2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

Machine Learning Approaches for Community Detection in Online Social Networks

Aurélio Ribeiro Costa, Rafael Henrique Nogalha de Lima, Célia Ghedini Ralha
University of Brasília (UnB), Campus Darcy Ribeiro, Brasília, Federal District, Brazil
arcosta@gmail.com, rafael.nogalha@aluno.unb.br, ghedini@unb.br

***Abstract*—Network analysis is responsible for taking insights or generating predictions from networked data sources where community detection finds chunks of related data in a network. The importance of community detection spans in different domain applications, from social network formation to protein interaction predictions. This work compares five state-of-the-art solutions to community detection using machine learning approaches in the context of online social networks - GraphGAN, SDNE, ComE, AC2CD, and CLARE. The experiments using real-world online social network datasets (Email-EU-Core, BlogCatalog3, Flickr) with micro-F1, macro-F1, and NMI scores demonstrate that graph neural networks and deep reinforcement learning approaches are better suited for the community detection task than others based on probabilistic or shallow networks.**

***Index Terms*—Deep Reinforcement Learning, Graph Neural Network, Network analysis**

I. Introduction

Social network platform usage shows growth statistics in the last decade, with the total number of users tripling from 970 million in 2010 to more than 4.48 billion in July 2021 [1]. The continuous growth in the number of people with internet access and smartphones increases the necessity to analyze online social network (OSN) data with computational methods. Thus, analyzing such data is becoming a relevant aspect of data and network science.

Tasks in network analysis include community detection (CD), node classification, and link prediction. CD views networks as graphs looking for which nodes are greatly attached to each other [2]. The CD challenge is an NP-complete problem demanding further investigation. Real-world OSNs exhibit significant irregularities in the degree of nodes and edge distribution, bringing out a high level of network organization. The inhomogeneity in the distribution of the edges among the nodes results from a high density of edges within special groups of nodes and low tightness between the nodes across different special groups. These special groups of subgraphs are known as communities or clusters within the network. The nodes that belong to the same community should have common interests or similarities. Disclosing these communities reveals the relationship between the structure and the functionality of the network [3]. Thus, CD in OSN can used in practical applications like recommendation systems or network summarization and privacy as described in [4].

The research of CD solutions seems mature, presenting classic methods to solve the problem, such as spectral clustering and statistical inference-based. However, we perceive a growing volume of publications towards the usage of high volume datasets [5] or machine learning (ML) to enhance the quality of scoring response [6], [7], [8].

Although the literature presents many ML and DL approaches for the CD problem in OSN, we note a comparison gap between such works. Thus, the contribution of this work is a comparative study of five approaches using public real-world datasets. We consider the experimental results might be valuable to the CD, OSN, and DL communities.

Section II presents the concepts related to CD, including Graph Convolutional Networks (GCN), Graph Neural Networks (GNN), Graph Attention Networks (GAT). Section III covers the CD ML approaches. Section IV displays the experimental method with the results in Section V. Finally, conclusion and future work are in Section VI.

II. Preliminaries

CD is a task of network analysis to find groups of tied nodes in a network. In a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{v_1, v_2, ...v_n\}$, $\mathcal{E} = \{e_1, e_2, ..., e_m\}$, a community $C = \{v_i, ..., v_j\}$ is typically defined as a group of nodes densely interconnected and the nodes are sparsely connected among communities. A community structure $\mathcal{C} = \{C_1, C_2, ..., C_n\}$ is a set of possible communities of nodes from a graph $\mathcal{G}$. Finding communities in a network helps to discover the internal organization of its nodes. CD is a valuable tool to characterize the entities that compose it (e.g., groups of people with shared interests, products with common properties) [9].

In the study by Schulman et al. [10], Proximal Policy Optimization (PPO) is introduced as a type of policy gradient method with several benefits compared to Trust Region Policy Optimization (TRPO). PPO is simpler to implement, more versatile, and exhibits better sample efficiency in empirical evaluations than TRPO.

Applying ML, one can undertake CD tasks from different perspectives, using DL, Gaussian models, generative adversarial models, GCN, and reinforcement learning (RL) [11]. In an RL general architecture, there are two main elements the agent and the environment. These elements continually interact from a starting state to a final one by finding the best action that guides the agent between each state. The environment is the *locus* where the agent operates and is composed by state. The agent is responsible for observing the environment, taking actions that change it, and receiving rewards for each action at each timestep. The reward received for each action taken is a stimulus that can be positive or negative, and the agent's final objective is to maximize the accumulated reward through the episodes. In [10], the Proximal Policy Optimization (PPO) is defined as a kind of policy gradient method and has some of the benefits of Trust Region Policy Optimization (TRPO). The PPO method is simpler to implement, more general, and has better sample complexity (empirically) than TRPO.

GNN is a specialization of neural network (NN), defined by [12], to deal with networked data. GNN implements a function $\tau(G, n) \in \mathbb{R}^m$ that maps a graph G and one of its n nodes into an m-dimensional Euclidean space. A GNN processes an input graph by a set of units, each unit corresponding to a graph node, linked according to the graph connectivity. The units update their states and exchange information until they reach a stable equilibrium. The output of a GNN is then computed locally at each node on the base of the unified state. The diffusion mechanism is constrained to ensure that a unique stable equilibrium always exists. Some specializations of GNNs are GCN and GAT.

GCN is a GNN architecture present by [13] successfully applied to the CD problem as described by [14] joining GCN to Markov random fields. GCN defines a spectral graph convolution by multiplying a graph signal with a spectral filter in the Fourier domain, using two graph convolution layers to derive a network embedding, and applying the *softmax* function to classify nodes into different categories. In training, the prior information on community memberships of a few nodes, network topology, and possibly node attributes are used to learn the NN weight parameters. GCN has an excellent global search capability with at least two drawbacks: it aims primarily at deriving a network embedding of the input data in the hidden layers of CNN, but such an embedding is not community-oriented, and it does not consider community properties, and can only obtain a relatively coarse community result since it lacks smoothness constraints to reinforce similar or nearby nodes to have compatible community labels.

GAT represents an attention-based NN architecture designed to perform node classification of graph-structured data. It computes the hidden representations of each node in the graph with a self-attention strategy. The attention architecture has interesting properties: the operation is efficient and parallelizable across node neighbor pairs, can apply to graph nodes having different degrees by specifying arbitrary weights to the neighbors, and the model is directly applicable to inductive learning problems, including tasks where the model has to generalize to completely unseen graphs [15].

III. Community Detection Approaches

The authors in [16] implement an architecture based on a network diffusion module to capture malicious behavior in networks. The model represents a message passing through the network. In [17], a semi-supervised CD solution is implemented using GNN to combine topological and context information. This approach models OSN connections as sparse matrices (SparseConv2D). The study undertaken by [18] also implemented a semi-supervised CD solution based on GCN and RL. GraphGAN is a graph representation framework proposed by [19] to learn node embeddings based on edge-wised information. The main objective is a method to represent nodes in a low-dimension vector space, it implements the GAN approach with generative and discriminative thinking for graph representation learning. ComE [20] defines a technique that relies on node and community embedding for learning graph embeddings. SDNE [21] is a method that depends on a deep model using *Laplacian eigenmaps*.

1) Generative Adversarial Approach: GraphGAN is a graph representation framework that implements the GAN approach with generative and discriminative thinking for graph representation learning [19].

The GAN is formulated in GraphGAN in the following terms. Let $G = (\mathcal{V}, \mathcal{E})$ be a given graph, where $\mathcal{V} = \{v_1, ..., v_V\}$ represents the set of vertices and $\mathcal{E} = \{e_{ij}\}_{i,j=1}^{V}$ represents the set of edges. For a given vertex v_c, $\mathcal{N}(v_c)$ is defined as the set of vertices directly connected to v_c, the size of which is typically much smaller than the total number of vertices V. The conditional probability $p_{true}(v|v_c)$ denotes the underlying true connectivity distribution for vertex v_c, which reflects v_c's connectivity preference distribution over all other vertices in $\mathcal{V}$. From this point of view, $\mathcal{N}(v_c)$ can be seen as a set of observed samples drawn from $p_{true}(v|v_c)$.

GraphGAN trains two models during the learning process. The Generator $G(v|v_c)$ tries to fit the underlying true connectivity distribution $p_{true}(v|v_c)$ as much as possible and generates the most likely vertices to connect with v_c. The Discriminator $D(v, v_c)$ tries to distinguish well-connected vertex pairs from ill-connected ones and calculates the probability of whether an edge exists between v and v_c. In GraphGAN, the generator G and the discriminator D are two players in a *minimax* game. The generator produces the most indistinguishable "fake" vertices under guidance provided by the discriminator. The discriminator draws a clear line between the ground truth and "counterfeits" to avoid being fooled by the generator. Competition drives them to improve their capability until the generator is indistinguishable from the proper connectivity distribution. Figure 1 presents the GraphGAN architecture and the evolution of an execution highlighting the role of the Generator G and Discriminator D.

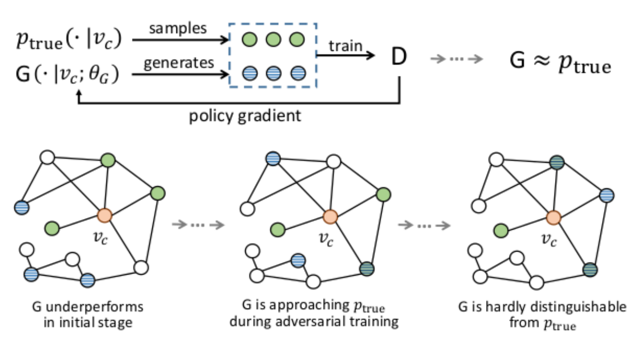

Fig. 1. GraphGAN architecture [19].

2) Deep Network Embedding Approach: Structural deep network embedding (SDNE) [21] is a semi-supervised deep network model that exploits the first and second-order proximity to preserve the network structure. The second-order proximity is used by the unsupervised component to capture the global network structure. The first-order proximity is used as the supervised information in the supervised component to preserve the local network structure. By jointly optimizing them in the semi-supervised deep model, this method can retain both the local and global network structure and is robust to sparse networks. The SDNE architecture overview is presented in Figure 2, each vertex embedded using an unsupervised approach and then pair-wised using a supervised component based on Laplacian Eigenmaps.

3) Gaussian Mixture Approach: ComE [20] defines a method that relies on the node and community embedding for learning graph embeddings in a closed loop among community embedding, community detection, and node embedding. On the one hand, node embedding can help improve community detection, which outputs good communities for fitting better community embedding. On the other hand, one can use community embedding to optimize the node embedding by introducing community-aware high-order proximity. ComE closed loop for learning includes community detection, community embedding, and node embedding.

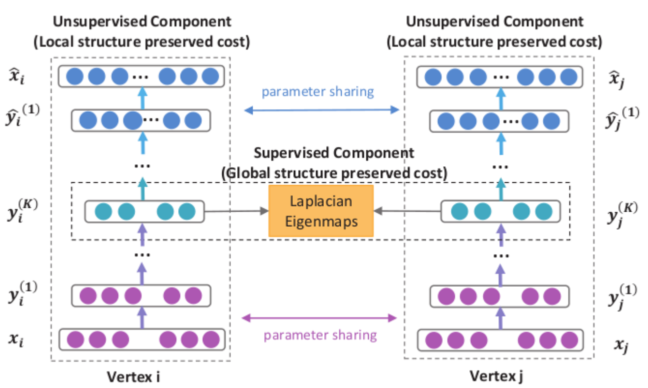

Fig. 2. SDNE architecture [21].

4) Actor-Critic Approach: The AC2CD architecture [22] consists of a DRL approach based on GAT to find the optimal community structure in a dynamic social network. AC2CD uses the message-passing feature of GAT as an element to propagate the label for each community. The RL method chosen is Actor-Critic with PPO in the clipped version and generalized advantage estimation (GAE) to compute the surrogate function of the policy gradient. According to [23], PPO performs the best in terms of profit and loss, training time, and quantity of data needed for training compared to Q-learning and deep Q-learning. It is worth noting that the proposed architecture is extensible to other implementations.

Figure 3 shows the AC2CD architecture overview highlighting the Actor-Critic components in gray. Inside these components, there are the GAT layers. Each node of the input graph is embedded in a vector with 256 positions using the Node2Vector strategy [24] resulting in a matrix M_{265xn} where n is the number of vertexes. Figure 3 presents the interaction between the agent (light gray), environment (blue), and internal aspects of these entities. AC2CD was developed to work with dynamic networks, however, it presented relevant results for CD in static networks as well.

The learning process begins with the agent observing the environment for changes in the network. Changes can be the creation or exclusion of an edge or a node. Once a change in the state of the network is observed, the Actor chooses the best community structure, and the Critic computes the modularity density for the community structure. The difference between the value issued by the Critic and the ground truth corresponds to the TD Error.

The RL action space represents the possible assignment combination between node and community. The reward function is implemented as the modularity density for the community structure of each network snapshot. A positive reward indicates an improvement in the modularity density, a negative otherwise.

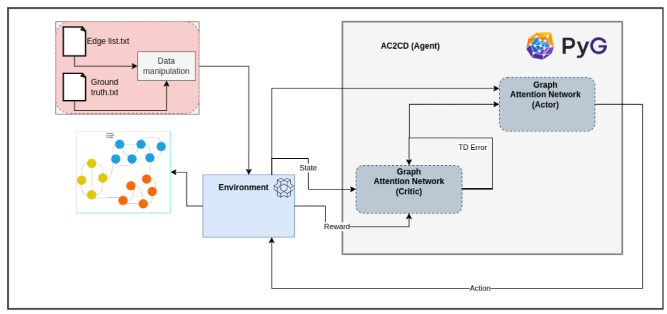

Fig. 3. AC2CD architecture [22].

5) Graph Convolution Network Approach: The study undertaken by [18] presented CLARE, a framework consisting of two key components, Community Locator and Community Rewriter. The community locator can quickly locate potential communities by seeking subgraphs similar to the training ones. Specifically, CLARE encodes communities into vectors, measures the similarities between communities in the latent space, and then discovers candidates based on the similarities with the nearest neighbors matching strategy. The community rewriter further adjusts those candidate communities by introducing global structural patterns. CLARE frames such refinement process as a DRL task and optimizes this process via policy gradient. For located communities, the rewriter provides two actions: adding outer nodes or dropping existing nodes, thus refining their structures flexibly and intelligently.

The CLARE core is a GCN that learns to encode nodes and community representations. Figure 4 presents the CLARE architecture emphasizing the two main components, at left the Community Locator and right the Community Rewriter. The Community Rewriter implements DRL, where the state is a predicted community united with its outer boundary. The action is a combination of $(a_t^{exclude}, a_t^{expand})$, i.e., at each time t one node can be excluded and another included in a community C_t. The reward signal is taken directly from the F1 score of the community structure.

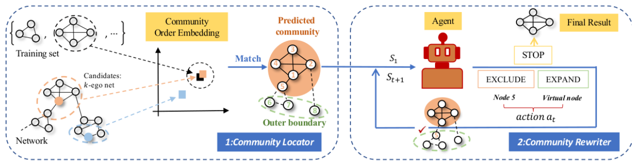

Fig. 4. CLARE architecture [18].

IV. Experimental Method

The empirical experimental setup includes eight executions of each ML approach using the available implementations in GitHub. The objective is to attenuate the non-determinism effect of the ML approaches by evaluating the influence of the mean maximum value. Such a process is a better alternative than only setting all seeds to a fixed value or getting just the top resulting value. The experiments use a computer with a CPU Intel® Xeon Gold 5220R with 48 cores, 187GB of RAM, and two GPU NVIDIA® V100S. The operating system is Ubuntu with the Conda project external libraries.[1] Each dataset represents a social network where a node is a person, and an edge is an interaction between them. The dataset partitioning follows the original works of GraphGAN, SDNE, ComE, and AC2CD as presented in Table I. We used labeled nodes for the Email-EU-Core and Flickr, 1%, 3%, 6%, and 9%, and BlogCatalog, 10%, 30%, 60%,90%. The CLARE partitioning is based on the number of communities using the same dataset percentage of labeled nodes.

TABLE I
Datasets characterization.

Name	# Nodes	# Edges	# Communities
Email-EU-Core	1,005	25,571	42
BlogCatalog3	10,312	333,983	39
Flickr	80,513	5,899,882	195

The Email-EU-Core is generated using email data from a large European research institution, with only nodes from inside the institution available on the Snap Project Home page.[2] Each edge (u, v) tells that a person u sent an email to v. Figure 5 illustrates the community network topology as presented by [25]. Figure 6 presents the community distribution of users. It is worth to note the community topology is homogeneous, but the distribution is not since it presents only two communities among 42 with more than 80 users.

BlogCatalog3 is a social blog directory where each edge represents a friendship between two bloggers, available at the Arizona State University repository.[3] In [27], a novel method is proposed for multi-task learning-based network embedding using the BlogCatalog3 dataset as presented in the network topology of Figure 7. The figure shows a homogeneous clustering, although it is not. Figure 8 presents the community distribution of bloggers showing the long tail aspect, where only one community has more than 1000 members.

[1]Conda Project available at https://docs.conda.io/en/latest/
[2]https://snap.stanford.edu/data/email-Eu-core.html
[3]http://datasets.syr.edu/pages/datasets.html

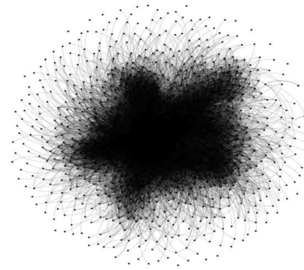

Fig. 5. Email-EU-Core topology.

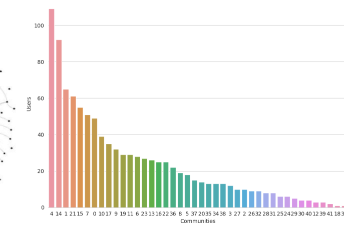

Fig. 6. Email-EU-Core community distribution.

Flickr is built by forming edges between images and sharing standard metadata from the Flickr platform. Edges are formed between images from the exact location, submitted to the gallery, group, or set, images sharing common tags, and images taken by friends, among other attributes. The Flicker dataset is available at the Arizona State University repository. The authors in [28] use Flickr to validate their peer prediction-based trustworthy service rating system for social networks, as presented by the complex network topology of Figure 9. Figure 10 shows the community distribution of users with an accentuated long-tail aspect, where one community aggregates 13,700 users, one with 6,000, and the other communities have fewer than 500.

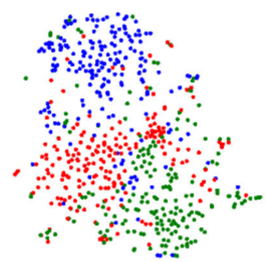

Fig. 7. BlogCatalog3 topology.

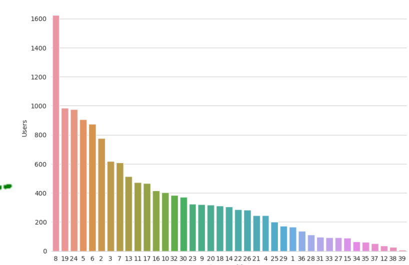

Fig. 8. BlogCatalog3 community distribution.

Fig. 9. Flickr topology.

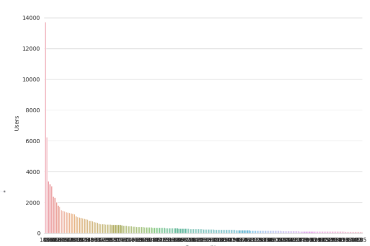

Fig. 10. Flickr community distribution.

Figure 11 presents the distribution of community users for the datasets. Note that Flickr has a heterogeneous distribution enforced by the topology (Figure 9). The Emai-EU-Core and BlogCatalog3 communities have regular aspects verified by the network topologies (Figures 5 and 7).

Metrics: The performance comparison of the presented approaches uses the F1-score [29] and normalized mutual information (NMI) [30]. F1-score is the harmonic measure of precision P, and recall R, and n represents the number of categories. Macro-averaged F1-score (Macro-F1) and

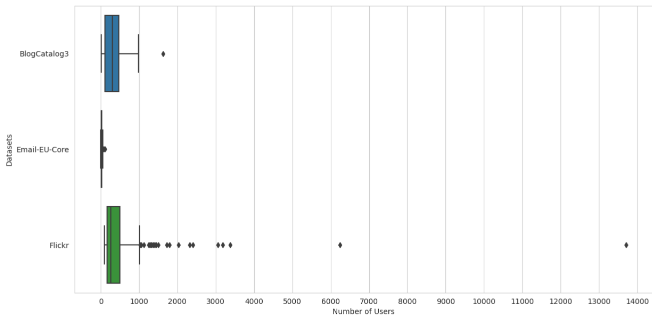

Fig. 11. Comparative box plot of dataset community distribution.

micro-averaged F1-score (Micro-F1) aggregate the F1-score measuring the performance of a classifier in a multi-label categorization. Macro-F1 is the arithmetic means of F1-scores of all categories:

$$\mathcal{F}_1 = \frac{1}{n}\sum_x F1_x = \frac{1}{n}\sum_x \frac{2P_x R_x}{P_x + R_x}.$$

Micro-F1 is the harmonic mean of the micro-precision and micro-recall computed with the sum of true positives, false positives, and false negatives values:

$$\mathbb{F}_1 = \frac{2\bar{P}\bar{R}}{\bar{P}+\bar{R}} = 2\frac{(\frac{1}{n}\sum_x P_x)(\frac{1}{n}\sum_x R_x)}{\frac{1}{n}\sum_x P_x + \frac{1}{n}\sum_x R_x}.$$

Given a reference community structure A and a detected community structure B, NMI computes the overlapping nodes in A and B. NMI approximates the marginal probability of a randomly selected node being in the community a and b by $P_A(a) = \frac{n_a}{n}$ and $P_B(b) = \frac{n_b}{n}$, where n_a and n_b denote community size of a and b. Moreover, $P_{AB}(a,b) = \frac{n_{ab}}{n}$, where n_{ab} is the number of nodes that are both in the community of partition A and the group b of partition B.

Macro-F1 and Micro-F1 represent good evaluation options to CD. However, they have a lack of sensibility in the permutation of communities that can be addressed by the MNI.

V. Results and Discussion

The comparative study undertook to show a little improvement in the NMI scores for approaches based on GNN's, with the best mean for the AC2CD solution and the best stability for CLARE using GCN, as presented in Figures 19 and 18. The solution based on Deep Learning (DL), ComE, performed worst on average, this issue may indicate a weakness in classical DL approaches to cope with more sophisticated data structures such as graphs.

This section presents the accuracy results considering the datasets evaluated through the F1 and NMI metrics. As cited by [31], 98% of nodes are concentrated in the largest community. Thus, the results of the Macro-F1 score resemble the respectful performance of CLARE to learn such concentrated edges in one node (Figure 12). However, the results of the Micro-F1 score in Figure 13 are not so good to capture such regularity in the density distribution of communities.

The results of Macro-F1 present good performance with CLARE to learn the community structure as the network is composed of a homogenous community distribution (Figure 14). However, the Micro-F1 in Figure 15 presents no good results of the GCN since this metric computes the sensitivity of the difference among density distribution of communities. Additionally, the results of Macro and Micro-F1 scores of the Flickr community resemble the GCN difficulty in detecting the community structure (Figures 16 and 17). Using the same dataset (Flickr), the actor-critic approach presents the best results by capturing such complex community structure. We highlight that such a complexity associated with the Micro-F1 metric sensitivity resembles the importance of more robust approaches to CD.

The results using the NMI score generate box plot graphs (Figures 18 and 19). Figures describe the mean, median, and standard deviation for the executions of each ML approach. Note that the GNN-based approaches (i.e., GraphGAN, AC2CD, CLARE) present on average superior performance than others. The GraphGAN presents a symmetric profile related to the medium, and the SDNE has a high median where most results are near the maximum NMI (Figure 18). The ComE with a single DL approach presents a low median where most results are near the minimum NMI value. The AC2CD presents the best NMI results compared to GraphGAN, SDNE, ComE, and CLARE. Nevertheless, the standard deviation of CLARE is low compared to the other approaches with a stable profile.

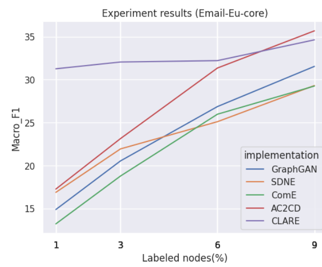

Fig. 12. Email-EU-Core Macro-F1.

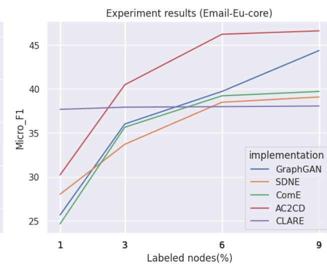

Fig. 13. Email-EU-Core Micro-F1.

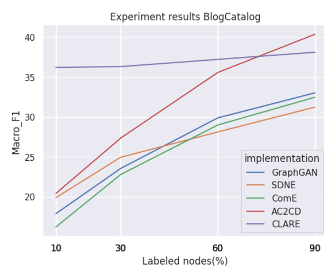

Fig. 14. BlogCatalog Macro-F1.

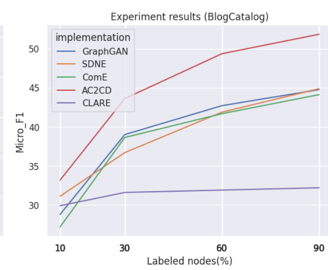

Fig. 15. BlogCatalog Micro-F1.

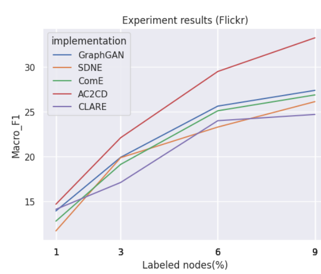

Fig. 16. Flickr Macro-F1.

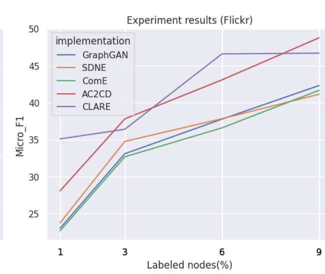

Fig. 17. Flickr Micro-F1.

VI. Conclusion

This work presents a comparative study of five ML approaches for CD problems in OSN. The approaches using GNN (i.e., GraphGAN, AC2CD, CLARE) seem to better adapt to the distinct dataset community network topology (Section IV). The AC2CD presents the best results with BlogCatalog and Email-EU-Core datasets compared to GraphGAN, SDNE, ComE, and CLARE using Micro-F1 (Figures 13 and 15), but CLARE is better using Macro-F1 (Figure 12 and 14).

The AC2CD presents the best results with the Flickr dataset compared to GraphGAN, SDNE, ComE, and CLARE using Macro-F1

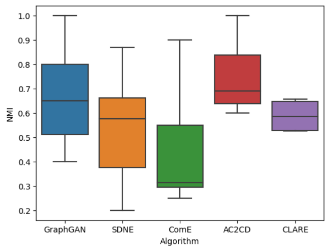

Fig. 18. NMI scores for Email-EU-Core.

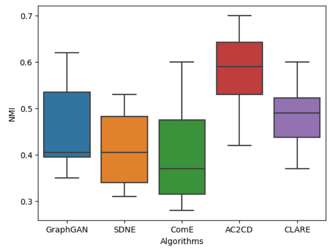

Fig. 19. NMI scores for BlogCatalog.

(Figure 16), but competitive to CLARE using Micro-F1 (Figure 17). The Flickr dataset presents an asymmetric distribution with the most complex network topology (Figure 9) highlighted by the different sizes of circles, representing the density of edges on each node. Flickr also exhibits an accentuated long-tail aspect (Figure 10). The AC2CD with the Email-EU-Core dataset presents the best stability results with adequate accuracy (Figure 18), with the BlogCatalog (Figure 19) achieving the top score and the best mean MNI value. The DL techniques handling well high-dimensional graph data demonstrated superior performance over classic methods [32].

Future work include aspects related to the diversity and complexity of the CD problem. As many approaches are constrained to heuristic solutions, there is still space for new ML-improved strategies. Also, formal verification to validate the solution's correctness is necessary. A valuable research investigation to evaluate proposals include dynamic OSN with reliable ground truth since data volume is growing.

REFERENCES

[1] Brian Dean. Social network usage & growth statistics: How many people use social media in 2022? Available at https://backlinko.com/social-media-users. Accessed on: Oct 20, 2022.

[2] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

[3] Annapurna Jonnalagadda and Lakshmanan Kuppusamy. A survey on game theoretic models for community detection in social networks. *Social Network Analysis and Mining*, 6(1):83, 2016.

[4] Arzum Karataş and Serap Şahin. Application areas of community detection: A review. In *IEEE International congress on big data, deep learning and fighting cyber terrorism (IBIGDELFT)*, pages 65–70, 2018.

[5] Rahil Sharma and Suely Oliveira. Community detection algorithm for big social networks using hybrid architecture. *Big Data Research*, 10:44–52, 2017.

[6] Yao Zhang, Yun Xiong, Yun Ye, Tengfei Liu, Weiqiang Wang, et al. SEAL: Learning heuristics for community detection with generative adversarial networks. In *Proc. of 26th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 1103–1113, 2020.

[7] Ali Mohammad Saghiri, M Daliri Khomami, and Mohammad Reza Meybodi. *Intelligent Random Walk: An Approach Based on Learning Automata*, chapter Random Walk Algorithms: Definitions, Weaknesses, and Learning Automata-Based Approach, pages 1–7. Springer, 2019.

[8] Eduardo C. Paim, Ana L.C. Bazzan, and Camelia Chira. Detecting communities in networks: a decentralized approach based on multiagent reinforcement learning. In *Proc. of IEEE Symp. Series on Comp. Intelligence*, pages 2225–2232, 2020.

[9] Ferran Parés, Dario Garcia Gasulla, Armand Vilalta, Jonatan Moreno, Eduard Ayguadé, Jesús Labarta, Ulises Cortés, and Toyotaro Suzumura. Fluid communities: A competitive, scalable and diverse community detection algorithm. In *Proc. of Int. Conf. on Complex Networks & Their Applications*, pages 229–240, 2017.

[10] John Schulman, Filip Wolski, Prafulla Dhariwal, et al. Proximal policy optimization algorithms. *arXiv:1707.06347*, 2017.

[11] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[12] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

[13] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv:1609.02907*, 2016.

[14] Di Jin, Ziyang Liu, Weihao Li, Dongxiao He, and Weixiong Zhang. Graph convolutional networks meet markov random fields: Semi-supervised community detection in attribute networks. In *Proc. of AAAI Conf. on Artificial Intelligence*, pages 152–159, 2019.

[15] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv:1710.10903*, 2017.

[16] Jose N. Paredes, Gerardo I. Simari, Maria Vanina Martinez, and Marcelo A Falappa. Netder: An architecture for reasoning about malicious behavior. *Information Systems Frontiers*, pages 1–17, 2020.

[17] Aniello De Santo, Antonio Galli, Vincenzo Moscato, et al. A deep learning approach for semi-supervised community detection in online social networks. *Knowledge-Based Systems*, 229:107345, 2021.

[18] Xixi Wu, Yun Xiong, Yao Zhang, Yizhu Jiao, Caihua Shan, Yiheng Sun, et al. CLARE: A semi-supervised community detection algorithm. In *Proc. of 28th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 2059–2069, 2022.

[19] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. GraphGAN: Graph representation learning with generative adversarial nets. In *Proc. of 32nd AAAI Conf. on Artificial Intelligence*, pages 2508–2515, 2018.

[20] Sandro Cavallari, Vincent W. Zheng, Hongyun Cai, et al. Learning community embedding with community detection and node embedding on graphs. In *Proc. of ACM on Conference on Information and Knowledge Management*, page 377–386, 2017.

[21] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *Proc. of 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, page 1225–1234. ACM, 2016.

[22] Aurélio R. Costa and Célia G. Ralha. AC2CD: An actor-critic architecture for community detection in dynamic social networks. *Knowledge-Based Systems*, 261:110202, 2023.

[23] Jiayi Du, Muyang Jin, Petter N Kolm, Gordon Ritter, Yixuan Wang, and Bofei Zhang. Deep reinforcement learning for option replication and hedging. *The Journal of Financial Data Science*, 2(4):44–57, 2020.

[24] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proc. of 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 855–864, 2016.

[25] A. Bharali. An analysis of email-eu-core network. *International Journal of Scientific Research in Mathematical and Statistical Sciences*, 5:100–104, 8 2018.

[26] R. Zafarani and H. Liu. Social computing data repository at arizona state university. Available at http://socialcomputing.asu.edu, 2009. Accessed on: Jan 16, 2022.

[27] Shanfeng Wang, Qixiang Wang, and Maoguo Gong. Multi-task learning based network embedding. *Frontiers in Neuroscience*, 13:1387, 2020.

[28] Jun Du, Erol Gelenbe, Chunxiao Jiang, Haijun Zhang, Yong Ren, and H Vincent Poor. Peer prediction-based trustworthiness evaluation and trustworthy service rating in social networks. *IEEE Transactions on Information Forensics and Security*, 14(6):1582–1594, 2018.

[29] Kanae Takahashi, Kouji Yamamoto, Aya Kuchiba, and Tatsuki Koyama. Confidence interval for micro-averaged F_1 and macro-averaged F_1 scores. *Applied Intelligence*, 52(5):4961–4972, 2021.

[30] Pan Zhang. Evaluating accuracy of community detection using the relative normalized mutual information. *Journal of Statistical Mechanics: Theory and Experiment*, 2015(11):P11006, 2015.

[31] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. Available at http://snap.stanford.edu/data., 2014. Accessed on: Fev 05, 2020.

[32] Fanzhen Liu, Shan Xue, Jia Wu, Chuan Zhou, Wenbin Hu, Cecile Paris, Surya Nepal, et al. Deep learning for community detection: Progress, challenges and opportunities. In Christian Bessiere, editor, *Proc. of 29th Int. Joint Conf. on Artificial Intelligence*, pages 4981–4987, 2020.