Future Generation Computer Systems 163 (2025) 107529

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Overlapping community detection using graph attention networks

Konstantinos Sismanis, Petros Potikas *, Dora Souliou, Aris Pagourtzis

School of Electrical and Computer Engineering, National Technical University of Athens, Polytecnheioupoli, Zografou, 15780, Athens, Greece

ARTICLE INFO

Keywords:
Overlapping community detection
Representation learning
GAT
GCN
GNN

ABSTRACT

Community detection is a research area with increasing practical significance. Successful examples of its application are found in many scientific areas like social networks, recommender systems and biology. Deep learning has achieved many successes (Miotto et al., 2018; Voulodimos et al., 2018) on various graph related tasks and is recently used in the field of community detection, offering accuracy and scalability. In this paper, we propose a novel method called Attention Overlapping Community Detection (AOCD) a method that incorporates an attention mechanism into the well-known method called Neural Overlapping Community Detection (NOCD) (Shchur and Günnemann, 2019) to discover overlapping communities in graphs. We perform several experiments in order to evaluate our proposed method's ability to discover ground truth communities. Compared to NOCD, increased performance is achieved in many cases.

1. Introduction

Communication systems, social networks, financial transactions, biological functions, and other systems can be represented in the form of a graph. In a graph, a community is described as a set of nodes that interact with each other more often than with other nodes outside the set. That is, they have a larger than expected number of edges among them, or share a common attribute [1]. Community detection in graphs allows us to study interconnectedness, the evolution, or the importance of a graph's structure.

Much of the research on community detection focuses on assigning each node to exactly one community and does not deal with overlapping communities. A much smaller number of publications study the formulation and discovery of overlapping communities (see Fig. 1).

Deep learning techniques have been recently applied to graph based applications. Graph Neural Networks (GNNs) [2] are specialized neural networks designed to work with graphs. GNNs leverage graph connectivity and node features to learn meaningful representations of graphs. Frequently used GNNs include the Graph Convolutional Network (GCN) [3] and Graph Attention Networks (GATs) [4,5].

We investigate the performance improvement that might occur if a GAT is used in the NOCD [6] model, instead of a GCN. To summarize, our main contributions are the following:

Model: We introduce a GAT to the NOCD model.

Evaluation: We perform a thorough evaluation of our proposed method and show that it performs better in many cases, compared to the NOCD method.

2. Community detection

Assuming an undirected, unweighted graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ is the set of nodes and $\mathbf{E}$ the set of edges, it can be represented as a binary adjacency matrix $\mathbf{A} \in (0,1)^{|V|\times|V|}$, where $A_{uv} = 1$ if $(u, v) \in E$ and $A_{uv} = 0$, otherwise. N is the number of nodes, i.e. $N = |\mathbf{V}|$ and M is the number of edges $\mathbf{E} = \{(u, v) \in \mathbf{V} \times \mathbf{V} : A_{uv} = 1\}$, i.e. $M = |\mathbf{E}|$. Every node can be associated with a D-dimensional attribute vector, represented by an attribute matrix $\mathbf{X} \in \mathbb{R}^{N\times D}$. The goal of overlapping community detection is to assign nodes into C communities .

Traditional methods

Traditional methods of community detection usually explore network structure [7]. They can be categorized as follows:

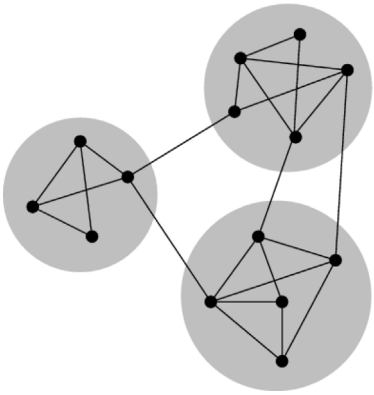
Graph Partition methods, also known as graph clustering, partition the network into a given number of K communities. The Kernighan–Lin algorithm and Spectral Bisection are two widely used methods of this type [1,8,9].

Statistical Inference methods are based on the Stochastic Block Model (SBM). A generative model that assigns nodes to communities, optimizing their probabilities of likelihood [10,11].

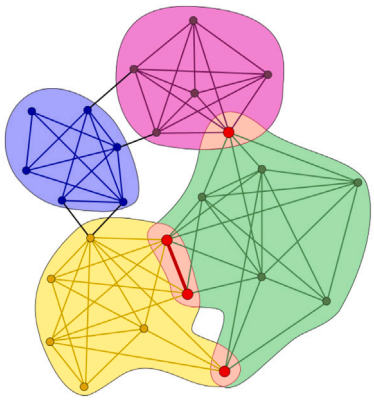
Hierarchical Clustering methods are discovering hierarchies in the network structure, They are further divided into agglomerative, divisive, or hybrid. The Girvan-Newman algorithm is a well known

* Corresponding author.
E-mail addresses: kostas.sismanis@gmail.com (K. Sismanis), ppotik@cs.ntua.gr (P. Potikas), dsouliou@gmail.com (D. Souliou), pagour@cs.ntua.gr (A. Pagourtzis).

https://doi.org/10.1016/j.future.2024.107529
Received 26 January 2024; Received in revised form 26 August 2024; Accepted 11 September 2024
Available online 21 September 2024



(a) Non-overlapping communities



(b) Overlapping communities

Fig. 1. Non-overlapping vs. overlapping communities, Wikipedia.

divisive algorithm that removes edges to discover community structures [12]. OST is a divisive algorithm also, measuring edge betweenness and employing minimum spanning trees [13]. Fast Modularity (FastQ) [14] is an agglomerative representative of this category of methods based on structural similarity [15].

Dynamical methods usually perform random walks to detect communities [16]. Information Mapping (InfoMap) calculates the minimal-length encoding [17] in order to find communities.

Spectral Clustering methods [18] discover a network partitioning, based on the normalized Laplacian matrix and the regularized adjacency matrix, that fits into the SBM. This category of algorithms is very efficient for large graphs with thousands of nodes and edges.

Density-based methods discover communities by measuring entity density. A well known example is the DBscan [19] method.

Optimizations driven methods are trying to maximize certain functions like Modularity [14] and its variant FastQ [20], The Louvain method [21] also belongs to this category. Extremal, or spectral optimization and simulated annealing are extensions of greedy methods. Normalized Mutual Information (NMI) [22], and Conductance [23] are also used as functions to maximize. Moreover, tree-based methods using modularity optimization were proposed [24–26].

Embedding based methods are converting a graph's nodes or edges to a low-dimensional vector space that is then feeded to a machine learning algorithm.

These methods are reviewed in detail in [27,28].

Deep learning methods

Methods that focus on overlapping community detection and use deep learning are reviewed in [7]:

Line Graph Neural Network (LGNN) [29] learns node representation features in directed networks, extending the Stochastic Block Model.

Neural overlapping community Detection [6] method combines the GCN representation learning with probabilistic inference.

Seed Expansion with Generative Adversarial Learning [30] is using generative adversarial learning. Its generated samples are communities, and a Graph Isomorphism Network (GIN) is used as a discriminator.

Community GAN [31] performs a competition between the motif-level AGM generator and discriminator, to optimize community membership embeddings.

ACNE: [32] expands generative adversarial learning, to optimize node and community embeddings.

Methods that implement a graph attention mechanism are reviewed in [7]. DMGI [33], HDMI [34], MAGNN [35], HeCo [36] and CP-GNN [37]. All, use K-Means for clustering, in order to detect non-overlapping communities.

3. Background

Usually, the methods proposed to detect overlapping community structure are based on the assumption, that community overlaps are less densely connected than the core of the communities. As explained in [38], the higher the number of communities a pair of nodes have in common, the higher the probability is, that they will be connected, meaning that the community overlaps formed between communities are often densely connected. Most methods based on the hypothesis stated above, will fail to detect dense overlaps, and usually detect them as separate communities, or as one community.

In [38,39], a probabilistic view of community structure is proposed that includes the formation of densely connected overlaps. An assignment of nodes to communities could be represented as a non-negative node-community affiliation matrix $\boldsymbol{F} \in \mathbb{R}_{\geq 0}^{N \times C}$, where F_{uc} is the strength of node u's membership in community c. If a community-based generative model $P(\mathcal{G}|\boldsymbol{F})$ is posited, then in order to infer the unobserved affiliation matrix $\boldsymbol{F}$ of a graph $\mathcal{G}$ would mean to detect the communities.

4. Model

4.1. Bernoulli-Poisson model

The NOCD model is based on a Bernoulli–Poisson model [39–41]. According to it, given the affiliations $\boldsymbol{F} \in \mathbb{R}_{\geq 0}^{N \times C}$, the adjacency matrix is produced, with each A_{uv} entry being sampled with mutual independence and following the same probability distribution:

$$A_{uv} \sim \text{Bernoulli}(1 - \exp(-\boldsymbol{F}_u \boldsymbol{F}_v^T)) \tag{1}$$

4.2. Cluster affiliation model for big networks

The likelihood that the graph is produced by $\boldsymbol{F}$ as introduced in [39], will be:

$$\begin{aligned} P(\boldsymbol{G}|\boldsymbol{F}) &= \prod_{(u,v)\in G} P(u,v) \prod_{(u,v)\notin G} (1 - P(u,v)) \\ &= \prod_{u.v\in E} (1 - \exp(-\boldsymbol{F}_u^T \boldsymbol{F}_v)) \prod_{u.v\notin E} \exp(-\boldsymbol{F}_u^T \boldsymbol{F}_v) \end{aligned} \tag{2}$$

The maximum log likelihood is then

$$\begin{aligned}\log(P(G|F)) &= log(\prod_{u,v\in E}(1-\exp(-F_u^T F_v))\prod_{u,v\notin E}\exp(-F_u^T F_v))\\ &= \sum_{u,v\in E}\log(1-\exp(-F_u^T F_v)) - \sum_{u,v\notin E}(F_u^T F_v)\\ &= \ell(F)\end{aligned} \tag{3}$$

The optimization problem to solve is described in Eq. (4):

$$\arg\max_F \log(P(G|F)) \tag{4}$$

4.3. Neural Overlapping Community Detection, (NOCD)

NOCD [6] instead of treating the affiliation matrix F as a free variable over which optimization is performed, calculates it using a GCN.

$$F := \mathrm{GCN}_\theta(A, X) \tag{5}$$

The negative log-likelihood that is to be minimized, is shown in Eq. (6).

$$-\log p(A|F) = -\sum_{(u,v)\in E}\log(1-\exp(-F_u F_v^T)) + \sum_{(u,v)\notin E} F_u F_v^T \tag{6}$$

Uniform distributions of existing or non-existing edges are considered to balance the two terms. The (balanced) negative log-likelihood is

$$\mathcal{L}(F) = -\mathbb{E}_{(u,v)\sim P_E}\left[\log(1-\exp(-F_u F_v^T))\right] + \mathbb{E}_{(u,v)\sim P_N}\left[F_u F_v^T\right], \tag{7}$$

where P_E and P_N are uniform distributions over edges and non-edges, respectively. The parameters $\theta^\star$ to search for, are those that minimize the loss function:

$$\theta^\star = \operatorname*{argmin}_\theta \mathcal{L}(\mathrm{GCN}_\theta(A, X)) \tag{8}$$

4.4. Overlapping Community Detection using Graph Attention Networks, (AOCD)

We propose to switch the GCN, with a GAT, for the calculation of the F affiliation matrix

$$F := \mathrm{GAT}_\theta(A, X) \tag{9}$$

The parameters we search for will now be calculated by a GAT.

$$\theta^\star = \operatorname*{argmin}_\theta \mathcal{L}(\mathrm{GAT}_\theta(A, X)) \tag{10}$$

The key idea behind a GAT is to compute each node's hidden representations, by attending over its neighbors, following a self-attention strategy.

A GCN [3] treats equally the neighbors of the target node, while a GAT [4] utilizes a self attention mechanism to allow the assignment of different weights to nodes, in the same neighborhood.

In a GCN layer, an embedding calculation for node v at layer l is explicitly assigning equal importance to all nodes.

$$h_v^{(l)} = \sigma\Big(\sum_{u\in N(v)} \underbrace{\frac{1}{|N(v)|}}_{\text{fixed importance}} W^{(l)} h_u^{(l-1)}\Big) \tag{11}$$

In a GAT layer, nodes attend over their neighborhood's message, implicitly specifying different weights to different nodes in the neighborhood

$$h_v^{(l)} = \sigma\Big(\sum_{u\in N(v)} \underbrace{\alpha_{vu}}_{\text{attention weights}} W^{(l)} h_u^{(l-1)}\Big), \tag{12}$$

where α_{vu} is the attention weights "learned" by the attention mechanism, calculated in Eqs. (13)–(15).

Table 1
Dataset statistics. K stands for 1000, C.S. for Computer Science and FB for Facebook.

Dataset	Network type	N	M	D	C
Facebook 348	Social	224	3.2K	21	14
Facebook 414	Social	150	1.7K	16	7
Facebook 686	Social	168	1.6K	9	14
Facebook 698	Social	61	270	6	13
Facebook 1684	Social	786	14.0K	15	17
Facebook 1912	Social	747	30.0K	29	46
Chemistry	Co-authorship	35.4K	157.4K	4.9K	14
Computer Science	Co-authorship	22.0K	96.8K	7.8K	18
Engineering	Co-authorship	14.9K	49.3K	4.8K	16
Medicine	Co-authorship	63.3K	810.3K	5.5K	17

The weight coefficients are calculated as:

$$e_{vu}^{(l)} = LeakyReLu(a^{(l-1)^T}(\mathbf{W}^{(\mathbf{l})} h_v^{(l-1)} \parallel \mathbf{W}^{(\mathbf{l})} h_u^{(l-1)})), \tag{13}$$

where $\parallel$ is the concatenation operation. The attention coefficients's calculation is:

$$a_{vu}^{(l)} = \mathrm{softmax}(e_{vu}) = \frac{\exp(e_{vu}^{(l)})}{\sum_{k\in N_v}\exp(e_{vk}^{(l)})} \tag{14}$$

and the embeddings at a layer l are defined as:

$$h_v^{(l)} = \sigma\Big(\sum_{u\in N(v)} a_{vu}^{l} \mathbf{W}^{(\mathbf{l})} h_u^{(l-1)}\Big). \tag{15}$$

The attention learnable parameters will be learned along with the weight parameters of the graph neural network $W^{(l)}$.

To stabilize the learning process of self-attention, a multi-head attention mechanism is used. K independent attention mechanisms, known as "heads", compute embeddings with different random initializations for their parameters. The calculated embeddings are concatenated or averaged to produce the final output.

The problem of community detection can be examined from a representation learning point of view. The affiliation matrix F could be considered an embedding of nodes into $\mathbb{R}^C_{\geq 0}$, while preserving the graph structure. Shchur and Günnemann [6] proposed the combination of probabilistic and representation points of view, to "learn" community affiliations using a graph convolutional network. This is the model we propose to extend with the use of a graph attention network and evaluate its performance through a number of tests on different kinds of datasets.

5. Evaluation

5.1. Datasets

A collection of real-world graph datasets was used in our tests. **Facebook** [42] is a collection of small ego-networks. A Facebook ego-network is a network of connections between someone's friends from the Facebook graph. The ego-networks selected consist of 50 to 800 nodes.

Four larger real-world datasets were used, with reliable ground truth overlapping community information and node attributes: **Chemistry**, **Computer Science**, **Medicine**, **Engineering** are co-authorship networks, constructed from the Microsoft Academic Graph [43]. Communities correspond to research areas, and node attributes are based on the keywords of the papers by each author. Statistics for all the datasets processed through the tests, are presented in Table 1.

5.2. GCN configuration

A two layer GCN [3] was used, like in NOCD, with its configuration unchanged.

$$F := \mathrm{GCN}_\theta(A, X) = \mathrm{ReLU}(\hat{A}\ \mathrm{ReLU}(\hat{A}X W^{(1)}) W^{(2)}) \tag{16}$$

where $\hat{A} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$, is the normalized adjacency matrix, $\tilde{A} = A + I_N$ is the adjacency matrix with self-loops and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$ is the diagonal degree matrix.

The hidden layer size is set at 128, while the final layer output size is equal to the number of ground truth communities C.

Batch normalization is applied after the first graph convolution layer and (2) L_2 regularization is applied to all weight matrices.

The values of the rest of the hyperparameters were set equal to the values proposed by the authors of NOCD. They were optimized according to the performance achieved only for the Computer Science dataset.

5.3. GAT configuration

The GAT network consists of two layers. The first layer, computes embeddings of size, $\boldsymbol{F}' = 128$, in $K = 2$ heads. The average of the two heads results is calculated and finally passed through a ReLU non-linearity [44]. The second layer consists also of $K = 2$ heads of size $\boldsymbol{F}'' = C$. The average of the two heads results is calculated and finally passed through another ReLU non-linearity. This configuration is named GAT2 in the results.

In order to test if the attention mechanism will be benefited by an increase in the number of the attention heads, we evaluated the performance of the GAT network with the same configuration but with $K = 6$ attention heads per layer. This configuration is named GAT6 in the results.

Testing took place trying different values for the hidden layer of sizes 16, 32, 64. We also tested the model's performance, applying, concatenation instead of averaging them at the first layer attention calculation. But the choice of hidden layer of size equal to 128 and the averaging of the attention heads in both layers, lead to the best results.

5.4. Hyperparameters

The remaining network hyperparameters were set equally between the two networks. Batch normalization is applied after the first graph layer. Dropout with a 50% keep probability is applied before every layer. Weight decay is applied to both weight matrices with regularization strength of $\lambda = 10^{-2}$. The feature matrix $\mathbf{X}$ is $\mathcal{L}_2$ normalized before input.

5.5. Training

The training process was performed exactly as it was implemented for the NOCD method. The model training is repeated 50 times with different initializations, for each dataset and their results are averaged. The Adam optimizer [45] is used with the default parameters. Every 50 epochs, the full training loss is computed, and if no improvement has happened for the last 500 iterations, or after 5000 epochs, the training stops.

5.6. Assigning nodes to communities

In order to compare the detected community partitionings to the ground truth community structure, the predicted continuous community affiliations $\boldsymbol{F}$ are transformed into binary community assignments. Each node u is assigned to a community c if its affiliation strength F_{uc} is above a fixed threshold ρ. The threshold is $\rho = 0.5$.

5.7. Implementation and hardware specifications

We used the reference Python implementation of NOCD, that was provided by the authors [46]. Our implementation is based on the PyTorch and PyTorch Geometric libraries. Our code can be found at https://github.com/ksismanis/overlapping-community-detection-gat.

Both models were trained on a T4 GPU with 16 GB of RAM at a time.

Table 2
Recovery of ground-truth communities measured by NMI (in %). The adjacency matrix is used as input. Results are averaged over 50 initializations. Best results are in **bold**. DNF means ran out of memory.

Dataset	GCN	GAT2	GAT6
Facebook 348	**34.6 ± 1.8**	34.5 ± 2.2	34.1 ± 2.5
Facebook 414	**55.1 ± 1.2**	51.7 ± 1.6	53.3 ± 3.5
Facebook 686	**18.7 ± 0.8**	17.2 ± 0.1	17.8 ± 1.4
Facebook 698	45.8 ± 1.7	46.0 ± 2.2	**48.5 ± 2.6**
Facebook 1684	35.4 ± 2.4	37.6 ± 2.2	**40.7 ± 2.9**
Facebook 1912	39.7 ± 1.6	39.6 ± 1.7	**39.7 ± 1.3**
Chemistry	18.2 ± 2.7	20.7 ± 2.6	**22.3 ± 2.8**
Computer Science	30.0 ± 2.5	30.1 ± 2.8	**32.3 ± 3.2**
Engineering	16.7 ± 1.8	18.2 ± 1.7	**20.2 ± 2.1**
Medicine	23.7 ± 2.5	**26.4 ± 2.6**	DNF

Table 3
Recovery of ground-truth communities measured by NMI (in %). The node attributes are used as input. Results are averaged over 50 initializations. Best results are in **bold**. DNF means ran out of memory.

Dataset	GCN	GAT2	GAT6
Facebook 348	31.9 ± 2.3	32.1 ± 2.8	**33.6 ± 2.6**
Facebook 414	52.0 ± 3.1	50.4 ± 4.0	**53.6 ± 3.6**
Facebook 686	**19.0 ± 1.4**	17.2 ± 1.3	17.2 ± 1.8
Facebook 698	35.2 ± 3.7	31.6 ± 4.4	**35.3 ± 4.0**
Facebook 1684	27.8 ± 2.2	29.9 ± 2.3	**34.6 ± 2.2**
Facebook 1912	**35.9 ± 3.0**	35.7 ± 2.8	35.4 ± 1.6
Chemistry	**43.4 ± 2.3**	42.9 ± 2.5	43.2 ± 2.7
Computer Science	**47.1 ± 2.7**	44.9 ± 3.0	45.5 ± 2.4
Engineering	35.9 ± 3.7	37.1 ± 3.0	**38.5 ± 3.2**
Medicine	**37.6 ± 2.6**	36.0 ± 2.5	DNF

6. Results and discussion

6.1. Metrics

Normalized Mutual Information (NMI) [22] is chosen to evaluate the partitioning performance because of its comprehensive meaning and its ability to compare two partitions of different size [6]. It is more robust and accurate than other metrics usually used in clustering like the Jaccard index and F1 score [39,47].

6.2. Recovery performance

Table 2 shows the results when the adjacency matrix was used as input during training and Table 3, shows the results when the node features were used as input. All results are averaged over 50 initializations, and the standard deviation of the NMI values is calculated.

When the adjacency matrix is used as input, for the seven larger datasets, a better performance is achieved with the GAT models. For four of them, the increase in performance is about 2%. When the feature matrix is used as input, the GAT2 model achieves a better score compared to the GCN model for three datasets.

In general, the GAT2 model, with two heads per layer, improves the performance, compared to the initial GCN model for more than half of the datasets tested. This shows that the attention weights calculation contributes to the model's increase in performance, when the GAT network is in use.

The GAT6 network performs even better than GAT2. This leads to the conclusion that as the attention head mechanism becomes more stable, a performance increase is achieved.

6.3. Scalability

The GAT model is converging quickly. At about 100 epochs, the Total loss is reduced close to its minimum value. Fig. 2 shows the training curves for all the datasets using the AOCD approach, when the adjacency matrix is used as input.

Looking at the execution time results in Tables 4, 5, there is a small increase in computation time for the GAT model with 2 attention heads, and a much larger one for the GAT model with 6 heads.

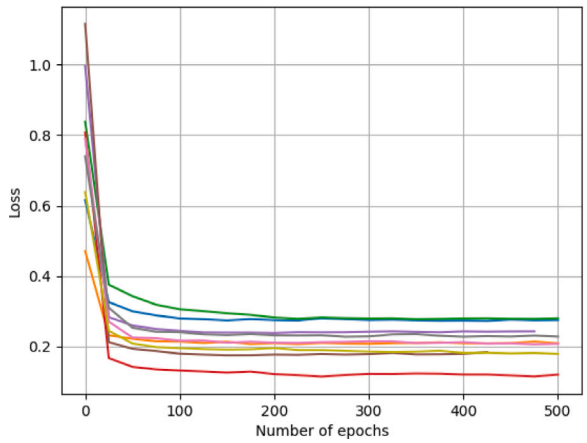


Fig. 2. Total loss convergence during training, for all datasets.

Table 4
Real time of execution, measured in seconds. The adjacency matrix is used as input. Results are averaged over 50 initializations.

Dataset	GCN	GAT2	GAT6
Facebook 348	6.946	9.782	9.465
Facebook 414	6.852	9.433	9.363
Facebook 686	7.245	9.655	9.671
Facebook 698	7.37	9.703	9.718
Facebook 1684	8.042	10.544	11.793
Facebook 1912	12.427	15.792	19.393
Chemistry	30.404	48.824	97.79
Computer Science	27.378	39.127	64.477
Engineering	16.824	25.496	41.787
Medicine	85.942	163.035	DNF

Table 5
Real time of execution, measured in seconds. The node attributes are used as input. Results are averaged over 50 initializations.

Dataset	GCN	GAT2	GAT6
Facebook 348	6.939	9.049	9.361
Facebook 414	6.75	8.806	9.533
Facebook 686	7.212	9.034	9.744
Facebook 698	7.209	9.007	9.649
Facebook 1684	7.833	10.228	12.132
Facebook 1912	12.178	14.03	19.872
Chemistry	33.976	52.238	100.691
Computer Science	37.664	50.824	97.936
Engineering	19.565	26.667	48.384
Medicine	88.916	154.728	DNF

7. Conclusions and future work

The results confirmed both models' ability to discover overlapping communities and their high scalability. Our proposed AOCD model produced an increase in performance in more than half the datasets tested.

A number of questions could be answered about the usage of Graph Attention Networks in the future. An interesting area to explore is whether quantifying the attention coefficients in relation to the structure of the community could help to create an analysis to interpret the model.

The assessment of the model's inductive capability [48] is an interesting area for future work. Could the model successfully predict the communities, a newly added to the graph node will belong to, without repeating the training process? Inductive performance could be evaluated on dynamically evolving networks.

So far, many interesting improvements to GATs have been proposed [2,5]. Further testing and evaluations, with these proposed graph attention modifications, could achieve better results.

The model's scalability could also be further improved, by performing the computation over multiple attention heads in parallel.

For most of the overlapping community methods, a crucial issue that remains open, is their dependence on explicitly declaring the number of communities K as an input. In real-world datasets, there is no prior knowledge of the number of communities that exist. Research should be focused on the question of how to determine the number of hidden communities in a graph.

CRediT authorship contribution statement

Konstantinos Sismanis: Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Conceptualization. **Petros Potikas:** Writing – original draft, Methodology, Investigation, Conceptualization. **Dora Souliou:** Methodology, Investigation, Conceptualization. **Aris Pagourtzis:** Methodology, Investigation, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] S. Fortunato, Community detection in graphs, Phys. Rep. 486 (3–5) (2010) 75–174.
[2] H. Zhang, Z. Yu, G. Dai, G. Huang, Y. Ding, Y. Xie, Y. Wang, Understanding gnn computational graph: A coordinated computation, io, and memory perspective, Proc. Mach. Learn. Syst. 4 (2022) 467–484.
[3] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, 2016, arXiv preprint arXiv:1609.02907.
[4] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, 2017, arXiv preprint arXiv:1710.10903.
[5] S. Brody, U. Alon, E. Yahav, How attentive are graph attention networks? 2021, arXiv preprint arXiv:2105.14491.
[6] O. Shchur, S. Günnemann, Overlapping community detection with graph neural networks, 2019, arXiv preprint arXiv:1909.12201.
[7] X. Su, S. Xue, F. Liu, J. Wu, J. Yang, C. Zhou, W. Hu, C. Paris, S. Nepal, D. Jin, et al., A comprehensive survey on community detection with deep learning, IEEE Trans. Neural Netw. Learn. Syst. (2022).
[8] B.W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, Bell Syst. Tech. J. 49 (2) (1970) 291–307.
[9] E.R. Barnes, An algorithm for partitioning the nodes of a graph, SIAM J. Algebr. Discrete Methods 3 (4) (1982) 541–550.
[10] P.W. Holland, K.B. Laskey, S. Leinhardt, Stochastic blockmodels: First steps, Soc. Netw. 5 (2) (1983) 109–137.
[11] B. Karrer, M.E. Newman, Stochastic blockmodels and community structure in networks, Phys. Rev. E 83 (1) (2011) 016107.
[12] M. Girvan, M.E. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. 99 (12) (2002) 7821–7826.
[13] A. Pagourtzis, D. Souliou, P. Potikas, K. Potika, Overlapping community detection via minimum spanning tree computations, in: 2020 IEEE Sixth International Conference on Big Data Computing Service and Applications (BigDataService), IEEE, 2020, pp. 62–65.
[14] M.E. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69 (6) (2004) 066133.
[15] F.D. Zarandi, M.K. Rafsanjani, Community detection in complex networks using structural similarity, Phys. A 503 (2018) 882–891.
[16] P. Pons, M. Latapy, Computing communities in large networks using random walks, in: International Symposium on Computer and Information Sciences, Springer, 2005, pp. 284–293.
[17] M. Rosvall, C.T. Bergstrom, Maps of random walks on complex networks reveal community structure, Proc. Natl. Acad. Sci. 105 (4) (2008) 1118–1123.
[18] A.A. Amini, A. Chen, P.J. Bickel, E. Levina, Pseudo-likelihood methods for community detection in large sparse networks, Ann. Statist. 41 (4) (2013) 2097–2122.
[19] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al., A density-based algorithm for discovering clusters in large spatial databases with noise, in: Kdd, Vol. 96, 1996, pp. 226–231.
[20] A. Clauset, M.E. Newman, C. Moore, Finding community structure in very large networks, Phys. Rev. E 70 (6) (2004) 066111.

[21] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Statist. Mech.:Theory Exp. 2008 (10) (2008) P10008.
[22] A.F. McDaid, D. Greene, N. Hurley, Normalized mutual information to evaluate overlapping community finding algorithms, 2011, arXiv preprint arXiv:1110.2515.
[23] Y. Gao, H. Zhang, Y. Zhang, Overlapping community detection based on conductance optimization in large-scale networks, Phys. A 522 (2019) 69–79.
[24] K. Kulkarni, A. Pagourtzis, K. Potika, P. Potikas, D. Souliou, Community detection via neighborhood overlap and spanning tree computations, in: Algorithmic Aspects of Cloud Computing - 4th International Symposium, ALGOCLOUD 2018, Helsinki, Finland, August 20-21, 2018, Revised Selected Papers, in: Lecture Notes in Computer Science, vol. 11409, Springer, 2018, pp. 13–24.
[25] P. Potikas, D. Souliou, A. Pagourtzis, K. Potika, A parallel community detection algorithm based on spanning trees, in: Eighth IEEE International Conference on Big Data Computing Service and Applications, BigDataService 2022, Newark, CA, USA, August 15-18, 2022, IEEE, 2022, pp. 61–65.
[26] D. Souliou, P. Potikas, K. Potika, A. Pagourtzis, Weight assignment on edges towards improved community detection, in: Proceedings of the 23rd International Database Applications & Engineering Symposium, IDEAS 2019, Athens, Greece, June 10-12, 2019, ACM, 2019, pp. 3:1–3:5.
[27] P. Goyal, E. Ferrara, Graph embedding techniques, applications, and performance: A survey, Knowl.-Based Syst. 151 (2018) 78–94.
[28] H. Cai, V.W. Zheng, K.C.-C. Chang, A comprehensive survey of graph embedding: Problems, techniques, and applications, IEEE Trans. Knowl. Data Eng. 30 (9) (2018) 1616–1637.
[29] Z. Chen, X. Li, J. Bruna, Supervised community detection with line graph neural networks, 2017, arXiv preprint arXiv:1705.08415.
[30] Y. Zhang, Y. Xiong, Y. Ye, T. Liu, W. Wang, Y. Zhu, P.S. Yu, SEAL: Learning heuristics for community detection with generative adversarial networks, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 1103–1113.
[31] Y. Jia, Q. Zhang, W. Zhang, X. Wang, Communitygan: Community detection with generative adversarial nets, in: The World Wide Web Conference, 2019, pp. 784–794.
[32] J. Chen, Z. Gong, J. Mo, W. Wang, C. Wang, X. Dong, W. Liu, K. Wu, Self-training enhanced: Network embedding and overlapping community detection with adversarial learning, IEEE Trans. Neural Netw. Learn. Syst. (2021).
[33] C. Park, D. Kim, J. Han, H. Yu, Unsupervised attributed multiplex network embedding, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 5371–5378.
[34] B. Jing, C. Park, H. Tong, Hdmi: High-order deep multiplex infomax, in: Proceedings of the Web Conference 2021, 2021, pp. 2414–2424.
[35] X. Fu, J. Zhang, Z. Meng, I. King, Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding, in: Proceedings of the Web Conference 2020, 2020, pp. 2331–2341.
[36] X. Wang, N. Liu, H. Han, C. Shi, Self-supervised heterogeneous graph neural network with co-contrastive learning, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 1726–1736.
[37] L. Luo, Y. Fang, X. Cao, X. Zhang, W. Zhang, Detecting communities from heterogeneous graphs: A context path-based graph neural network model, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 1170–1180.
[38] J. Yang, J. Leskovec, Community-affiliation graph model for overlapping network community detection, in: 2012 IEEE 12th International Conference on Data Mining, IEEE, 2012, pp. 1170–1175.
[39] J. Yang, J. Leskovec, Overlapping community detection at scale: a nonnegative matrix factorization approach, in: Proceedings of the Sixth ACM International Conference on Web Search and Data Mining, 2013, pp. 587–596.
[40] M. Zhou, Infinite edge partition models for overlapping community detection and link prediction, in: Artificial Intelligence and Statistics, PMLR, 2015, pp. 1135–1143.
[41] A. Todeschini, X. Miscouridou, F. Caron, Exchangeable random measures for sparse and modular graphs with overlapping communities, J. R. Stat. Soc. Ser. B Stat. Methodol. 82 (2) (2020) 487–520.
[42] J. Mcauley, J. Leskovec, Discovering social circles in ego networks, ACM Trans. Knowl. Discov. Data (TKDD) 8 (1) (2014) 1–28.
[43] D. Herrmannova, P. Knoth, An analysis of the microsoft academic graph, D-lib Mag. 22 (9/10) (2016) 37.
[44] V. Nair, G.E. Hinton, Rectified linear units improve restricted boltzmann machines, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 807–814.
[45] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.
[46] O. Shchur, S. Günnemann, https://github.com/shchur/overlapping-community-detection, (2019).
[47] Y. Li, C. Sha, X. Huang, Y. Zhang, Community detection in attributed graphs: an embedding approach, in: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence, 2018, pp. 338–345.
[48] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, Advances Neural Inf. Process. Syst. 30 (2017).

Konstantinos Sismanis received his diploma from the School of Electrical and Computer Engineering at the National Technical University of Athens (NTUA) in 2009. He completed his Master degree in Data Science and Machine Learning at NTUA in 2022. Currently, is a PHD student in the field of algorithms and computer science at the NTUA. He has been working as a software developer and a data curator.

Petros Potikas is a member of the Laboratory Teaching and Research Staff in the School of Electrical and Computer Engineering of the National Technical University of Athens (NTUA) since 2014. Also, he taught at the Department of Informatics and Telecommunications of the National and Kapodistrian University of Athens (NKUA) and the Athens University of Economics and Business (AUEB). His research interests are focused on topics such as community detection, logic programming theory, deductive databases, and logic, as well as on computational theory and cryptography. He has authored articles in international scientific journals and conferences and has participated in research projects. Moreover, he has been in the committee of international conferences and a member of the organizing committee of conferences/workshops. In addition, he has been a member of the organizing committee of Panhellenic Informatics Competitions for students of the Hellenic Society of Scientists and Professionals in Informatics and Communications (EPY). He holds a B.Sc. degree from the Department of Informatics and Telecommunications of the NKUA and a Ph.D. from the School of Electrical and Computer Engineering of the NTUA. He was also a postdoctoral researcher at the City University of New York (2006). From March 2022 he is a member of the Plennary of the Hellenic Telecommunications & Post Commission (EETT).

Ms. Souliou Dora is a graduate of the Department of Electrical and Computer Engineering of the National Technical University of Athens and Doctor of Engineering of the same School. She works as Teachning and Research Associate at the School of Electrical And Computer Engineering NTUA.She has held academic positions at the University of Piraeus. She has authored two books on the foundations of computer science and Discrete Mathematics and has also contributed to the translation of two books in English. She has authored articles in international scientific journals and conferences and has participated in research projects. Her main contributions lie in the areas of Frequent Itemsets Mining, Clustering, Classification, Community Detection and Distributed/Parallel Algorithms.

Aris Pagourtzis is a Professor of Computer Science at the School of Electrical and Computer Engineering of the National Technical University of Athens (NTUA) and a Lead Researcher at Archimedes Unit, Athena Research Center. He holds a Diploma in Electrical Engineering and a Ph.D. in Electrical and Computer Engineering, both from the School of Electrical and Computer Engineering of NTUA. He has held academic positions at the University of Ioannina, the University of Liverpool, the ETH Zuerich, the University of Athens, and the Athens University of Economics and Business. He has co-authored four books on foundations of computer science, distributed computing and cryptography and about a hundred scientific publications. His main contributions lie in the areas of network algorithms, distributed computing, approximation algorithms, counting complexity and cryptography. He has served in the program and organizing committee of a number of theoretical computer science and cryptography conferences, and co-chaired CIAC 2017, FCT 2021, and several workshops. He has been invited several times as a visiting researcher to universities in Europe and the US. His research has been funded by grants from the US, UK, France, Switzerland, EU, and national resources.