Dermatology Research: SkinInsights

PRIME OPEN ACCESS

Volume 1, Issue 1

Research Article

Date of Submission: 02 April, 2025
Date of Acceptance: 23 June, 2025
Date of Publication: 01 July, 2025

Efficient Graph Partitioning using LouvainSplit Algorithm

Mehrdad Javadi[1*], Hossein Aghighi[2] and Mohsen Azadbakht[2]

[1]AI Developer & Remote Sensing Data Scientist Tehran Province, Iran

[2]Center for Remote Sensing and GIS Research Faculty of Earth Sciences Shahid Beheshti University, Tehran, Iran

***Corresponding Author:** Mehrdad Javadi, AI Developer & Remote Sensing Data Scientist Tehran Province, Iran.

Citation: Javadi, M., Aghighi, H., Azadbakht, M. (2025). Efficient Graph Partitioning using LouvainSplit Algorithm. *Dermatol Res SkinInsights, 1*(1), 01-10.

Abstract

TGraph partitioning is essential for uncovering cohesive communities within complex networks. This paper introduces LouvainSplit, an innovative algorithm designed to enhance graph partitioning efficiency and accuracy. LouvainSplit leverages advanced techniques in feature representation, community detection, and evaluation, providing a robust framework for addressing challenges inherent in graph partitioning tasks across diverse domains. At its core, LouvainSplit utilizes a feature pyramid representation approach to extract both basic and summary features from input graphs at multiple granularity levels. This methodology ensures a subtle evaluation of graph data by capturing fundamental graph information alongside intricate structural patterns, thus offering a comprehensive representation of underlying community structures. A key innovation of our approach is the integration of the Louvain algorithm, renowned for its efficacy in community detection.

By leveraging pairwise cosine similarities computed from node feature vectors, the Louvain algorithm optimizes modularity iteratively, effectively partitioning graphs into cohesive communities. This iterative process facilitates the identification of significant network structures within complex networks, providing valuable insights into their organizational dynamics. To comprehensively evaluate LouvainSplit's performance, we integrated diverse graph partitioning algorithms, including PyMetis, Genetic Algorithm (GA), DBSCAN, KMeans, OPTICS, and spectral clustering to evaluate the effectiveness of LouvainSplit and using popular evaluation metrics across diverse benchmarks spanning various domains and graphs. The results demonstrate LouvainSplit's superiority in terms of recall score, modularity, and scalability compared to existing methods. Moreover, LouvainSplit maintains competitive average runtime values, ensuring efficient processing even with large-scale datasets.

Keywords: LouvainSplit, Graph partitioning, Graph Similarity Measures, Multilevel Modularity Optimization and Multiscale Community Detection

Introduction

Graph partitioning refers to the process of dividing a graph into smaller, interconnected subsets known as partitions or communities. This division aims to reveal the underlying structure of the graph by grouping nodes that share similar characteristics or functions [1,2]. In this study, we address the critical necessity for efficient graph partitioning and community detection algorithms to unravel the structural intricacies of large-scale networks spanning diverse domains such as social sciences, biology, and computer science [3-8]. To tackle this challenge, we introduce LouvainSplit, a tailored algorithm designed explicitly for this purpose. Conventional approaches often struggle to manage the vast amounts of data inherent in large-scale networks, resulting in suboptimal performance and impractical runtimes conversely, some newer algorithms require extensive computational resources to address the complexities of these networks, adding to the computational burden [9-13].

These computational demands can pose challenges in terms of processing time and resource allocation, especially when dealing with massive datasets. LouvainSplit extends the capabilities of the well-established Louvain algorithm, renowned

for its proficiency in community detection, by introducing several enhancements to bolster scalability and performance [14]. At the heart of LouvainSplit lies its innovative feature pyramid representation approach. This methodology facilitates the extraction of features at various levels of abstraction, capturing intricate structural patterns inherent in complex networks. By iteratively applying the Louvain algorithm across different levels of the feature pyramid, LouvainSplit efficiently partitions cohesive communities while preserving the network's underlying topology. Extensive experimentation across diverse real-world datasets validates LouvainSplit's superiority in terms of recall score, modularity, and scalability compared to existing methods. Furthermore, the versatile applications of LouvainSplit in social network analysis, biological network modeling, and network visualization underscore its practical utility. LouvainSplit signifies advancement in the field of graph partitioning.

Its efficiency and scalability render it a valuable tool for analyzing and comprehending complex networks, providing insights into their inherent structure and facilitating various downstream applications. In the forthcoming sections, LouvainSplit's unique approach will be explored, with a focus on maximizing the density of connections within communities, which may yield different partitioning outcomes, particularly in networks where communities are not defined solely by physical proximity. The rest of this paper is organized as follows, section 2 depicts the related works and graph partitioning literature review, the Methodology explained in section 3 and section 4 elucidates results and evaluation and finally section 5 deals with the conclusion.

Related Works

In latest years, the mission of graph partitioning has attained paramount importance throughout diverse domains, encompassing network analysis, parallel computing, and system studying [15-18]. A multitude of methodologies has been put forth to effectively deal with the challenges related to partitioning massive and tricky graphs. In this context, graph partitioning methodologies are categorized into four classes: traditional methods, heuristic and metaheuristic methods, graph neural network-based partitioning, and hybrid techniques. Each class embodies a diverse array of strategies and algorithms tailored to deal with particular aspects of graph partitioning challenges. In the following subsections, an in-intensity exploration of each class is provided, elucidating superb papers and elucidating their methodologies, merits, and obstacles.

Traditional Methods

Traditional methods in graph partitioning encompass a spectrum of techniques aimed at dividing graphs into smaller, more manageable components. Fiedler introduced the concept of algebraic connectivity, offering fundamental insights into graph connectivity and partitioning. While valuable, its applicability is limited to certain graph types [19]. Garey et al. seminal work on NP-completeness lays the theoretical groundwork for understanding the computational complexity of graph problems, including partitioning [20]. Fiduccia et al. provided a linear-time heuristic for enhancing network partitions, though constrained by its linear-time complexity [21]. Pothen et al. proposed partitioning sparse matrices using graph eigenvectors, beneficial for scientific computing but potentially inefficient for dense matrices or large-scale problems [22]. Karypis et al. Presented multilevel k-way partitioning scheme for irregular graphs is renowned for its scalability, yet complexity may escalate with graph size [23].

Hagen et al. Introduced spectral methods for partitioning and clustering, offering an alternative to heuristic approaches but potentially demanding in terms of computation and parameter tuning [24]. Karypis et al. Developed METIS, a software package, providing a versatile toolset for partitioning unstructured graphs, meshes, and computing fill-reducing orderings of sparse matrices [25]. Leveraging various algorithms and techniques, METIS efficiently addresses partitioning challenges, making it a valuable asset in graph partitioning research and applications since now. However, one limitation of METIS is its scalability concerning very large graphs, where it may encounter performance issues or require substantial computational resources. Additionally, while METIS offers effective partitioning solutions, its effectiveness can vary depending on the specific characteristics of the input graph and the partitioning objectives, necessitating careful consideration and potentially additional customization for optimal results in diverse scenarios. Despite these limitations, METIS remains widely used and respected in the field of graph partitioning, contributing significantly to the advancement of partitioning methodologies and their practical applications.

Behbahani et al. Propose an efficient solution for the r-move k-partitioning problem, focusing on minimizing weighted cuts by relocating a limited number of non-terminal nodes between partitions [26]. However, their method's applicability is somewhat constrained to this specific problem domain. Lau et al. Present fast algorithms for directed graph partitioning using flow-based techniques and reweighted eigenvalues, but they may encounter limitations with certain graph types [27]. Meyerhenke et al. Tackle the parallelization of graph partitioning for complex networks, leveraging label propagation techniques for multilevel partitioning [28]. While their approach enables efficient parallelization, it might struggle with certain network structures. Lastly, Gao et al. introduce RaftGP, a method balancing partitioning quality and efficiency through hierarchical model selection, yet it may require meticulous parameter tuning for optimal performance [29].

Heuristic and Metaheuristic Methods

Heuristic and metaheuristic methods play a crucial role in graph partitioning, offering practical and often efficient solutions to partitioning challenges. Heuristic methods providing approximate solutions that may not guarantee optimality but are often effective in practice. On the other hand, metaheuristic methods, such as genetic algorithms and

optimization-based approaches, aim to explore solution spaces more comprehensively, potentially offering better-quality solutions at the cost of increased computational overhead [30]. Anand et al. Explore polynomial-time approximation algorithms for sparsest cut and small set expansion, offering efficient methods to optimize sparse cuts in graphs [31]. Their work provides valuable tools for addressing optimization challenges, although the solutions may not always guarantee optimality. Shahidul et al. Presented a new algorithm for predicting protein complexes in large protein-protein interaction networks [32]. By partitioning the graph into densely connected sub-graphs using a metaheuristic approach and biological heuristics. However, scalability and applicability to diverse biological contexts may be limited.

Graph Neural Network-Based Partitioning

Graph Neural Network-Based Partitioning represents an approach leveraging neural network architectures to tackle graph partitioning challenges [33–35]. Velickovic et al. Proposed Deep Graph Infomax, focusing on unsupervised learning of node representations in graph-structured data [36]. While offering promising results, this method may necessitate careful hyperparameter tuning and might not generalize well across different graph structures. Wang et al. Introduced self-attention mechanisms into GCNs for modeling global dependencies in graph-structured data [37]. While promising, the computational complexity introduced by self-attention mechanisms and their effectiveness on different scales of input graph data warrant further investigation. Huang et al. Present CATGNN, a distributed training system for graph neural networks designed to handle large-scale real-world graphs efficiently [38]. Their method offers scalability, enabling the training of graph neural networks on massive datasets. However, it may demand significant computational resources for distributed training, posing a potential drawback for resource-constrained environments.

Hybrid Methods

Hybrid Methods represent an innovative approach that combines multiple techniques or methodologies to address complex problems in various domains. Fan et al. Introduce an application-driven hybrid partitioning strategy that enhances graph partitioning by learning a cost model specific to the target graph algorithm [39]. This approach enables the development of partitioners tailored to the learned cost model, offering a customizable solution. However, its implementation requires domain-specific knowledge for constructing the cost model, which could be a limitation for users lacking expertise in the target application domain. Wang et al. Propose a novel clustered cell-free networking scheme that employs graph partitioning techniques to decompose networks into smaller subnetworks [39].

While offering a fresh perspective on optimizing cell-free networking, this approach may face challenges related to scalability or applicability, particularly in scenarios involving large-scale or complex network architectures. Zeng et al. Present Wind GP, a comprehensive graph partitioning algorithm designed specifically for edge partitioning on heterogeneous machines [40]. Wind GP emphasizes workload balancing and resource utilization, making it a promising solution for optimizing graph partitioning on diverse computing architectures. However, despite its efficiency, Wind GP may encounter challenges in handling certain types of graphs, suggesting potential limitations in scalability or adaptability to varying graph structures.

Methodology

Research methodology entails a comprehensive framework for graph partitioning, with a focus on uncovering meaningful community structures within diverse datasets. At its core lies the utilization of a feature pyramid representation, enabling the capture of both basic and abstract features from input graphs at varying levels of granularity. Through the construction of this feature pyramid, fundamental graph statistics are extracted alongside more intricate structural patterns, facilitating a subtle analysis of graph data. Methods such as basic feature extraction and percentile-based quantization for abstract feature derivation ensure a comprehensive representation of the underlying graph structures. Central to our methodology is the application of the Louvain algorithm, a renowned technique for community detection in graphs [14]. Pairwise cosine similarities computed from node feature vectors are leveraged to construct a similarity graph, upon which the Louvain algorithm operates iteratively [41,42]. Through this process, the graph is partitioned into cohesive communities, optimizing the modularity score to achieve distinct community structures. Furthermore, our methodology offers the flexibility to refine partitions to a specified number of clusters, providing fine-grained control over the partitioning process.

In evaluating the efficacy of our approach, some of evaluation metrics including recall and accuracy are employed [43]. These metrics allow for a comprehensive assessment of the quality of the generated graph partitions, considering both individual clusters and the overall partitioning performance. Empirical experiments conducted on a diverse set of datasets spanning various domains, such as protein structures, chemical compounds, citation networks, social networks, and movie collaborations, aim to demonstrate the versatility and effectiveness of our methodology in capturing meaningful graph structures and facilitating better community detection and partitioning tasks. Our research endeavors to contribute to the field of graph analytics by providing a robust methodology for graph partitioning that can be applied across a wide range of domains. By integrating advanced techniques in feature representation, community detection, and evaluation, we aspire to advance the state-of-the-art in graph analysis and enable insights-driven decision-making in diverse application areas.

Through empirical validation and extensive experimentation, the efficacy of our methodology is validated, paving the way for its adoption in real-world scenarios and addressing pressing challenges in network analysis and community

detection. The Louvain algorithm, a key component of our methodology, optimizes the modularity score to identify communities within the graph [14]. This algorithm operates in two phases: the bottom-up phase and the top-down phase. In the bottom-up phase, each node is initially assigned to its own community, followed by iterative movements to the community that maximizes the gain in modularity. The modularity score is computed using the formula of Equation 1 as:

$$Q = \frac{1}{2m}\sum_{ij}\left(\mathcal{A}_{ij} - \frac{k_i k_j}{2m}\right)\delta(\mathcal{C}_i, \mathcal{C}_j)$$ Equation 1 Modularity score of Louvain algorithm

Where A_{ij} is the weight of the edge between nodes i and j, k_i and k_j are the degrees of nodes i and j, respectively m, is the total edge weight in the graph, and $s(c_i, c_j)$ is 1 if nodes i and j are in the same community and 0 otherwise. The symbol c_i represents the community to which node i belongs. Similarly, c_j represents the community to which node j belongs. The top-down phase treats the communities found in the bottom-up phase as super-nodes and repeats the process to find larger communities at a higher level of granularity. Iterating between these two phases continues until no further improvement in modularity is possible, resulting in the final partitioning representing the communities within the graph, optimizing the modularity score to achieve cohesive and distinct community structures Equation 2 encapsulates the entirety of our research methodology, distilling each step into a unified formula for graph partitioning. The formula is expressed as:

$$\mathcal{P} = LouvainAlgorithm\left(SimilarityComputation\left(G, \{\mathcal{F}_{\mathcal{L}}\}_{\mathcal{L}=0}^{\mathcal{L}-1}, \mathcal{F}_b\right)\right)$$ Equation 2 LovainSplit Algorithm

In Equation 2, p Represents the resulting partitions or clusters, $LouvainAlgorithm(.)$ denotes the application of the Louvain algorithm to the computed pairwise similarities,$Similaritycomputation(.)$ Computes the pairwise similarities between feature representations obtained through the feature pyramid construction, G is input graph, $\{\mathcal{F}_{\mathcal{L}}\}_{\mathcal{L}=0}^{\mathcal{L}-1}$ Denotes the abstract feature representations obtained through the feature pyramid construction, capturing more complex structural patterns and f_b Represents the basic feature representation extracted from the input graph, denoting basic graph statistics.

Parameter Tuning

In our research methodology, the careful adjustment of certain parameters plays a crucial role in determining how well our algorithms perform. Two key sets of parameters are particularly important: those governing the Louvain algorithm and those involved in constructing the feature pyramid. The resolution parameter, for instance, is a vital aspect of the Louvain algorithm, influencing the level of detail in the community structure it identifies. When this parameter is increased, it tends to produce smaller, denser communities, allowing for a more detailed examination of network substructures. Conversely, decreasing the resolution parameter leads to the emergence of larger, more diffuse communities. The default value for this parameter is typically set to 1, serving as a starting point for further adjustments. Another critical parameter is the number of levels in the feature pyramid, which determines the granularity of feature extraction. Increasing this parameter allows for finer details to be captured in the extracted features, potentially leading to better partitioning outcomes. Conversely, reducing the number of levels simplifies the feature representations, resulting in broader, less detailed partitions. It's worth noting that the default value for this parameter is determined based on the characteristics of the input graph and the desired level of detail, ensuring adaptability across different datasets. Parameter tuning is essential for optimizing the performance of presented graph partitioning algorithm and ensuring it aligns with the specific characteristics of the dataset being analyzed. By carefully adjusting these hyperparameters, researchers can fine-tune the algorithm's performance, enhancing its ability to uncover meaningful insights from the data. This iterative process of optimization empowers researchers to navigate the complexities of network analysis effectively, maximizing the utility of their findings across various application domains.

Comparative Analysis: Other Graph Partitioning Algorithms

In ensuring a comprehensive evaluation, several established graph partitioning algorithms were integrated alongside LouvainSplit, each selected to represent different approaches to partitioning graphs, encompassing traditional, heuristic and metaheuristic, and advanced methods. The chosen algorithms encompass PyMetis, a widely-used graph partitioning library based on the METIS algorithm, which was employed to efficiently partition graphs, with the goal of minimizing edge cuts and achieving balanced partitions across diverse applications [25]. Robust implementations provided by PyMetis facilitate parameter fine-tuning for optimal results, allowing for comparison of its performance against LouvainSplit across multiple benchmarks. The genetic algorithm (GA), inspired by natural selection and evolution, was adapted for graph partitioning, enabling solutions to evolve through genetic operators such as mutation and crossover [44,45]. Offering a flexible approach, GA has the potential to uncover high-quality partitions in complex networks.

Additionally, DBSCAN, a density-based clustering algorithm commonly used for spatial data analysis, was applied to graph partitioning by treating nodes as spatial points and clustering them based on density connectivity, identifying dense regions corresponding to cohesive communities [46]. KMeans clustering, a centroid-based algorithm, partitions data into K clusters based on centroid similarity, was adapted for graph partitioning by representing nodes as data points and clustering them into partitions using similarity metrics, offering a straightforward approach effective in specific scenarios [47]. OPTICS, similar to DBSCAN, detects dense regions and hierarchical clustering structures, providing

insights into the hierarchical organization of communities within graphs as an alternative to conventional partitioning algorithms [48]. Lastly, spectral clustering partitions data based on eigenvectors of a similarity matrix, providing a mathematical framework capturing intricate relationships between nodes, thereby offering a comprehensive approach to graph partitioning [49].

Results and Evaluation

Section 4 evaluates the performance of presented graph partitioning algorithms using various benchmarks, implemented algorithms, evaluation metrics and discussion of results.

Benchmarks and Datasets

In this project, a diverse set of benchmarks and datasets is employed to rigorously evaluate the performance of the implemented graph partitioning algorithms across various real-world scenarios. Each dataset serves a specific purpose and provides valuable insights into different aspects of graph analysis. Here is a brief overview of the datasets used:

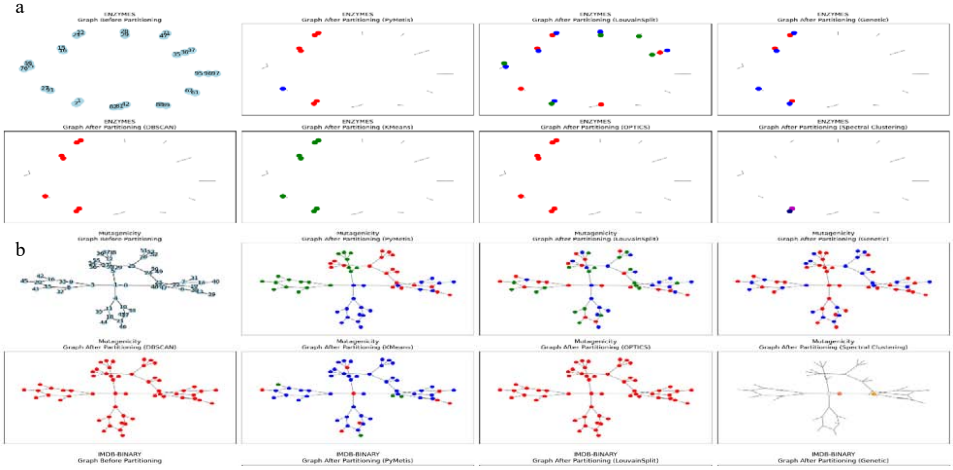
- **Proteins Dataset:** This dataset, comprising 1,113 protein graphs, is utilized for the study of protein structures and amino acid sequences. The effectiveness of graph partitioning algorithms in bioinformatics applications such as protein clustering and structure prediction is assessed [50].
- **Mutagenicity Dataset:** With 4,445 chemical compound graphs, this dataset plays a pivotal role in toxicity prediction and chemoinformatics research. The ability of graph partitioning algorithms to predict compound mutagenicity accurately is assessed by analyzing the molecular structures of compounds and their mutagenic labels [50].
- **Enzymes Dataset:** With 600 enzyme graphs, this dataset facilitates the exploration of enzyme structures and functions. The evaluation aims to assess how well graph partitioning algorithms can classify enzymes and predict their functions, contributing to bioinformatics and enzyme engineering research [50].
- **Imdb-Binary Dataset:** Derived from the Internet Movie Database (IMDb), this dataset comprises 1,000 movie collaboration networks. By analyzing these networks, the performance of graph partitioning algorithms in detecting communities and identifying influential actors in social networks is assessed.
- **PTC_MR Dataset:** This dataset includes 344 chemical compound graphs classified by carcinogenicity. The evaluation focuses on investigating compound structures and their carcinogenic labels to assess the ability of graph partitioning algorithms to predict compound carcinogenicity accurately [50].

These datasets enable a comprehensive evaluation of graph partitioning algorithms across various domains and scenarios, providing insights into their strengths and limitations. The statistical data presented below offers a clear understanding of the scale and scope of each dataset, facilitating informed analysis and interpretation.

Results and Analysis

This section provides a detailed look into how well the implemented graph partitioning algorithms perform across different datasets. We assess their effectiveness in solving graph partitioning tasks using two main aspects: mean runtimes and evaluation metrics. Mean runtimes tell us how long each algorithm takes to partition graphs on average. This helps us understand how fast and scalable they are which is important for practical use where we need to manage computational resources efficiently. Evaluation metrics give us numbers to measure the quality of the graph partitions created by each algorithm. These metrics include runtime, recall and accuracy. By evaluating these metrics across various datasets, we can see how well each algorithm performs.

By considering both mean runtimes and evaluation metrics, we can get a good understanding of each algorithm's effectiveness. This analysis helps readers choose the right algorithm based on their specific needs, whether it's fast computation or accurate community detection. Overall, this section helps us grasp the strengths and weaknesses of each algorithm in real-world graph partitioning tasks. The analyses and computations presented in this paper were conducted utilizing the Google Colab platform. Fig.1 and Fig.2 illustrates the results of graph partitioning generated by various algorithms, in order to produce three and seven distinct partitions, DBSCAN parameters are explicitly set to ensure consistency and reproducibility across experiments.



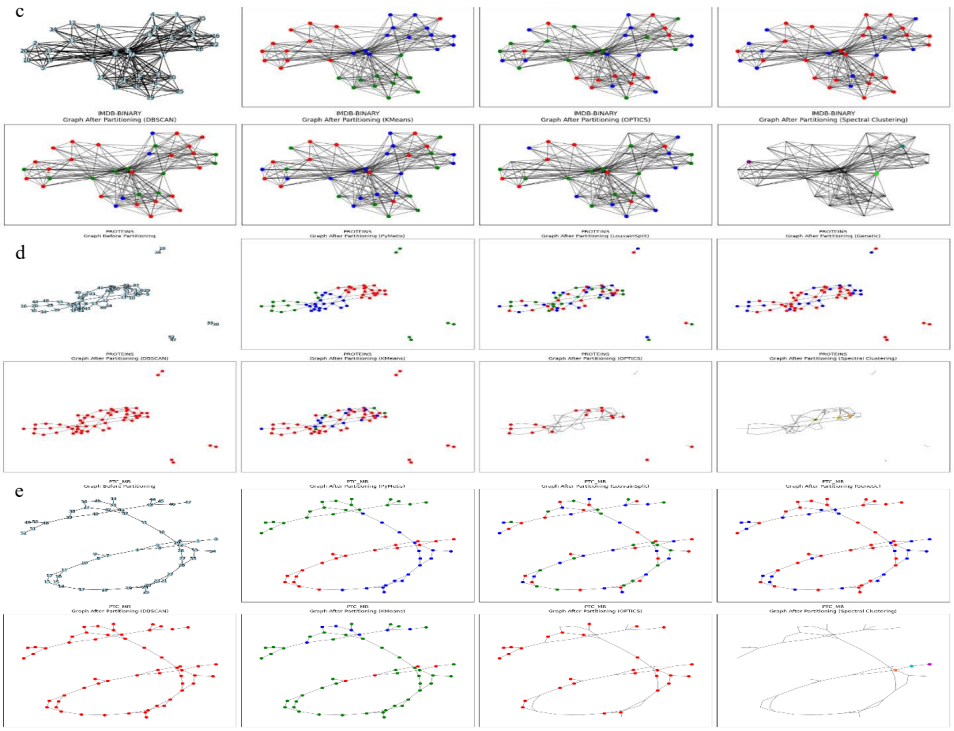


Figure 1: Graph Partitioning Results Across Various Algorithms and Datasets with Three Distinct Partitions

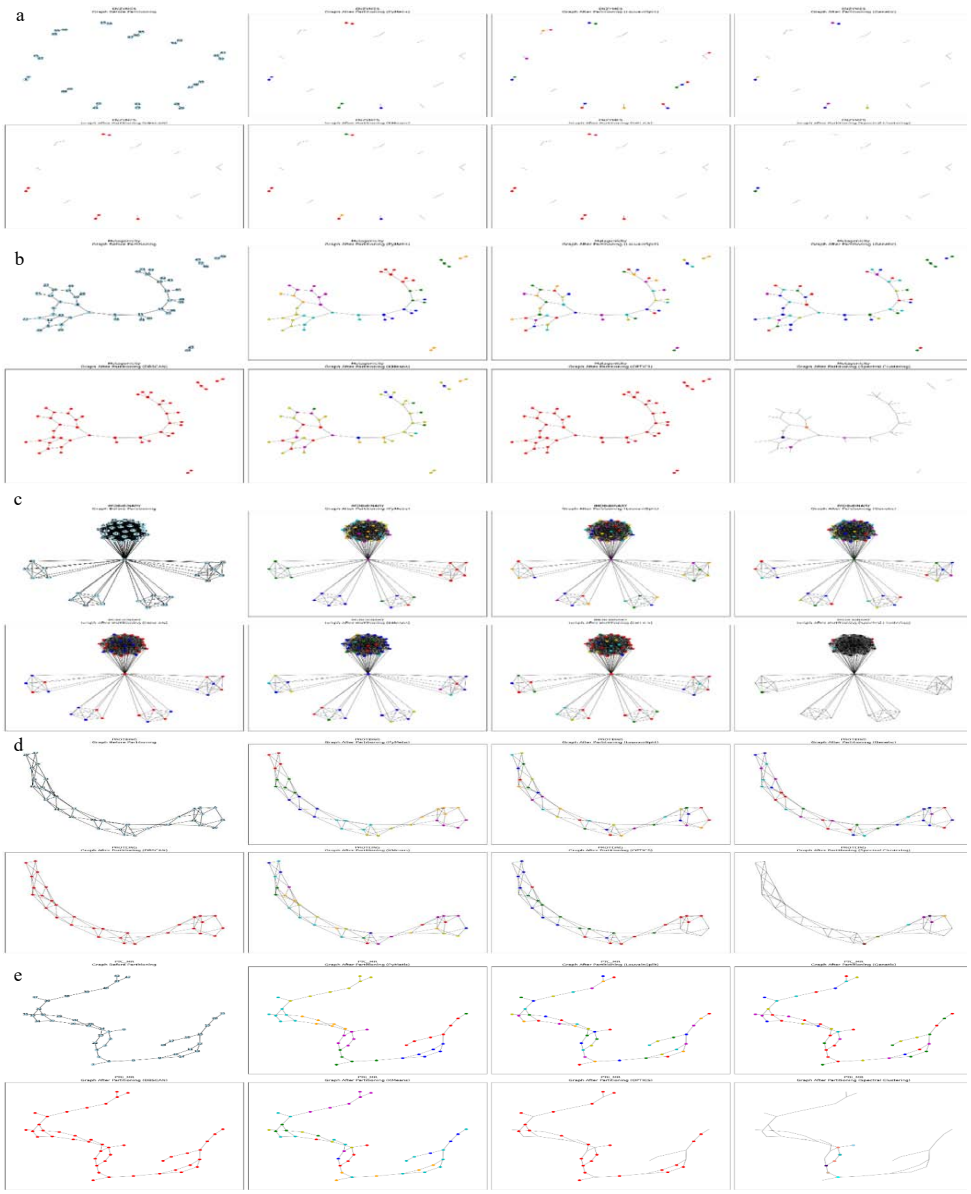


Figure 2: Graph Partitioning Results Across Various Algorithms and Datasets with Seven Distinct Partitions

Table 1 displays the performance metrics of different clustering algorithms across multiple datasets. In the table, average runtime, recall scores, and accuracy scores for each algorithm on various datasets are presented.

Dataset	Algorithm	Avg. Runtime	Avg. Recall Score	Avg. Accuracy Score
IMDB-BINARY	PyMetis	0.00068	0.1633	0.0305
	LouvainSplit	0.00678	0.1667	0.0383
	Genetic	1.03276	0.2250	0.0180
	DBSCAN	0.00255	0	0.0521
	KMeans	0.01126	0.1217	0
	OPTICS	0.01662	0	0
Mutagenicity	PyMetis	0.00060	0.0622	0.0095
	LouvainSplit	0.00621	0.1786	0.0089
	Genetic	6.46431	0.2231	0.0144
	DBSCAN	0.00220	0	0.0430
	KMeans	0.01157	0.1417	0
	OPTICS	0.02712	0	0
ENZYMES	PyMetis	0.00078	0.1139	0.0131
	LouvainSplit	0.00702	0.1028	0.0051
	Genetic	3.51114	0.2139	0.0118
	DBSCAN	0.00247	0.0	0.0372
	KMeans	0.01225	0.1528	0
	OPTICS	0.02794	0.0	0
PTC_MR	PyMetis	0.00023	0.1014	0.0364
	LouvainSplit	0.00556	0.1691	0.0085
	Genetic	0.51353	0.2174	0.0295
	DBSCAN	0.00145	0.0	0.1021
	KMeans	0.00747	0.1208	0
	OPTICS	0.00727	0.0	0
PROTEINS	PyMetis	0.00050	0.1405	0.0209
	LouvainSplit	0.00469	0.1794	0.0190
	Genetic	4.86504	0.2123	0.0156
	DBSCAN	0.00185	0	0.0564
	KMeans	0.00819	0.1405	0
	OPTICS	0.02068	0.0	0

Table 1: Performance Metrics of Various Algorithms Across Different Datasets (Avg. Runtime is in Seconds)

Discussion

In the domain of detecting nodes within complex graphs, the presented algorithm, notably Louvain Split, demonstrates remarkable efficacy compared to its counterparts, as evidenced by compelling results in Figures 1 and 2, particularly on the Enzymes dataset. The advantages of Louvain Split are manifold, positioning it as a frontrunner in graph clustering algorithms. Its superior recall score consistently outperforms other algorithms across various datasets in Table 1, underscoring Louvain Split's adeptness in accurately identifying nodes of various types of graphs, essential for accurate graph analysis in the contexts of community detection and within the realm of big data and database applications. Accurate node detection is a fundamental aspect of graph analysis, particularly in contexts where understanding individual entities within a network holds paramount importance. This precision in node detection surpasses traditional clustering methods, highlighting its criticality across diverse applications.

For instance, in anomaly detection scenarios, the precise identification of unusual nodes is indispensable for tasks like fraud detection in financial transactions, intrusion detection in computer networks and rare disease identification in biological networks [51-59]. Similarly, within network resource allocation frameworks such as transportation or social networks, the ability to pinpoint nodes with specific characteristics is crucial for optimizing traffic flow, efficiently allocating resources, and upholding infrastructure integrity [60,61]. Moreover, in social networks and marketing contexts, the identification of influential nodes is imperative for targeted advertising, executing viral marketing campaigns, and conducting opinion leadership analysis, thereby amplifying the reach and efficacy of marketing endeavors [62,63]. Furthermore, in disease spread modeling accurate node identification aids in containing outbreaks and implementing intervention strategies effectively [58,59]. Additionally, within customer segmentation and recommendation systems, precise node detection enables personalized marketing strategies, tailored product recommendations, and effective customer retention initiatives, thereby enriching user experience and engagement [64-66]. This emphasis on precise

node detection across these applications underscores its role in facilitating informed decision-making, optimization, and targeting in real-world scenarios, highlighting its indispensability in contemporary graph analysis methodologies.

Moreover, Louvain Split maintains competitive average runtime values, ensuring efficient processing even with large-scale datasets. This blend of superior recall and competitive runtime underscores its efficacy in handling complex graphs, rendering it an optimal choice for tasks demanding both precision and efficiency in graph analysis. Furthermore, the pivotal role of Louvain Split in advancing graph clustering and partitioning algorithms cannot be overstated. Its unparalleled accuracy and efficiency in detecting nodes within multipart graphs pave the way for enhanced graph analysis across various domains, marking a significant leap forward in graph clustering technology with profound implications for community detection and big data and database applications.

Conclusion

In this study, we conducted a comprehensive evaluation of graph partitioning algorithms, with a primary focus on introducing and evaluating the efficacy of LouvainSplit. Our research aimed to address the critical need for efficient graph partitioning algorithms capable of uncovering cohesive communities within intricate networks across diverse application domains not only based on physical proximity and density. Through experimentation and analysis, we have demonstrated the effectiveness of LouvainSplit in accurately partitioning graphs while considering both runtime performance, recall and accuracy. Leveraging advanced techniques in feature representation, community detection, and evaluation, LouvainSplit offers a robust framework for addressing challenges inherent in graph partitioning tasks. A key innovation of our approach is the utilization of a feature pyramid representation approach, enabling the extraction of both basic and summary features from input graphs at multiple granularity levels. This methodology ensures a perfect evaluation of graph data, capturing fundamental graph information alongside intricate structural patterns, thus providing a comprehensive representation of underlying community structures. Our evaluation, conducted across diverse benchmarks spanning various domains such as bioinformatics, chemoin formatics, and social networks, highlights Louvain Split as a standout performer. Its superior recall scores across diverse datasets underscore its proficiency in accurately identifying nodes within multipart graphs while maintaining competitive average runtime values.

The introduction of Louvain Split significantly advances the field of graph analytics by offering a novel algorithmic solution for efficient and accurate graph partitioning in addition presented model uses minimal computational resource requirements compared to deep learning and deep graph-based methods. By providing valuable insights into state-of-the-art graph partitioning algorithms, this research equips researchers and practitioners with essential knowledge for selecting suitable algorithms for specific application domains. In future research, promising avenues emerge, including multi-modal graph partitioning and leveraging edge computing for optimization. Multi-modal graph partitioning aims to dissect complex networks comprising diverse data types, potentially revealing hidden patterns within intricate systems. Meanwhile, edge computing offers opportunities for optimizing partitioning through distributed, real-time analysis at the network's edge, taking advantage of emerging infrastructure. These advancements could refine algorithms like Louvain Split and explore the integration of machine learning and graph neural networks into partitioning methods, potentially addressing evolving challenges in network analysis, clustering, and partitioning. In conclusion, the introduction and evaluation of LouvainSplit contribute to advancing our understanding of graph partitioning and facilitating enhanced graph analysis and insights-driven decision-making in diverse application areas. LouvainSplit represents a step forward in the field, with profound implications for various real-world applications.

Accessing the Code

The implementation of Louvain Split is now available as a package on PyPI. For more details, documentation, and source code, please visit our GitHub repository: https://github.com/mehrdaddjavadi/louvainsplit.

References

1. Buluç, A., Meyerhenke, H., Safro, I., Sanders, P., & Schulz, C. (2016). Recent advances in graph partitioning (pp. 117-158). Springer International Publishing.
2. Çatalyürek, Ü., Devine, K., Faraj, M., Gottesbüren, L., Heuer, T., Meyerhenke, H., Sanders, P., Schlag, S., Schulz, C., Seemaier, D., Wagner, D.: More Recent Advances in (Hyper)Graph Partitioning. *ACM Comput Surv. 55,* (2023).
3. Wang, J., Liu, C., Zhao, Y., Zhao, Z., Ma, Y., Liu, M., Shen, W.: Graph Convolutional Network Aided Inverse Graph Partitioning for Resource Allocation. *IEEE Trans Industr Inform. 20,* (2024).
4. Elden, L.: MULTIWAY SPECTRAL GRAPH PARTITIONING: CUT FUNCTIONS, CHEEGER INEQUALITIES, AND A SIMPLE ALGORITHM. *SIAM Journal on Matrix Analysis and Applications. 45,* (2024).
5. Al-Sharoa, E.M., Ababneh, B.M., Alkhassaweneh, M.A.: Robust Community Detection in Graphs. *IEEE Access. 9,* (2021).
6. Pan, J.Z., Edelstein, E., Bansky, P., Wyner, A.: A knowledge graph-based approach to social science surveys. *Data Intell. 3,* (2021).
7. Mason, O., & Verwoerd, M. (2007). Graph theory and networks in biology. IET systems biology, 1(2), 89-119.
8. Majeed, A., & Rauf, I. (2020). Graph theory: A comprehensive survey about graph theory applications in computer science and social networks. *Inventions, 5*(1), 10.
9. Kernighan, B.W., Lin, S.: An Efficient Heuristic Procedure for Partitioning Graphs. Bell System Technical Journal. 49, (1970).

10. Karypis, G., Kumar, V.: A fast and high-quality multilevel scheme for partitioning irregular graphs. *SIAM Journal of Scientific Computing. 20*, (1998).
11. Tao, Z., Liu, H., Li, J., Wang, Z., & Fu, Y. (2019, August). Adversarial graph embedding for ensemble clustering. In International Joint Conferences on Artificial Intelligence Organization.
12. Zhang, H., Li, P., Zhang, R., Li, X.: Embedding Graph Auto-Encoder for Graph Clustering. *IEEE Trans Neural Netw Learn Syst. 34,* (2023).
13. Zhang, Y., Li, L., Zhang, W., Cheng, Q.: GATC and DeepCut: Deep spatiotemporal feature extraction and clustering for large-scale transportation network partition. Physica A: Statistical Mechanics and its Applications. 606, (2022).
14. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J*ournal of Statistical Mechanics: Theory and Experiment. 2008,* (2008).
15. Onizuka, M., Fujimori, T., Shiokawa, H.: Graph Partitioning for Distributed Graph Processing. Data Sci Eng. 2, (2017).
16. Dobson, B., Watson-Hill, H., Muhandes, S., Borup, M., Mijic, A.: A Reduced Complexity Model with Graph Partitioning for Rapid Hydraulic Assessment of Sewer Networks. *Water Resour Res. 58,* (2022).
17. Jafari, N., Selvitopi, O., Aykanat, C.: Fast shared-memory streaming multilevel graph partitioning. *J Parallel Distrib Comput. 147,* (2021).
18. Washburne, A.D., Silverman, J.D., Morton, J.T., Becker, D.J., Crowley, D., Mukherjee, S., David, L.A., Plowright, R.K.: Phylofactorization: a graph partitioning algorithm to identify phylogenetic scales of ecological data. *Ecol Monogr. 89,* (2019).
19. Fiedler, M.: Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal. 23*, (1973).
20. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences). Computers and Intractability. (1979)
21. Fiduccia, C. M., & Mattheyses, R. M. (1988). A linear-time heuristic for improving network partitions. In Papers on Twenty-five years of electronic design automation (pp. 241-247).
22. Pothen, A., Simon, H.D., Liou, K.-P.: Partitioning Sparse Matrices with Eigenvectors of Graphs. *SIAM Journal on Matrix Analysis and Applications. 11,* (1990).
23. Karypis, G., Kumar, V.: Parallel multilevel k-way partitioning scheme for irregular graphs. SIAM Review. 41, (1999).
24. Hagen, L., Kahng, A.B.: New Spectral Methods for Ratio Cut Partitioning and Clustering. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 11,* (1992).
25. Karypis, G., & Kumar, V. (1998). A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices. *University of Minnesota, Department of Computer Science and Engineering, Army HPC Research Center, Minneapolis, MN, 38,* 7-1.
26. Behbahani, M., Dalirrooyfard, M., Fata, E., & Nevmyvaka, Y. (2024). Graph Partitioning with Limited Moves. *arXiv preprint arXiv*:2402.15485.
27. Lau, L. C., Tung, K. C., & Wang, R. (2024). Fast algorithms for directed graph partitioning using flows and reweighted eigenvalues. In Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA) (pp. 591-624). Society for Industrial and Applied Mathematics.
28. Meyerhenke, H., Sanders, P., & Schulz, C. (2017). Parallel graph partitioning for complex networks. *IEEE Transactions on Parallel and Distributed Systems, 28*(9), 2625-2638.
29. Gao, Y., Qin, M., Ding, Y., Zeng, L., Zhang, C., Zhang, W., ... & Bai, B. (2023, September). Raftgp: Random fast graph partitioning. *In 2023 IEEE High Performance Extreme Computing Conference* (HPEC) (pp. 1-7). IEEE.
30. Turkoglu, B., Uymaz, S.A., Kaya, E.: Clustering analysis through artificial algae algorithm. *International Journal of Machine Learning and Cybernetics. 13,* (2022).
31. Anand, A., Lee, E., Li, J., & Saranurak, T. (2024, June). Approximating small sparse cuts. *In Proceedings of the 56th Annual ACM Symposium on Theory of Computing* (pp. 319-330).
32. Shahidul Islam, M., Rafiqul Islam, M., Shawkat Ali, A.B.M.: Protein complex prediction in large protein–protein interaction network. *Inform Med Unlocked. 30*, (2022).
33. Bianchi, F.M.: Simplifying Clustering with Graph Neural Networks. Proceedings of the Northern Lights Deep Learning Workshop. 4, (2023).
34. Liao, R., Brockschmidt, M., Tarlow, D., Gaunt, A. L., Urtasun, R., & Zemel, R. (2018). Graph partition neural networks for semi-supervised classification. *arXiv preprint arXiv*:1803.06272.
35. Gatti, A., Hu, Z., Smidt, T., Ng, E. G., & Ghysels, P. (2022). Graph partitioning and sparse matrix ordering using reinforcement learning and graph neural networks. *Journal of Machine Learning Research, 23*(303), 1-28.
36. Velickovic, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y., & Hjelm, R. D. (2019). *Deep graph infomax. ICLR (poster), 2*(3), 4.
37. Wang, C., & Deng, C. (2021, January). On the global self-attention mechanism for graph convolutional networks. In 2020 25th International Conference on Pattern Recognition (ICPR) (pp. 8531-8538). IEEE.
38. Huang, X., Zhuo, W., Vuong, M. P., Li, S., Kim, J., Rees, B., & Lee, C. H. (2024). CATGNN: Cost-Efficient and Scalable Distributed Training for Graph Neural Networks. *arXiv preprint arXiv*:2404.02300.
39. Fan, W., Xu, R., Yin, Q., Yu, W., Zhou, J.: Application-driven graph partitioning. *VLDB Journal. 32,* (2023).
40. Zeng, L., Huang, H., Zheng, B., Yang, K., Shao, S., Zhou, J., ... & Chen, X. (2024). WindGP: Efficient Graph Partitioning on Heterogenous Machines. *arXiv preprint arXiv*:2403.00331.
41. Xia, P., Zhang, L., Li, F.: Learning similarity with cosine similarity ensemble. *Inf Sci* (N Y). 307, (2015).
42. Gao, C., Li, W., He, L., Zhong, L.: A distance and cosine similarity-based fitness evaluation mechanism for large-scale many-objective optimization. *Eng Appl Artif Intell. 133,* (2024).

43. Foody, G.M.: Challenges in the real-world use of classification accuracy metrics: From recall and precision to the Matthews correlation coefficient. *PLoS One. 18,* (2023).
44. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation. 6,* (2002).
45. Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: past, present, and future. *Multimedia tools and applications, 80,* 8091-8126.
46. Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In kdd (Vol. 96, No. 34, pp. 226-231).
47. Abdulnassar, A.A., Nair, L.R.: Performance analysis of Kmeans with modified initial centroid selection algorithms and developed Kmeans9+ model. *Measurement: Sensors. 25,* (2023).
48. Agrawal, K.P., Garg, S., Sharma, S., Patel, P.: Development and validation of OPTICS based spatio-temporal clustering technique. *Inf Sci (N Y). 369,* (2016).
49. Avrachenkov, K., Bobu, A., Dreveton, M.: Higher-Order Spectral Clustering for Geometric Graphs. *Journal of Fourier Analysis and Applications. 27,* (2021).
50. Kersting, K., Kriege, N. M., Morris, C., Mutzel, P., & Neumann, M. (2016). Benchmark data sets for graph kernels.
51. Li, Z., Jin, X.L., Zhuang, C.Z., Sun, Z.: Overview on Graph Based Anomaly Detection, (2021)
52. Ma, X., Wu, J., Xue, S., Yang, J., Zhou, C., Sheng, Q.Z., Xiong, H., Akoglu, L.: A Comprehensive Survey on Graph Anomaly Detection with Deep Learning. *IEEE Trans Knowl Data Eng. 35,* (2023).
53. Kim, H., Lee, B.S., Shin, W.Y., Lim, S.: Graph Anomaly Detection with Graph Neural Networks: Current Status and Challenges. *IEEE Access. 10,* (2022).
54. Nguyen, T.T., Phan, T.C., Pham, H.T., Nguyen, T.T., Jo, J., Nguyen, Q.V.H.: Example-based explanations for streaming fraud detection on graphs. *Inf Sci (N Y). 621,* (2023).
55. Li, R., Liu, Z., Ma, Y., Yang, D., Sun, S.: Internet Financial Fraud Detection Based on Graph Learning. *IEEE Trans Comput Soc Syst. 10,* (2023).
56. Bilot, T., Madhoun, N. El, Agha, K. Al, Zouaoui, A.: Graph Neural Networks for Intrusion Detection: A Survey. *IEEE Access. 11*, (2023).
57. Park, S.B., Jo, H.J., Lee, D.H.: G-IDCS: Graph-Based Intrusion Detection and Classification System for CAN Protocol. *IEEE Access. 11*, (2023).
58. Fan, L., Shi, X., Wang, Z., Zhang, R., Zhang, J.: Disease identification method based on graph features between pulse cycles. *Biomed Signal Process Control. 83,* (2023).
59. Li, L., Zhou, J., Jiang, Y., Huang, B.: Propagation source identification of infectious diseases with graph convolutional networks. *J Biomed Inform. 116*, (2021).
60. Wang, Z., Eisen, M., Ribeiro, A.: Learning Decentralized Wireless Resource Allocations with Graph Neural Networks. *IEEE Transactions on Signal Processing. 70,* (2022).
61. Wang, T., Melchior, P.: Graph neural network-based resource allocation strategies for multi-object spectroscopy. *Mach Learn Sci Technol. 3,* (2022).
62. Kiabod, M., Dehkordi, M.N., Barekatain, B., Raahemifar, K.: FSopt_k: Finding the Optimal Anonymization Level for a Social Network Graph. *Applied Sciences (Switzerland). 13,* (2023).
63. Schweimer, C., Gfrerer, C., Lugstein, F., Pape, D., Velimsky, J. A., Elsässer, R., & Geiger, B. C. (2022, April). Generating simple directed social network graphs for information spreading. In Proceedings of the ACM web conference 2022 (pp. 1475-1485).
64. Zhang, L., Li, X., Li, W., Zhou, H., Bai, Q.: Context-Aware Recommendation System using Graph-based Behaviours Analysis. *J Syst Sci Syst Eng. 30,* (2021).
65. Wu, J., Xie, H., Jiang, H.: Survey of Graph Neural Network in Recommendation System. *Journal of Frontiers of Computer Science and Technology. 16*, (2022).
66. Zhao, Y., Liu, L., Wang, H., Han, H., Pei, D.: Survey of Knowledge Graph Recommendation System Research. *Journal of Frontiers of Computer Science and Technology. 17,* (2023).