph generation approach with no prior knowledge of the $\mathbf{A}$. Compared to other methods, the most advantage is the encoder can directly infer the edge types between node pairs from the dynamic flying trajectory data. The edge types represent the pattern information between running drones.

1.2 Influential Nodes Re-ranking of Drone Swarm

We employ two methods to rank the influential nodes of drone swarm. Invoking the traditional graph model and referring to the DC [18] to measure the significance of the nodes, we can recognize the core node and rank the influential ones of drone swarm by using the node degree. In addition, the node degree is the number of other nodes directly connected to the node on the latent interaction graph. We define the core node with the maximum node degree on the graph. The method only considers the influence of the node itself on the latent interaction graph without considering the nodes' influence on the global structure of graph. When the degree of two or more nodes is the same, the approach cannot rank them effectively.

To consider the global structure of graph, we introduce the Formation Stability Coefficient(FSC) as a metric of swarm formation stability to re-rank the candidates, inspired by the method of measuring the significance of nodes in Graph Theory [32]. In Graph Theory, the node's significance refers to its influence on the graph networks when the one is deleted from the graph networks [33]. In other words, the node's significance can be regarded as its propagation or bridging capability. The node with smaller value of FSC means the node plays more significant role in drone swarm and gives more disturbance to formation stability of drone swarm when the one is deleted. Based on the analysis above, we define the core node is the one with the minimum FSC value on the whole graph networks.

In the influential nodes re-ranking, we delete one node from the latent interaction graph successively and employ the method of dynamics reconstruction to predict the trajectory data on the basis of the latent interaction graph after deleting the node. In addition, we calculate and normalize the trajectory similarity between the original trajectory and the prediction by using the Hausdorff distance [34] as the deleted node's FSC. Finally, we re-rank the candidates by using the node's FSC, considering the nodes' influence on the swarm formation stability of running drones.

1.3 Identifying the Influential Nodes of Drone Swarm

Figure 1 depicts the architecture for identifying the influential nodes of drone swarm. The architecture is categorized into three modules, Latent Interaction Graph Extraction (LIGE),

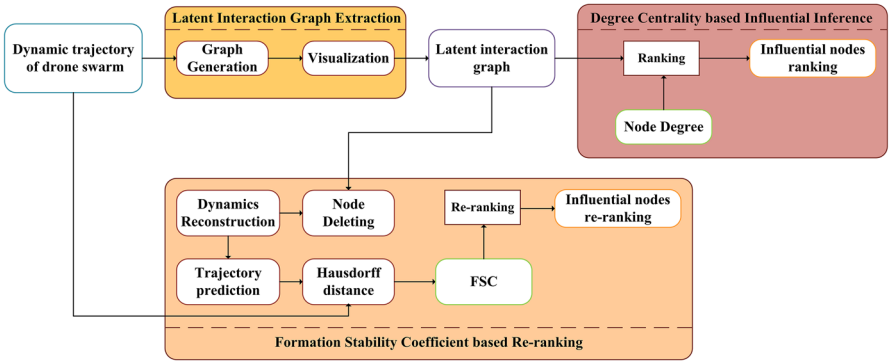

Fig. 1 Architecture for identifying the influential nodes for drone swarm. In the Latent Interaction Graph Extraction, we employ the method of graph generation to capture the pattern information and extract the latent interaction graph between drones from the dynamic trajectory data. In the Degree Centrality based Influential Inference, we invoke the traditional graph model and use the node degree to rank the influential nodes. In the Formation Stability Coefficient based Re-ranking, the FSC as a novel metric of swarm formation stability is introduced to re-rank the candidates

Degree Centrality based Influential Inference (DCII) and Formation Stability Coefficient based Re-ranking (FSCR). In the LIGE, we employ the method of graph generation to capture the pattern information between drones from the dynamic trajectory data, which consists of an array of location and velocity data of drones. In addition, the latent interaction graph can be extracted by visualizing the pattern information.

In the DCII, we invoke the traditional graph model [35] and use the node degree to rank the influential nodes. The node with maximum node degree can be recognized as the core node. In the FSCR, the FSC as a novel metric of swarm formation stability is introduced to re-rank the candidates and the node with minimum FSC can be recognized as the core node.

1.4 Overview of Our Contributions

In summary, we resolve the issue of capturing the pattern information by exploiting the encoder in NRI to reconstruct the latent interaction graph from the dynamic flying trajectory data. In addition, we employ two methods to rank the influential nodes of drone swarm. The influential nodes can be ranked by using the node degree. Moreover, the FSC as a novel metric of swarm formation stability is introduced to re-rank the candidates. A series of experiments demonstrate that our framework can effectively identify the influential nodes of drone swarm and is robust when the noise exists.

To be more specific, the main contributions of our work are as follows.

(i) We propose the framework to identify the influential nodes in drone swarm without prior knowledge of the physical principle.
(ii) We introduce the FSC to measure the swarm formation stability , which indicates the nodes' influence on the global structure of graph.
(iii) We optimize the network structure by utilizing the skip connections and adjusting the number of hidden layer units for the Multi-Layer Perceptron(MLP)in the encoder, reducing the time consumption by at least half.

The remainder of the paper is organized as following. In Sect. 2, we review the related works about the identification of influential nodes for graph networks, edge prediction for

interacting systems networks based on traditional methods and Graph Neural Networks (GNNs), the trajectory similarity matching. In Sect. 3, we introduce the FSC as a swarm formation stability to re-rank the influential nodes. In Sect. 4, we demonstrate that our framework can effectively recognize the core node and identify the influential nodes in drone swarm. The main conclusions are covered in Sect. 5.

2 Related Works

In this section, we review the works about influential nodes recognition in graph networks. Besides, we also review the approaches about the link (edges) prediction in graph networks since the link prediction can help to infer the latent interaction graph. In addition, the approaches of trajectory similarity matching are briefly reviewed in that they can calculate the node's global influence on the graph structure.

2.1 Identification of the Influential Nodes in Graph Neural Networks

Traditional approaches [18–21] to recognize the influential nodes in graph networks include Degree Centrality(DC), Closeness Centrality(CC), Betweenness Centrality(BC), and Eigenvector Centrality(EC). DC [18] utilizes the node degree to measure the one' influence on the undirected graph networks. It only considers the influence of the node itself on graph networks without considering the influence of the one on the graph' global structure. As to CC [19], the core node is geometrically located in the center of the graph. BC [20] calculates all the shortest paths of any two nodes in the graph networks and makes up for the above problems. Unfortunately, it is not suitable for situations where many nodes do not pass other node pairs'shortest distance. The basic idea of EC [21] is that the centrality of a node is a function of the centrality of adjacent nodes. The method is not suitable for sparse large graph networks. Besides, PageRank [36] and LeaderRank [37] are the variants of EC. They are the evaluation indicators that measure the significance of the nodes in directed networks. Their ideas are based on the same probability of randomly jumping from one node to other nodes.

Soheila et al. [38] propose an entropy-based approach to recognize the influential nodes in heterogeneous networks. The spreading power of the method is the strongest, but the method has a large amount of calculation with large time consumption. Jin et al. [39] propose a method based on the local neighbor contribution (LNC). LNC combines the influence of the node itself with the contribution of the nearest neighbor and the second nearest neighbor. Its noticeable advantage is that its time complexity is considerably low. Wen et al. [40] propose a new fuzzy local dimension to rank the influential nodes in complex networks. The method utilizes fuzzy sets to consider the influence of the distance from the core node on the local dimension of the core node to change the influence ability, only concentrating on individuals in complex networks, not the entire global structure. Tang et al. [41] propose an improved discrete particle swarm optimization algorithm and an impact maximization strategy based on enhanced network topology. The algorithm has lower time complexity and better robustness. Zareie et al. [42] propose two new approaches for influential nodes ranking. These two methods exploit Shannon entropy [43] and Jensen-Shannon Divergence [44]with higher precision and validity than others.

2.2 Edge (Link) Prediction

Previous studies [45–49] are based on the similarity, probability and maximum likelihood methods to predict the link of graph. Similarity-based methods assume that the nodes tend to form links with other similar nodes through a given distance function to determine whether two nodes are similar. The definition of node similarity is an extraordinary challenge. The purpose of the probability model is to extract the underlying structure from the observed network and use the learned model to predict the missing links. It is difficult to optimize the established the objective function. Maximum likelihood-based methods obtain detailed rules and specific parameters by maximizing the likelihood of the observed structure. The probability of any unobserved link can be calculated based on these rules and parameters. The maximum likelihood method is very time-consuming from the perspective of practical application.

With the development of GNNs, Kipf et al. [50] propose the Graph Convolutional Networks. After that, velivckovic et al. [51] propose the Graph Attention Network and schlichtkrull et al. [52] propose the Relational Graph Convolutional Network. These models have the ability to tackle the issue of the graph analysis tasks. Graph analysis tasks [53] can be roughly abstracted into the following four categories: node classification [54,55], link prediction [31], clustering [56,57] and visualization [58]. The approaches used to deal with the issue of extracting the latent interaction graph for drone swarm are closely related to the methods of link prediction.

Ferreira et al. [59] proposed the Time-delay Added Evolutionary, which is a new method for time series prediction. The method performs an evolutionary search for the minimum number of dimensions necessary to represent the underlying information that generates the time series. Paasen et al. [60] phrase time series prediction as a regression issue and applied the dissimilarity or kernel-based regression techniques, such as 1-nearest neighbor, kernel regression and Gaussian process regression to predict the future state of the whole graph. In addition, the simple regression approaches are sufficient to capture the dynamics in the theoretical models. Oneto et al. [61] propose a new family of kernels for graphs which exploits an abstract representation of the information inspired by the multi-layer perception architecture. The method not only extracts the potential of the most advanced graphical node kernel, but also develops a multi-layer architecture through a series of stacked kernel pre-image estimators. The method can better deal with the data coming from structured domains.

2.3 Trajectory Similarity Matching

The approaches for calculating the similarity of two trajectories can be divided into the following two categories, point based distance and shape based distance [62–64].

Point based distance approaches include Euclidean Distance(ED), Dynamic Time Warping(DTW), etc. Danielsson et al. [65] propose the ED to represent the true distance between two points in space. The computational time is linear and the trajectories must be the same length. Berndt et al. [66] propose the DTW by exploiting the dynamic programming to calculate time warping to better measure the similarity of two time series with different lengths. The approach can provide a better match than ED.

Shape based distance approaches include Hausdorff distance, Frechet distance, etc. Huttenlocher et al. [34] propose the Hausdorff distance, which is a measure to calculate the similarity between two sets of points. When the image is polluted by noise occlusion, the original Hausdorff distance is easy to cause mismatching. Alt et al. [67] propose the Frechet

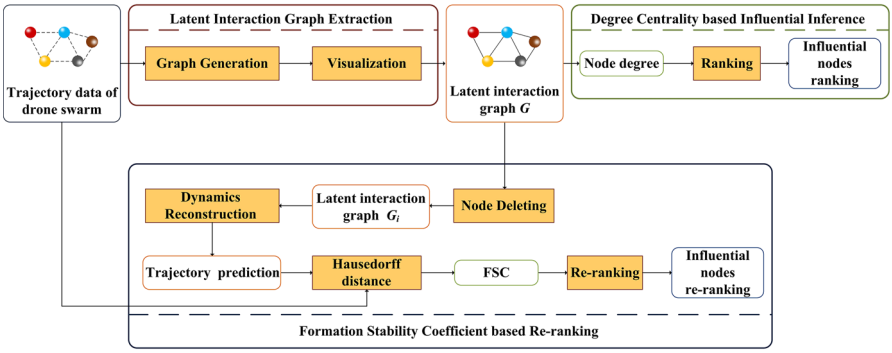

Fig. 2 Framework to identify the influential nodes of drone swarm. The framework is categorized into three modules, Latent Interaction Graph Extraction (LIGE), Degree Centrality based Influential Inference (DCII) and Formation Stability Coefficient based Re-ranking (FSCR)

distance, which is an algorithm for calculating the distance between two curves, which is used to judge the similarity between the two curves.

With the development of neural networks, Liao et al. [68] propose multiple relative descriptors of trajectories to model the relative information of pairs of trajectories. To be more specific, the relative motion and location descriptors are exploited to extract the relative motion information and location information. Wangflorence et al. [69] propose a data navigation approach for identifying the shape similarity by utilizing the enhanced Self Organizing Maps. The approach overcomes the limitation of not being able to show detailed local distance information.

3 Method

3.1 Framework to Identify the Influential Nodes of Drone Swarm

Figure 2 depicts the framework of identifying the influential nodes for drone swarm. Compared with other methods, the innovation is that our framework directly extracts the latent interaction graph from the dynamic flying trajectory data. The framework is categorized into three modules, Latent Interaction Graph Extraction (LIGE), Degree Centrality based Influential Inference (DCII) and Formation Stability Coefficient based Re-ranking (FSCR).

We reconstruct the latent interaction graph from the dynamic flying trajectory data by exploiting the method of graph generation in the LIGE. Then, we utilize two approaches to rank the influential nodes. In the ranking module (DCII), the influential nodes are ranked according the nodes' degree. Moreover, in the re-ranking module (FSCR), the FSC is introduced to re-rank the candidates, considering the nodes' influence on the formation stability of drone swarm.

3.2 Latent Interaction Graph Extraction

Figure 3 presents the module for the Latent Interaction Graph Extraction. We extract the pattern information by exploiting the method of the graph generation. The core model of graph generation is the encoder, which is a probabilistic latent variable inference model.

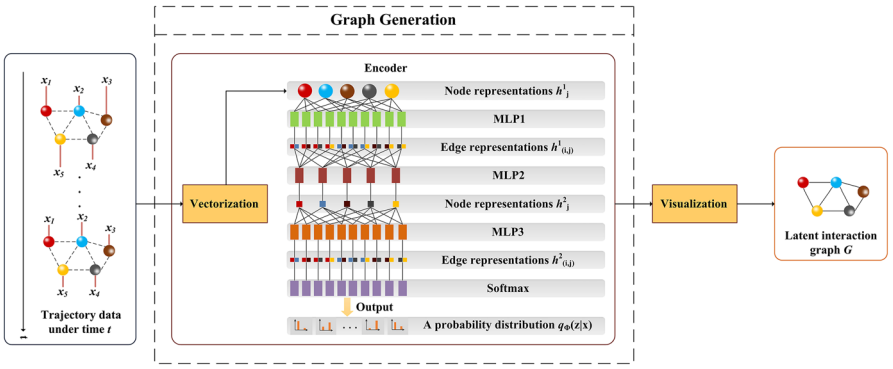

Fig. 3 Illustration of Latent Interaction Graph Extraction module. Graph generation is exploited to extract the pattern information. The core module of graph generation is the encoder, which capture the pattern information between nodes. The input of the encoder is the dynamic trajectory data of drone swarm. The output is the probability distribution $q_\phi(z_{(i,j)}|\boldsymbol{x})$. $z_{(i,j)}$ is a discrete categorical variable, representing the pattern information between the node $\boldsymbol{x_i}$ and the node $\boldsymbol{x_j}$

In addition, the encoder can infer the edge types, which represent the pattern information between nodes in drone swarm. The input of the encoder is the dynamic trajectory data. The output is the probability distribution $q_\phi(z_{(i,j)}|\boldsymbol{x})$. $z_{(i,j)}$ is a discrete categorical variable, representing the pattern information between the node $\boldsymbol{x_i}$ and the node $\boldsymbol{x_j}$.

The process of the encoder is shown in Fig. 3. There are two message passing operations, Node to Edge and Edge to Node representations. The Multi-Layer Perceptron (MLP) is used to achieve these two message passing operations. The encoder vectorizes the trajectory data of each node as the node representations $\boldsymbol{h^1_j}$. The edge representations $\boldsymbol{h^1_{(i,j)}}$ can be obtained from the neighboring node representations $\boldsymbol{h^1_j}$ through using MLP with two or three layers. Since the edge representations $\boldsymbol{h^1_{(i,j)}}$ only depends on $\boldsymbol{h^1_i}$ and $\boldsymbol{h^1_j}$ without considering the interactions with other nodes, the node representations $\boldsymbol{h^2_j}$ can be obtained from the edge representations $\boldsymbol{h^1_{(i,j)}}$ by the MLP using information from the whole graph. The edge representations $\boldsymbol{h^2_{(i,j)}}$ can be updated based on the node representations $\boldsymbol{h^2_j}$. Finally, the encoder is modeled as $q_\phi(z_{(i,j)}|\boldsymbol{x})$ by utilizing the softmax function to capture the pattern information.

3.3 Degree Centrality Based Influential Inference

In the ranking module, we invoke the traditional graph model and measure the significance of nodes referring to DC. We rank the influential nodes by the node degree. In addition, the node degree is the number of other nodes directly connected to the node on the graph. Figure 4 shows the ranking process in the Degree Centrality based Influential Inference. The input of module is the latent interaction graph G. The output is the result of influential nodes ranking by the node degree.

3.4 Formation Stability Coefficient Based Re-ranking

To consider the nodes' influence on the global structure of the graph, we introduce the FSC to calculate the nodes' influence on formation stability of drone swarm.

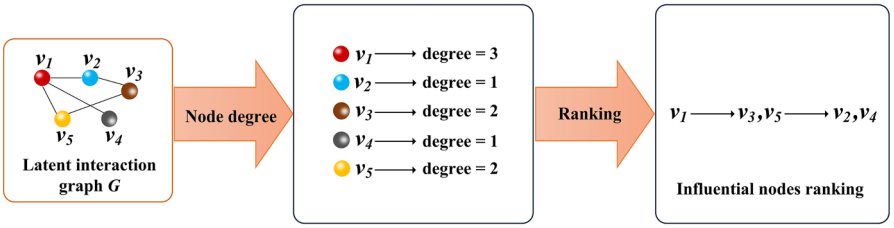

Fig. 4 Ranking process of Degree Centrality based Influential Inference. The input of module is the latent interaction graph G. The output is the result of influential nodes ranking by the node degree

We present the re-ranking module of Formation Stability Coefficient based Re-ranking in Fig. 5. We delete one node from the latent interaction graph successively and employ the method of dynamics reconstruction to predict the trajectory data based on the latent interaction graph after deleting the nodes. Besides, we calculate and normalize the trajectory similarity value between the original dynamic trajectory data and the trajectory prediction data by using the Hausdorff distance [34] as the deleted node's FSC. Finally, we re-rank the candidates by using the nodes' FSC, considering the node's influence on the formation stability of drone swarm.

The core model of dynamic reconstruction is the decoder, which is exploited to learn the dynamical model and predict the trajectory data. The process of the decoder is depicted in Fig. 5. The input of the decoder is the latent interaction graph and the trajectory data at the current moment. There are two message passing operations, Node to Edge and Edge to Node representations. The MLP is adopted to achieve these two message passing operations.

The decoder vectorizes the trajectory data at the current moment as the node representations $\boldsymbol{h}_j^1$. The edge representations $\boldsymbol{h}_{(i,j)}^1$ can be obtained from neighboring node representations $\boldsymbol{h}_j^1$ by using MLP with two or three layers. Finally, the node representations $\boldsymbol{h}_j^2$ can be obtained from the edge representations $\boldsymbol{h}_{(i,j)}^1$ by the MLP. Additionally, the edge representations $\boldsymbol{h}_j^2$ denotes the state difference between at the next and current moment.

Hausdorff Distance is a measure to calculate the trajectory similarity between two sets of points, which is a definition form of the distance between two sets of points. Assuming that there are two sets of sets $A = \{a_1, \ldots, a_p\}$, $B = \{b_1, \ldots, b_q\}$, then the Hausdorff distance between these sets of points is defined as:

$$H(A, B) = \max(h(A, B), h(B, A)), \tag{1}$$

with

$$h(A, B) = \max(a \in A) \min(b \in B)||a - b||, \tag{2}$$

and

$$h(B, A) = \max(b \in B) \min(a \in A)||b - a||. \tag{3}$$

(1) is called bi-directional Hausdorff distance. $||\cdot||$ is norm distance between point sets A and B, such as L2 or Euclidean distance. $h(A, B)$ and $h(B, A)$ in (2)(3) are called the unidirectional Hausdorff distance from set A to set B and set B to set A, respectively. $h(A, B)$ actually sorts the distance $||a_i - b_j||$ from each point a_i in the point set A to the point b_j in the set B closest to the point a_i and takes the maximum value in the distance as the value of $h(A, B)$. The same is true for $h(B, A)$. According to (1), the the bi-directional Hausdorff

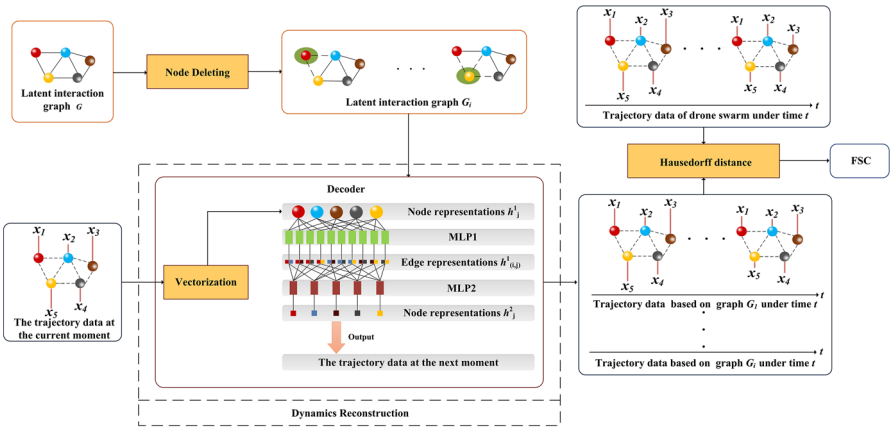

Fig. 5 Illustration of Formation Stability Coefficient based Re-ranking module. Dynamic Reconstruction is exploited to predict the trajectory data based on the latent interaction graph after deleting. The core module of Dynamic Reconstruction is the decoder. The input of the decoder is the latent interaction graph and the trajectory data at the current moment. The output is the trajectory prediction data

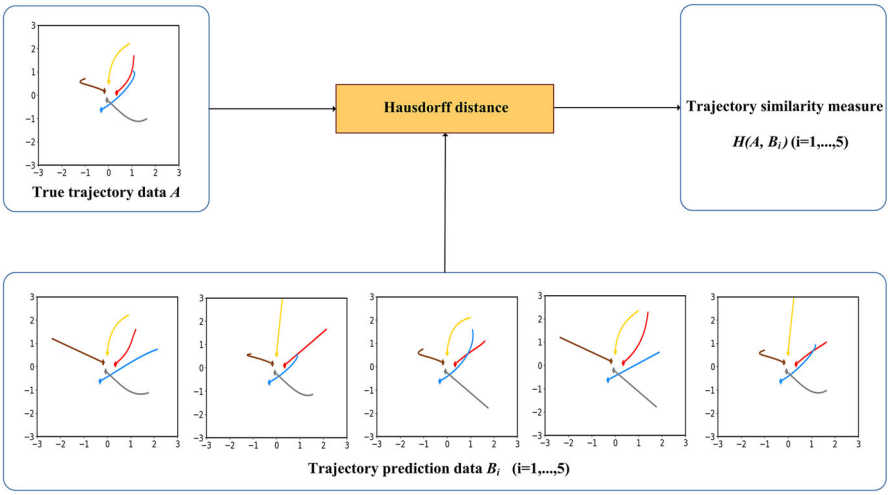

Fig. 6 Process of the Hausdorff distance $H(A, B_i)$ between true trajectory data A and trajectory prediction data B_i. B_i represents the trajectory data predicted based on the latent interaction graph after deleting the i-th node

distance $H(A, B)$ represents the greater of the unidirectional $h(A, B)$ and $h(B, A)$, which measures the maximum degree of mismatch between two sets of points.

Referring to the Hausdorff distance, we introduce the FSC as metric of swarm formation stability, considering the nodes' influence on the global structure of the graph. The definition of λ_i as the FSC parameter is the following:

$$\lambda_i = 1 - \frac{H(A, B_i)}{\sum_{i=1}^{N} H(A, B_i)} \tag{4}$$

$H(A, B_i)$ represents the Hausdorff distance value between the true trajectory data and the trajectory data predicted based on the latent interaction graph after deleting the i-th node. λ_i calculates the nodes' influence on the formation stability of drone swarm. The smaller value of λ_i means the node plays more significant role in drone swarm and gives more disturbance to formation stability of drone swarm when the one is deleted from the latent interaction graph.

Figure 6 depicts the process of the trajectory similarity measure $H(A, B_i)$ between true trajectory data A and trajectory prediction data B_i. B_i represents the trajectory data predicted based on the latent interaction graph after deleting the i-th node.

4 Experiments

4.1 Datasets

Drone Swarm datasets are generated from the public available Springs / Charged particles datasets[1] based on the mapping table in our previous work [70]. The initial location and velocity of each node in each set of trajectory data are randomly generated. In addition, the ground truth for the latent interaction graph in each set of trajectory data is also randomly generated, which represents the type edges between nodes. According to the latent interaction graph and the initial trajectory data, the trajectory data at the time series is generated by the simulation.[2]

A series of experiments [31] have been experimented to demonstrate that the encoder and decoder in NRI can greatly extract the latent interaction graph and predict the future state in the public Spring/Charged particles datasets. The public Spring / Charged particles datasets have a large scale and rich latent interaction graph categories. Therefore, the Drone swarm datasets generated from the Spring/Charged particles datasets can support us to extract the latent interaction graph and predict the trajectory for drone swarm. Examples of the trajectory and the ground truth for the latent interaction graph of drone swarm with 5 nodes are shown in Fig. 7. For the drone swarm with 5 nodes, the degree of the core nodes ranges from 1 to 4. In addition, the number of core nodes is not unique. It is possible that there are two or more core nodes in drone swarm.

For the Springs interacting system. the nodes are modeled by Hooke's law $F_H = -k(r_i - r_j)$. One edge type represents that there is the pattern information between nodes, and the other edge type represents that there is no the pattern information between nodes. In addition, the pattern information is the Hooke's force F_H. However, in the Charged particles interacting system, the edge types between nodes is different from the edge types in the Springs interacting system. The nodes in the Charged particles interacting system are modeled by Coulomb's law $F_C = Cq_iq_j\frac{r_i - r_j}{|r_i - r_j|^3}$. In addition, both types of edges indicate that there is the pattern information between nodes. One edge type indicates that the pattern information is the attraction force F_C, and the other edge type indicates that the pattern information is the repulsive force F_C. For our drone swarm interacting system, its edge types are the same as the Springs' edge types. Besides, our drone swarm is modeled by a rule-law of universal gravitation $F_U = Gm_im_j\frac{r_i - r_j}{|r_i - r_j|^3}$.

[1] The public Springs / Charged particles datasets can be download at https://github.com/ethanfetaya/NRI.

[2] The code for the simulation process is available at https://github.com/ethanfetaya/NRI.

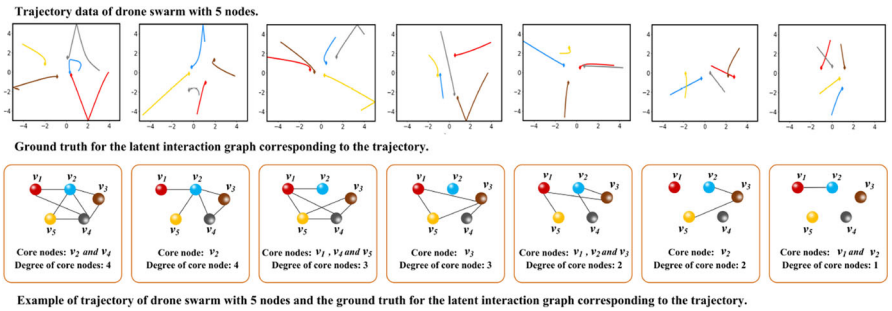

Fig. 7 Examples of the trajectory and the ground truth for the latent interaction graph of drone swarm with 5 nodes in drone swarm datasets. The initial location and velocity of each node in each set of trajectory data in time series are randomly generated. Besides, the ground truth for the latent interaction graph in each set of trajectory data is randomly generated

Table 1 Drone Swarm Datesets used for the identification of influential nodes

Datasets	Number	Length	Time steps	Edge types
Training set	50,000	5000	49	2
Validation set	10,000	5000	49	2
Test set	10,000	10,000	98	2

Table 1 represents the details for the Drone swarm datasets used for the identification of influential nodes. The number of training sets generated by the simulation is 50k. The number of validation sets generate by the simulation is 10k. The number of test sets generated by the simulation is 10k. The length of trajectory for the training and validation sets is 5k. And the length of trajectory for the testing sets is 10k. The number of nodes in the simulation is 5. The sampling frequency is 100. The time steps of the training and validation sets is 49. And the time steps of the testing sets is 98.

Generation of Drone Swarm Datasets with Noise Considering the impact of noise on the drone trajectory data, we generate the Drone swarm datasets with noise. The process is summarized as following:

Step 1 Randomly generate each node's location and velocity disturbances in each set of original trajectory data:

Location of the noise: $noise\_loc = numpy.random.randn(timesteps, 2, n\_balls)$

Velocity of the noise: $noise\_vel = numpy.random.randn(timesteps, 2, n\_balls)$

$timesteps$ means the number of trajectory data(containing the location and velocility) and n_balls presents the number of the nodes in the drone swarm.

Step 2 Set the disturbance parameter σ to 0.01, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1.

Step 3 Add the product of the nodes' location and velocity disturbances and the disturbance parameter σ to the each set of original trajectory data in order to generate the drone swarm datasets with noise.

Location with the noise: $loc+ = noise\_loc * \sigma$

Velocity with the noise: $vel+ = noise\_vel * \sigma$

Examples of trajectory data in drone swarm with 5 nodes interfered by the noise with different disturbance parameters σ are shown in Fig. 8. As we observe from Fig. 8, the disturbance of noise on the original trajectory data is not obvious with the disturbance parameter

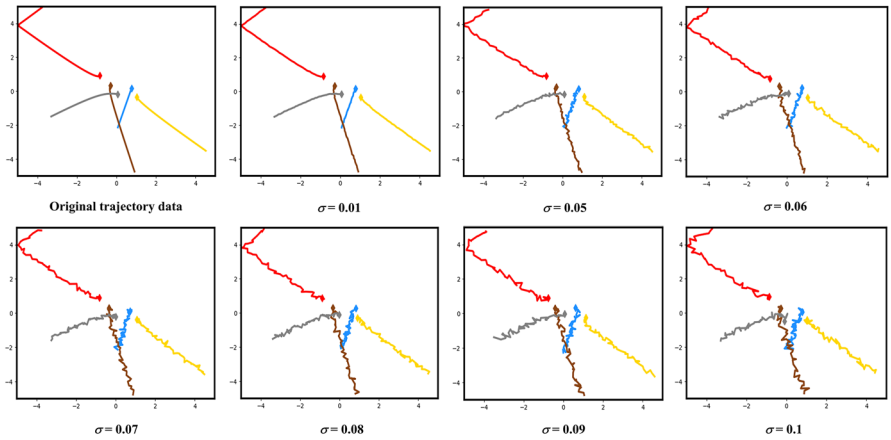

Fig. 8 Examples of trajectory data in drone swarm with 5 nodes interfered by the noise with different disturbance parameter σ. Randomly generate each node's location and velocity disturbances in each set of original trajectory data. Set the disturbance parameter σ to 0.01, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1

$\sigma \leq 0.05$ whereas the disturbance of noise on the original trajectory data is relatively large with the disturbance parameter $\sigma \geq 0.06$.

4.2 Experimental Setting

Learning All the experiments are performed on Pytorch 1.4 on a RTX 2080 Ti Ubuntu 18.04 machine. The learning process for our framework to extract the latent interaction graph is as follows:

Step 1 Based on training sets x, we compute the probability distribution $q_\phi(\boldsymbol{z_{(i,j)}}|\boldsymbol{x})$ for edge types by running the encoder. $\boldsymbol{z_{(i,j)}}$ represents the pattern information between $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$.

Step 2 Sample $\boldsymbol{z_{(i,j)}}$ from the probability distribution $q_\phi(\boldsymbol{z_{(i,j)}}|\boldsymbol{x})$ and run the decoder to compute the expectation μ^2, ..., μ^t.

Step 3 The loss function is divided into two parts, the reconstruction error $-\sum_j \sum_{t=2}^{T} \frac{||x_j^t - \mu_j^t||}{2\sigma^2}$ and KL divergence $\sum_{i \neq j} H(q_\phi(\boldsymbol{z_{(i,j)}}|\boldsymbol{x}))$.

Step 4 Calculate the gradients through back propagation and optimization in order to minimize the loss function. The Adam optimizer is exploited as the optimization with a initial learning rate of 0.0005 and the decay factor of 0.5. In addition, the learning rate is decayed by the decay factor of 0.5 every 200 epochs.

Step 5 Update the gradients for 500 epochs and learn with a batch size of 128. Calculate the accuracy of the edge prediction in the latent interaction graph by the following algorithm. In addition, the latent interaction graph can be extracted by visualizing the results of edge prediction.

To verity the effeciveness and efficiency of our framework, we evaluate our framework's accuracy for the edge prediction in the latent interaction graph by conducting the experiments on the public available / Charged particles datasets and Drone swarm datasets with 5 nodes. In addition, we also design the experiments on three datasets with 10 and 15 nodes respectively, considering the impact of the number of nodes on the prediction accuracy. Besides, we also

discuss the influence of computational time of extracting the latent interaction graph and noise on our framework for identifying the influential nodes.

Algorithm 1 The algorithm for the edge prediction accuracy in the latent interaction graph.

Input: The test set trajectory data x, the number of test set T
the ground truth of the edge for test set $Edge\_truth$
the number of nodes in the test set N
Output: The result of edge prediction accuracy in the latent interaction graph acc_pre
$timesteps = 49$
Encoder input data: $data\_encoder = x[:, :, : timesteps, :]$
The result of edge prediction: $edges =$ **Encoder**$(data\_encoder)$
$counter = 0, acc_ave = 0$
$N\_edge = N * (N - 1)$
for i *in* **range**$(1 : T)$ **do**
  for j *in* **range**$(1 : N_{edge})$ **do**
    if $(edges[i][j] == Edge\_true[i][j])$ **then**
      $counter+ = 1$
    end if
    $acc[i] = (counter/N\_edge)$
  end for
  $acc\_pre = acc\_pre + acc$
end for
$acc\_pre = acc\_pre/T * \%$

Edge prediction in the latent interaction graph. Algorithm 1 shows the whole process of the algorithm for the accuracy of the unsupervised edge prediction in the latent interaction graph by the LIGE in our framework. In order to better explain and denote our algorithm, the learned encoder is represented by **Encoder**. The input is trajectory data x in test sets, the number of test sets T, the ground truth of the edge for test set $Edge\_true$ and the number of nodes N in drone swarm. The output is the edge prediction accuracy in the latent interaction graph.

We firstly set the value of the timesteps to 49. In other words, the input data of the encoder is the trajectory data containing 49 timesteps. Then, we get the result of edge prediction in the latent interaction graph by the learning encoder, **Encoder**. Finally, we calculate the accuracy of the edge prediction for each set in test sets, the average accuracy of the entire test sets is calculated as the final accuracy.

Formation Stability Coefficient based Re-ranking (FSCR) The FSC as a metric of swarm formation stability to re-rank the influential nodes.

The algorithm for the **FSCR** is summarized below:

Step 1 Obtain the new latent interaction graph G_i by deleting one node from the latent interaction graph G successively. We utilize the adjacency matrix A to represent the type of edges in the latent interaction graph G. In addition, exploting the A_i represents the type of edges in the latent interaction graph G_i. The relationship between A_i and A is as follows:

$A[i, :] = 0$
$A[:, i] = 0$
$A_i = A$

Step 2 On the basis of the new latent interaction graph G_i(or A_i)and the initial trajectory data(containing the 49 timesteps), the trajectory prediction data can be predicted by the learned decoder.

Table 2 Accuracy (in %) of unsupervised edge prediction in the latent interaction graph for three datasets with different node number

Model	Number of objects		
	5	10	15
Springs	99.94	98.40	80.85
Charged	76.25	59.88	50.24
Drone swarm	92.28	78.25	71.20

Step 3 Calculate the FSC through utilizing the Hausdorff distance to measure the trajectory similarity between the true trajectory data and the trajectory prediction data.

Step 4 According to the nodes' FSC in the latent interaction graph to re-rank the candidates. In addition, the core node is recognized with minimum value of FSC.

4.3 Results and Discussion

We verity the efficiency and effectiveness of our framework according to three important factors: edge prediction accuracy, the robustness for the noise and computational time of extracting the latent interaction graph.

Edge Prediction Accuracy Table 2 summarizes the results for the unsupervised edge prediction in the latent interaction graph by the LIGE in our framework. Figure 9 shows the results of latent interaction extraction for there models with 5 nodes. LIGE shows the highest accuracy in the Springs model when the three interacting systems have the same number of nodes. In contrast, in the Charged interacting systems, LIGE presents the worst accuracy. In addition, in three interacting systems, the accuracy of LIGE for the latent interaction graph extraction will decrease as the number of nodes increases. Besides, we believe that LIGE is effective when the accuracy of the edge extraction is above 70%. That is to say, the upper limit of the number of nodes in drone swarm interacting system is 15. When the number exceeds 15, LIGE will no longer be useful.

From the two aspects of edge types and pattern information, we analyze the reason that the LIGE shows different performance and efficiency in different interacting systems.

For Springs and Charged particle interacting systems, the most obvious difference is the pattern information. In the Charged particle interacting system, the pattern information will be weak if the nodes stay far apart because there is a cubic term on the denominator in the Coulomb's law. LIGE predicts the edge type with weak pattern information as the other edge type without pattern information. Therefore, the weak pattern information can be explained for the latent interaction graph extraction with only 76.25% accurately.

For our drone swarm interacting system, its edge types are the same as the Springs' edge types. Besides, our drone swarm is modeled by a rule-law of universal gravitation, $F_u = Gm_i m_j \frac{r_i - r_j}{|r_i - r_j|^3}$. Therefore, its pattern information is also weak when nodes stay far away. For the edges without pattern information, LIGE can predict the edge type accurately. But for the edges with weak pattern information, the accuracy for the edge type prediction will drop. Comparing these two interacting systems of drone swarm and Charged particles, the number of edges with weak interaction relationship in drone swarm is less than the number of Charged particles. Therefore, LIGE performs better accuracy in drone swarm interacting system than Charged particles interacting system for the task of latent interaction graph extraction.

Robustness for the Noise In actual drone swarm combat environment, the trajectory data of drone swarm are likely to be not the clean data due to the interference of all kinds of

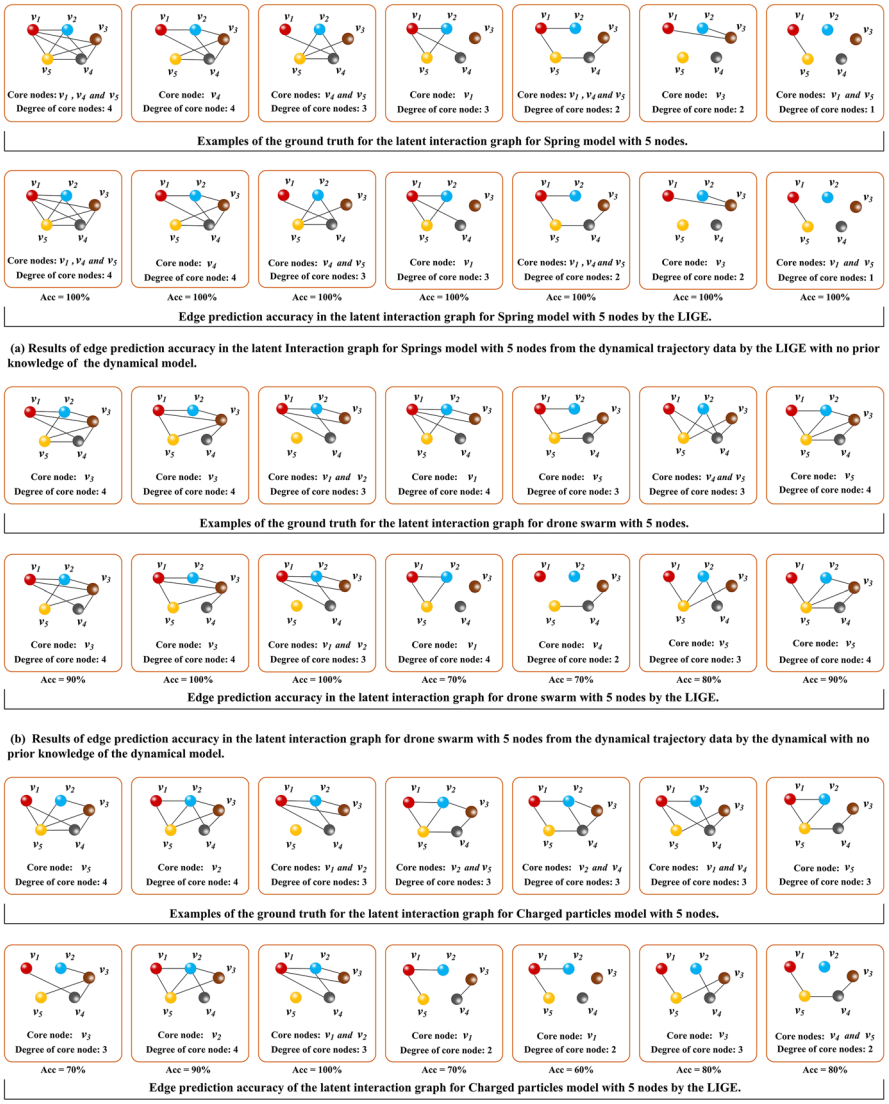

Fig. 9 **a** Results of the latent interaction extraction for Springs model with 5 nodes by the LIGE. **b** Result of the latent interaction extraction for drone swarm with 5 nodes by the LIGE. **c** Result of the latent interaction extraction for Charged particles model with 5 nodes by the LIGE

noise. Therefore, this experiment is designed to demonstrate the efficiency and robustness of our framework when the noise exist. In drone swarm, the most common kind of noise is the disturbance of the location and velocity of the nodes. Therefore, the disturbance of the location and velocity with different parameters is added into each clean trajectory data.

Table 3 summarizes the results for the accuracy of the unsupervised edge prediction in the latent interaction graph by the LIGE based on the Drone swarm datasets with noise. From the Table 3, it is easy to find that LIGE is effective and robust when the number of nodes does

Table 3 Accuracy (in %) of the unsupervised edge prediction in the latent interaction graph by our framework based on the Drone swarm datasets with noise

σ	Number of nodes		
	5	10	15
0.01	89.65	73.82	69.15
0.05	87.73	70.45	67.14
0.06	86.60	69.58	64.58
0.07	83.00	62.31	62.78
0.08	58.28	58.97	60.51
0.09	56.47	57.71	60.42
0.1	52.66	55.56	55.81

The significance of bolding the value of σ is to better reflect accuracy (in %) difference of the unsupervised edge prediction in the latent interaction graph when σ takes different values

Table 4 Accuracy (in %) of the unsupervised edge prediction in the latent interaction graph based on the encoder and **Im-Encoder**

Nodes	Framework	Datasets		
		Springs	Charged particles	Drone swarm
5	**Encoder**	99.87	76.25	92.28
5	**Im-encoder**	99.48	72.72	88.84
10	**Encoder**	98.40	59.88	78.25
10	**Im-encoder**	96.88	53.83	72.03

not exceed 10 and the value of σ does not exceed 0.05. In addition, the drone swarm with 5 nodes performs better accuracy and robustness on the LIGE than the drone swarm with 10 nodes.

Computational Time of Extracting the Latent Interaction Graph The number of hidden layer units for MLP in both the encoder and decoder is 256. We set the number of hidden layer units for MLP in the encoder to 392 whereas the hidden units for MLP in the decoder is set to 24 and optimize the encoder structure so as to reduce the computational time of extracting the latent interaction graph. In addition, we call the improved encoder as **Im-Encoder**. The schematic description of the original encoder and **Im-Encoder** structure is presented in Fig. 10. At the same time, to prevent the gradient from disappearing, we utilize a skip connection between the first and the second node update function.

Table 4 summarizes the accuracy for the unsupervised edge prediction in the latent interaction graph based on the encoder and the **Im-Encoder**. Figure 11 shows the accuracy of extracting the latent interaction graph based on the encoder and **Im-Encoder** on three datasets with 5 and 10 nodes under different epochs. Figure 12 shows the computational time of by exploiting the encoder and **Im-Encoder** on three datasets with different nodes.

As we observe from Table 4 and Fig. 11, the **Im-Encoder** shows the highest accuracy in the Springs datasets when the three datasets have the same number of nodes. In contrast, the **Im-Encoder** presents the worst accuracy in the Charged particles datasets. In addition, the accuracy of the edge prediction by exploiting the **Im-Encoder** on the public Charged particles datasets with 5 nodes has dropped by 3.53 % compared to the encoder. For each interacting system with 5 nodes, it has a total of 20 edges. The decreased accuracy of 3.53 % means

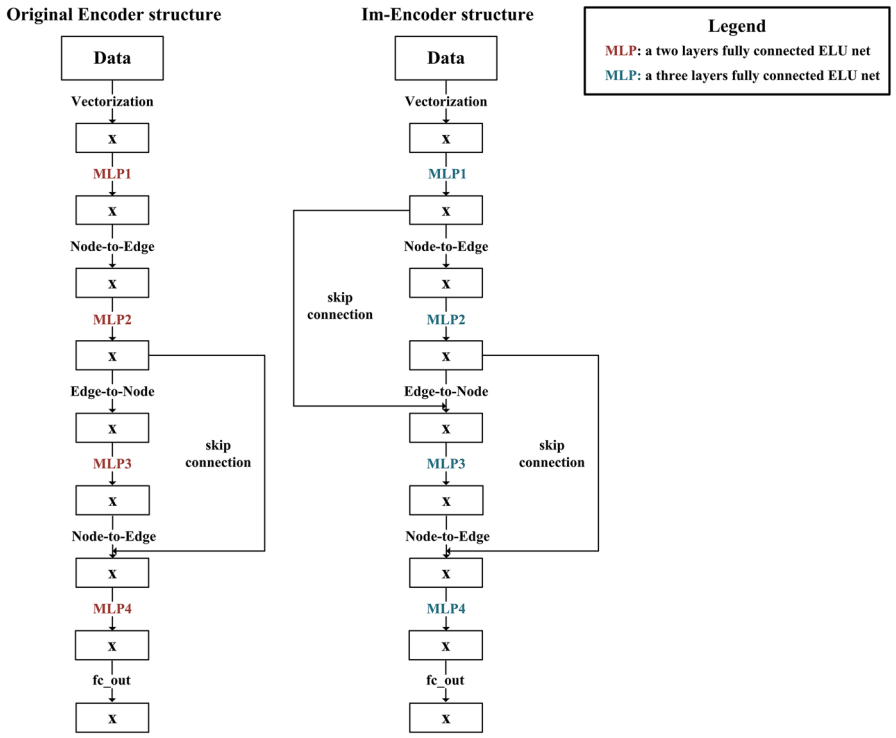

Fig. 10 Schematic description of the encoder structure and the improved encoder structure. The MLP in the original encoder is a two layers fully connected ELU net. In order to improve the performance, we utilize the MLP with a three layers fully connected ELU net in the **Im-Encoder**. At the same time, to prevent the gradient from disappearing, we utilize a skip connection between the first and the second node update function

that the number of edges that the **Im-Encoder** can correctly predict is **0.706** fewer edges than the encoder. Besides, the accuracy of the edge prediction by exploiting the **Im-Encoder** on the public Charged particles datasets with 10 nodes has dropped by 6.05 % compared to the encoder. For each interacting system with 10 nodes, it has a total of 90 edges. The decreased accuracy of 6.05 % means that the number of edges that the **Im-Encoder** can correctly predict is **5.445** fewer edges than the encoder.

Although the accuracy of the edge prediction by exploiting the **Im-Encoder** has decreased compared to the encoder, the decreased accuracy is acceptable. As we observe from Fig. 12, the computational time for the latent interaction graph extraction by exploiting the **Im-Encoder** on three datasets is reduced to at least twice the original. In addition, as the number of nodes in three datasets increases, the performance on the computational time for the latent interaction graph by exploiting the **Im-Encoder** perform better.

In the actual drone swarm combat applications, the computational time for extracting the latent interaction graph is a key factor in evaluating the efficiency and effectiveness of the LIGE to identify the influential nodes of drone swarm. The shorter computational time will contribute to boost the efficiency to recognize the core node within the limited time.

Re-ranking Figure 13 presents the example of re-ranking the influential nodes by the re-ranking module od FSCR. The input is the true trajectory data of drone swarm and latent

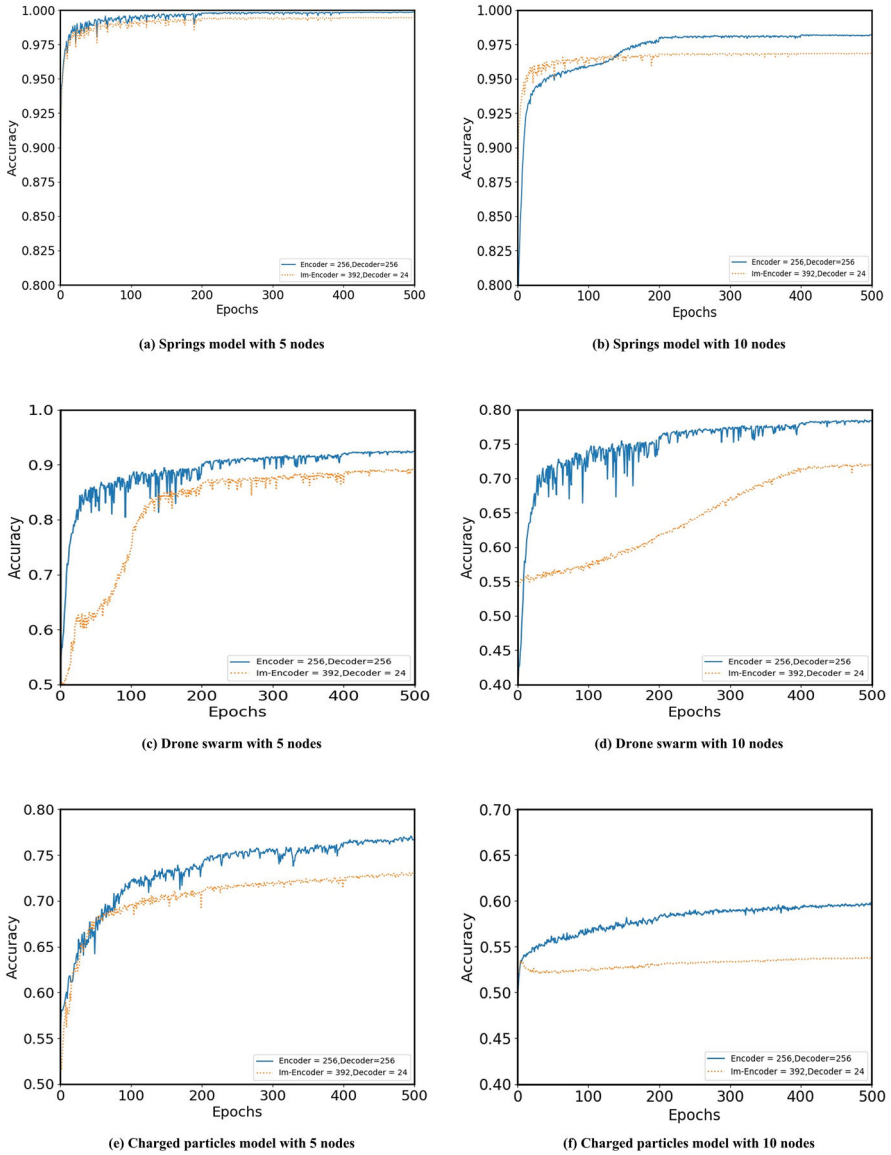

Fig. 11 Accuracy of the latent interaction graph extraction based on the encoder and **Im-Encoder** structure on three datasets with 5 and 10 nodes under different epochs

interaction graph G. The output is the results for influential nodes re-ranking by utilizing the FSC. According to the value of the FSC, we find that each node'FSC is different. The node with smaller value of FSC has a greater disturbance on the formation stability of the entire drone swarm.

Framework Figure 14 shows the example of identifying the influential nodes of drone swarm by our framework. In the DCII,, we can recognize that the core nodes are v_2, v_4 and v_5. The

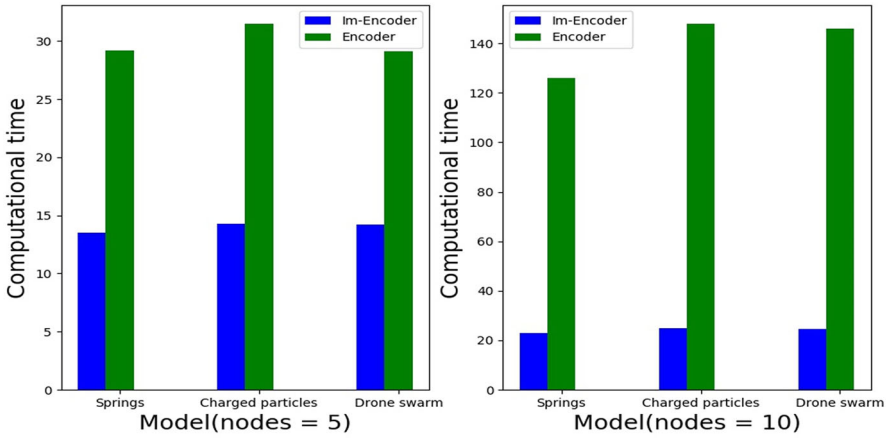

Fig. 12 Computational time of the latent interaction graph extraction by exploiting the encoder and **Im-Encoder** on three datasets with 5 and 10 nodes

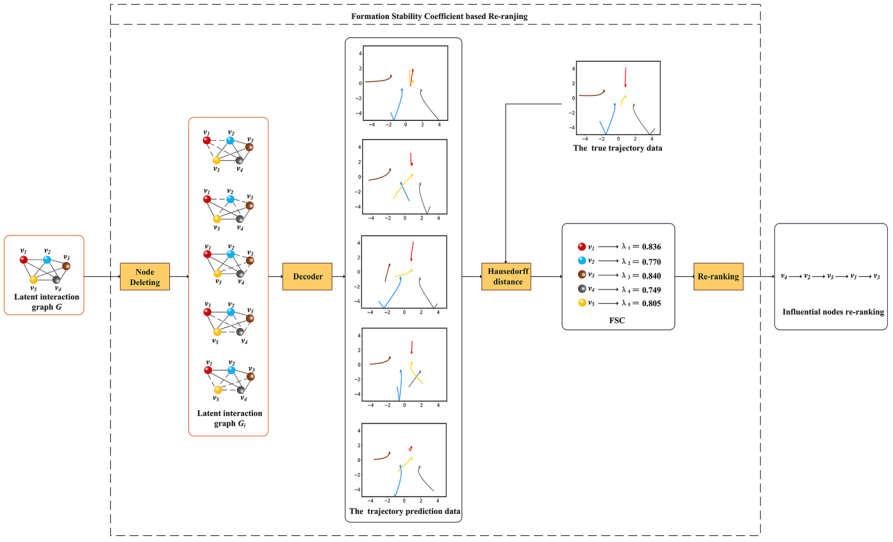

Fig. 13 Example of re-ranking the influential nodes by the module, **Formation Stability Coefficient based Re-ranking**. The input is the true trajectory data of drone swarm and latent interaction graph G. The output is the results for influential nodes re-ranking according to the novel metric of swarm formation stability FSC

influential nodes ranking result is $v_2(v_4)(v_5) \rightarrow v_1(v_3)$). Besides, we re-rank the influential nodes by the FSC in the FSCR. We can recognize that the core node is v_4. The influential nodes re-ranking result is $v_4 \rightarrow v_2 \rightarrow v_5 \rightarrow v_1 \rightarrow v_3$. It is easy to find that in the DCII, the degree of the nodes v_2, v_4 and v_5 is same, we cannot rank them by using the node degree. The FSC as a metric of swarm formation stability can handle the circumstance. In the FSCR, the node v_4 with minimum value of FSC can be recognized as the core node and gives maximum disturbance to the formation stability of drone swarm.

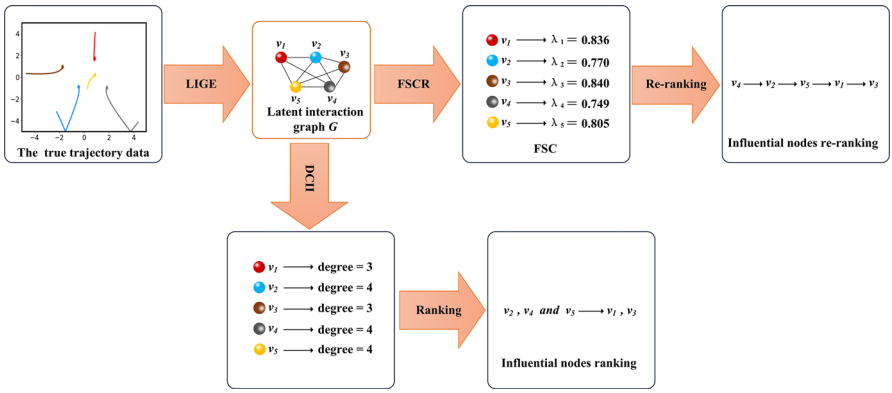

Fig. 14 A example of identifying the influential nodes of drone swarm based on our framework. The input is the true trajectory data of drone swarm. The output is the results of influential nodes ranking and re-ranking. Using the node degree as the selection indicator, we can recognize that the core nodes are v_2, v_4 and v_5. The influential nodes ranking result is $v_2(v_4)(v_5) \rightarrow v_1(v_3)$). Besides, we re-rank the influential nodes by the formation stable degree, FSC. We can recognize that the core node is v_4. The influential nodes re-ranking result is $v_4 \rightarrow v_2 \rightarrow v_5 \rightarrow v_1 \rightarrow v_3$

5 Conclusions and Future work

We deal with the core issue of capturing the pattern information of drones from the dynamic flying trajectory data by our framework. The innovation is that our framework directly extracts the latent interaction from the dynamic flying trajectory data. In addition, the influential nodes of drone swarm are ranked by using the node degree, which invokes the traditional graph model and refers to DC to measure the significance of nodes. Besides, the FSC as a metric of swarm formation stability is introduced to re-rank the influential nodes.

We measure the efficiency and effectiveness of our framework according to three factors: edge prediction accuracy, the robustness for the noise and computational time of extracting the latent interaction graph. The experimental results demonstrate that the LIGE in our framework effectively extracts the latent interaction graph by exploiting the method of graph generation, which successfully resolves the issue of capturing the pattern information between drones. In addition, we generate the Drone swarm datasets with different noise disturbance parameter and the LIGE is still effective in capturing the pattern information and extracting the latent interaction graph when the number of nodes do not exceed 10 and the noise disturbance parameters do not exceed 0.05. The experimental results demonstrate that our framework is robust when the noise exists and is suitable for the actual drone swarm combat environment. Moreover, we adjust the number of hidden layer units for MLP and utilize the skip connection in the encoder of LIGE. The experimental results demonstrate that the LIGE largely reduces the computational time of extracting the latent interaction graph. The evaluation based on these three factors demonstrate that our framework effectively extracts the latent interaction graph and accurately recognizes the core node. In addition. recognizing the core node can significantly boost the performance of locating and tracking it, which improves the control or defense against the drone combat.

In future work, we will boost our framework's ability to extract the interaction graph by exploiting sequential latent variable models to predict independent latent interaction graph for every time-step when there are more nodes in the drone swarm. On the other hand, the pattern information in the latent interaction graph in this paper is static and unchanged. In

applications, the pattern information between drones generally change as time processes. Therefore, the dynamic latent interaction graph extraction is going to become a hot issue. In short, the enhanced framework can not only boost the ability to extract the latent interaction graph, but also extract the dynamic latent interaction graph more accurately.

References

1. Weng L, Liu Q, Xia M, Song Y (2014) Immune network-based swarm intelligence and its application to unmanned aerial vehicle (uav) swarm coordination. Neurocomputing 125:134–141
2. Sharma R, Ghose D (2009) Collision avoidance between uav clusters using swarm intelligence techniques. Int J Syst Sci 40(5):521–538
3. Duan H, Li P, Yu Y (2015) A predator-prey particle swarm optimization approach to multiple ucav air combat modeled by dynamic game theory. IEEE/CAA J Autom Sin 2(1):11–18
4. Chmaj, G., Selvaraj, H.: Distributed Processing Applications for UAV/drones: A Survey. Springer International Publishing (2015)
5. Klaine PV, Nadas JPB, Souza RD, Imran MA (2018) Distributed drone base station positioning for emergency cellular networks using reinforcement learning. Cognitive Computation
6. Alfeo AL, Cimino MGCA, Francesco ND, Lega M, Vaglini G (2018) Design and simulation of the emergent behavior of small drones swarming for distributed target localization. J Comput Sci 29
7. Kim S (2011) Behavior-based decentralized control for multi-uav formation flight. Indian J Genet Plant Breed 74(4):409–413
8. Tan, K.H., Lewis, M.A.: Virtual structures for high-precision cooperative mobile robotic control. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS '96 (2002)
9. Janabi-Sharifi F, Vinke D (1993) Integration of the artificial potential field approach with simulated annealing for robot path planning. In: IEEE international symposium on intelligent control
10. Quesada WO, Rodriguez JI, Murillo JC, Cardona GA, Caldern JM (2018) Leader-follower formation for UAV robot swarm based on fuzzy logic theory. Springer, Cham
11. Rafifandi R, Asri DL, Ekawati E, Budi EM (2019) Leader-follower formation control of two quadrotor uavs. SN Appl Sci 1(6):539
12. Liu C, Wang M, Zeng Q, Huangfu W (2020) Leader-following flocking for unmanned aerial vehicle swarm with distributed topology control. Sci China Inf Sci **63**(4) (2020)
13. Mirzaeinia A, Hassanalian M, Lee K, Mirzaeinia M (2019) Performance enhancement and load balancing of swarming drones through position reconfiguration. In: AIAA aviation 2019 forum
14. Sadr H, Pedram MM, Teshnehlab M (2019) A robust sentiment analysis method based on sequential combination of convolutional and recursive neural networks. Neural Process Lett 50(3):2745–2761
15. Campion M, Ranganathan P, Faruque S (2018) A review and future directions of uav swarm communication architectures. In: 2018 IEEE international conference on electro/information technology (EIT), pp 0903–0908. IEEE (2018)
16. Quattrociocchi W, Caldarelli G, Scala A (2014) Opinion dynamics on interacting networks: media competition and social influence. Sci Rep 4
17. West DB et al (1996) Introduction to graph theory, vol 2. Prentice Hall, Upper Saddle River
18. Freeman LC (1978) Centrality in social networks conceptual clarification. Soc Netw 1(3):215–239
19. Opsahl T, Agneessens F, Skvoretz J (2010) Node centrality in weighted networks: generalizing degree and shortest paths. Soc Netw 32(3):245–251
20. Okamoto K, Chen W, Li XY (2008) Ranking of closeness centrality for large-scale social networks. In: International workshop on frontiers in algorithmics, pp 186–195. Springer
21. Bonacich P (2007) Some unique properties of eigenvector centrality. Soc Netw 29(4):555–564
22. Peng Y, Long X, Lu B (2015) Graph based semi-supervised learning via structure preserving low-rank representation. Neural Process Lett 41(3):389–406
23. Guo T, Tan X, Zhang L, Liu Q, Deng L, Xie C (2019) Learning robust weighted group sparse graph for discriminant visual analysis. Neural Process Lett 49(1):203–226
24. Shi L, Zhang Y, Cheng J, Lu H (2019) Skeleton-based action recognition with directed graph neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7912–7921
25. Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G (2008) The graph neural network model. IEEE Trans Neural Netw 20(1):61–80

26. Yang J, Lu J, Lee S, Batra D, Parikh D (2018) Graph r-cnn for scene graph generation. In: Proceedings of the European conference on computer vision (ECCV), pp 670–685
27. Kipf TN, Welling M (2016) Variational graph auto-encoders
28. Cameron P (2001) The random graph revisited
29. Johnson MJ, Duvenaud DK, Wiltschko A, Adams RP, Datta SR (2016) Composing graphical models with neural networks for structured representations and fast inference, pp 2946–2954
30. Gmez-Bombarelli R, Wei JN, Duvenaud D, Hernndez-Lobato JM, Snchez-Lengeling B, Sheberla D, Aguilera-Iparraguirre J, Hirzel TD, Adams RP, Aspuru-Guzik A (2016) Automatic chemical design using a data-driven continuous representation of molecules. ACS Cent Sci
31. Kipf T, Fetaya E, Wang KC, Welling M, Zemel R (2018) Neural relational inference for interacting systems
32. Graph theory
33. Kimura M, Saito K, Nakano R (2007) Extracting influential nodes for information diffusion on a social network. AAAI 7:1371–1376
34. Huttenlocher DP, Klanderman GA, Rucklidge WJ (1993) Comparing images using the hausdorff distance. IEEE Trans Pattern Anal Mach Intell 15(9):850–863
35. Peng Y, Long X, Lu BL (2015) Graph based semi-supervised learning via structure preserving low-rank representation. Kluwer, Dordrecht
36. Page L, Brin S, Motwani R, Winograd T (1999) The pagerank citation ranking: bringing order to the web. pp 161–172
37. Gu YR, Zhu ZY (2017) Node ranking in complex networks based on leaderrank and modes similarity. J Univ Electron Sci Technol China 46(2):441–448
38. Molaei S, Farahbakhsh R, Salehi M, Crespi N (2020) Identifying influential nodes in heterogeneous networks. Expert Syst Appl
39. Dai J, Wang B, Sheng J, Sun Z, Duan G (2019) Identifying influential nodes in complex networks based on local neighbor contribution. IEEE Access (99), 1
40. Wen T, Jiang W (2019) Identifying influential nodes based on fuzzy local dimension in complex networks
41. Tang J, Zhang R, Yao Y, Yang F, Zhao Z, Hu R, Yuan Y (2019) Identification of top-k influential nodes based on enhanced discrete particle swarm optimization for influence maximization. Physica A 513:477–496
42. Zareie A, Sheikhahmadi A, Jalili M (2019) Influential node ranking in social networks based on neighborhood diversity. Future Gener Comput Syst 94:120–129
43. Shannon CE (2001) A mathematical theory of communication. ACM SIGMOBILE Mobile Comput Commun Rev 5(1):3–55
44. Topsoe, F., Fuglede, B.: Jensen-shannon divergence and hilbert space embedding. In: International symposium on information theory (2005)
45. Martnez V, Berzal F, Cubero JC (2016) A survey of link prediction in complex networks. ACM Comput Surv 49(4):1–33
46. Lu L, Zhou T (2011) Link prediction in complex networks: a survey. Phys A Stat Mech Appl 390(6):1150–1170
47. Feng X, Zhao J, Xu K (2012) Link prediction in complex networks: a clustering perspective. Eur Phys J B 85(1):3
48. Li Z, Li S (2020) Saturated pi control for nonlinear system with provable convergence: an optimization perspective. IEEE Trans Circuits Syst II: Express Briefs
49. Li Z, Li C, Li S, Cao X (2019) A fault-tolerant method for motion planning of industrial redundant manipulator. IEEE Trans Ind Inf 16(12):7469–7478
50. Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks
51. Veličković P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y (2017) Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
52. Schlichtkrull M, Kipf TN, Bloem P, Van Den Berg R, Titov I, Welling M (2018) Modeling relational data with graph convolutional networks. In: European semantic web conference, pp 593–607. Springer
53. Tang J, Qu M, Wang M, Zhang M, Yan J, Mei Q (2015) Line: Large-scale information network embedding, pp 1067–1077
54. Park H, Neville J (2019) Exploiting interaction links for node classification with deep graph neural networks. In: IJCAI, pp 3223–3230
55. Li L, Zhao K, Li S, Sun R, Cai S (2020) Extreme learning machine for supervised classification with self-paced learning. Neural Process Lett, pp 1–22
56. Xiao X, Wei L (2020) Robust subspace clustering via latent smooth representation clustering. Neural Process Lett, pp 1–21

57. Liu T, Martin G (2020) Joint feature selection with dynamic spectral clustering. Neural Process Lett, pp 1–19 (2020)
58. Chang WL, Tay KM, Lim CP (2017) A new evolving tree-based model with local re-learning for document clustering and visualization. Neural Process Lett 46(2):379–409
59. Ferreira TAE, Vasconcelos GC, Adeodato PJL (2008) A new intelligent system methodology for time series forecasting with artificial neural networks. Neural Process Lett 28(2):113–129
60. Paasen B, Gopfert C, Hammer B (2018) Time series prediction for graphs in kernel and dissimilarity spaces. Neural Process Lett 48(2):669–689
61. Oneto L, Navarin N, Sperduti A, Anguita D (2018) Multilayer graph node kernels: stacking while maintaining convexity. Neural Process Lett 48(2):649–667
62. Atk A, Shuai LA, Xc B. Control framework for cooperative robots in smart home using bio-inspired neural network. Measurement 167
63. Khan AT, Li S, Kadry S, Nam Y (2020) Control framework for trajectory planning of soft manipulator using optimized rrt algorithm. IEEE Access 8:171730–171743
64. Khan AT, Li S. Human guided cooperative robotic agents in smart home using beetle antennae search. Sci China Inf Sci
65. Danielsson PE (1980) Euclidean distance mapping. Comput Graphics Image Process 14(3):227–248
66. Berndt DJ, Clifford J (1994) Using dynamic time warping to find patterns in time series. In: KDD workshop, vol 10, pp 359–370. Seattle, WA, USA (1994)
67. Alt H, Godau M (1995) Computing the fréchet distance between two polygonal curves. Int J Comput Geom Appl 5(01n02): 75–91
68. Liao Z, Hu H, Liu Y (2020) Action recognition with multiple relative descriptors of trajectories. Neural Process Lett 51(1):287–302
69. Wangflorence Y (2015) Takatsukamasahiro: Self-organizing map (som) based data navigation for identifying shape similarities of graphic logos. Neural Process Lett
70. Wang Q, Zhuang D, Qu X, Xie H (2020) Trajectory prediction of uav swarm based on neural relational inference model without physical control law. In: The 39th Chinese control conference