Would I regret later joining this Community ? Using temporal neighborhood information for community retention in a game theoretic community detection framework

Prateek Chanda[1] **and Susanta Chakraborty** [2] **and Malay Kule** [3]
[1]Microsoft Research Lab, Bangalore, India
prateekkol21@gmail.com (v-pratec@microsoft.com)
[2]Indian Institute of Engineering Science and Technology, Shibpur Institute, Botanical Garden Area, Howrah, West Bengal 711103, India
sc@cs.iiests.ac.in
[3] Indian Institute of Engineering Science and Technology, Shibpur Institute, Botanical Garden Area, Howrah, West Bengal 711103, India
malay.cs@faculty.iiests.ac.in

Abstract

Detection of relevant and meaningful communities from any social network is always a constant research challenge. Considering large social networks existing approaches often fail to capture perfect community partition or fail to converge fast. Inspired by non-myopic reinforcement learning domain, in this paper, we have proposed a novel game theory based community detection algorithm which considers community retention based on temporal information as part of a node's strategy (i.e. for a node whether their current assigned community is more profitable based upon future possibilities, or should they switch at the moment), thereby minimizing cross-community false node switches. The proposed method has the following properties (a) considers community retention for a node in the network when it considers switching to another community, (b) achieves significantly better/comparable performance w.r.t baselines in terms of metrics such as quality of partition on various real world datasets,(c) faster convergence w.r.t traditional game theoretic approach which only considers short-term utility by minimizing cross-community node switches (d) utility to interpret at each iteration for any particular node's strategy.

Index terms - game theory, temporality, fast convergence, modularity, probabilistic sampling, community retention

Introduction

Community is a well-defined structure in any network which refers to group of nodes that have more edge connections among themselves than the edges that connect them to the rest of the network. From the last two decades, discovery of communities from any complex network has been a well-researched problem with applications including social network analysis (Ji et al. 2020), online recommendation system (Ying et al. 2013), biological networks etc.

Girvan–Newman algorithm (Girvan and Newman 2002b) was first proposed for finding communities in a biological network but later widely accepted to apply in other real-world problems of community detection. In the recent years, different techniques like hierarchical clustering (Cheng et al. 2012), optimization-based algorithms and multiobjective evolutionary algorithm (Li, Liu, and Wu 2017) have been suggested to detect community structures.

Here we focus mainly on traditional game-theoretic models (Basu and Maulik 2015), where individual nodes in a graph participate in joining communities by choosing the best strategy from the strategy space. We hence show how our current methodology relates to them and in some cases overcome previous drawbacks.

In this paper, we aim to propose a game theoretic based community detection method to achieve a stable community structure through considering not only local neighborhood information but also temporal relations into account.

Our main contributions are as follows:

- *Fast Convergence with same community structure* We show our proposed method converges faster as compared to traditional game theoretic approach, having similar community structures.
- *Theoretical Guarantees* : In subsequent theorems we show how pure Nash Equilibria holds for our model along with faster convergence.
- *Real world Applicability and Effectiveness* We evaluate our model on real-world social network datasets and show case how community detection via temporal knowledge takes place through experiments.
- *Interpretability* We showcase how our method is easily interpretable w.r.t any particular node (i.e. why a node choses a particular community at some itera-

tion) as compared to neural models which are often hard to interpret.

The remaining of this paper is structured as follows. The existing works on the related topics are discussed in Section II. Preliminaries related to the community detection methods have been described in Section III. Section IV contains detailed walkthrough of our proposed community detection algorithm. The results are discussed in Section V. Conclusions from the study are contained in Section VI.

Motivation for Community Detection in Operation Research

Community Detection has a wide variety of applications in relation to operation research including detecting fraud detection of similar group of fraudsters in online networks (Li et al. 2021) or in healthcare (Gangopadhyay and Chen 2016), or in e-commerce fields where customer quality needs to be gauged (Arab 2021) where user communities need to be discovered and changes should be made towards the low engaging customers. Even in e-commerce platforms where the business profit lies in engaging users with more personalized recommendation of products, community detection is widely used (Feng et al. 2015). Hence it is imperative to solve the problem of community detection in large networks with less time so that it can be scaled to business within their applications.

Background and Related Work

Most of the methods proposed in the recent years have focused primarily towards modularity optimization problem in networks, (Newman 2004) being one of the first greedy approaches to modularity optimisation. Recent work proposes modularity optimization that follows an agglomerative technique, with each node as a distinct community, which can then be merged iteratively based on the modularity gain.

(McSweeney, Mehrotra, and Oh 2012) proposes a similar setting where there consists of several iterations consisting of the same node joining a particular community and then again switching to another community successively, thus leading to a longer time for convergence. Recent methods like (Jiang and Xu 2015) uses labelling mechanism, whereby at each iteration a node is randomly chosen which then performs its corresponding profitable action and its then labelled, not to be sampled in the later portions. While this clearly indicates a faster run time execution, however these might not lead to a stable Nash equilibrium, discussed later. While most methods fundamentally rely on modularity calculation through network dimensions, some including (Bu et al. 2018) uses similarity methods like neighborhood or cosine similarity to establish edge connections. Additionally (Bu et al. 2018) follows a non-cooperative game theoretic approach where partitions are formed in parallel following fast heuristics.

Preliminaries

In this section, some preliminary concepts necessary for our proposed approach are discussed.

Symbols	Definition
A	Real symmetric n x n adjacency matrix representing the network.
δ	Kronecker delta function. $\delta_{ij} = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases}$
Q	Modularity Function
γ	Resolution Parameter.
$q(u, v, C)$	The payoff function of node u when node v joins its community C
$\phi(u)$	Community label assigned to node u
m	Total edges in the network
k_i	Degree of node i
$\mu(i, C)$	Utility of node i to join community C
C_x	Community labelled as x
$\mu(C_x, C_y)$	Utility of C_x when it wants to merge with C_y

Concept of Modularity The fundamental goal of community detection algorithms is to partition a network into separate communities, where modularity represents the quality of a particular partition in the network.

Considering an undirected graph of n nodes, A represents a real symmetric n x n dimensional adjacency matrix, where $A_{ij} = 1$ when nodes i and j are connected else $A_{ij} = 0$. Let us also denote the degree of a node i by d_i.

Considering a partition of the network into ϕ groups, the number of edges that fall in each group is equal to $\sum_{ij} A_{ij}\delta_{\phi_i}\delta_{\phi_j}$ where δ_{ij} is the Kronecker delta function. The concept of modularity therefore is proposed as the difference between the current number and the expected number of edges placed randomly within the network. If P_{ij} be the probability that nodes i and j are connected, then the modularity value given by the Reichardt & Born modularity function can be written as follows:

$$Q(\gamma) = \frac{1}{2m}\sum_{i,j}(A_{ij} - \gamma\frac{k_i.k_j}{2m})\delta(C_i, C_j)$$

where γ is the resolution parameter, in our case $\gamma = 1$

Game Theoretic Framework Considering a setting with n players denoted as $\mathrm{N} = 1, 2, 3, \ldots n$, for each i_{th} player in the game, there is a set of strategies that player takes in order to maximize its own payoff in the game. Let's say the set of such strategies be S where

S = $S_1 \times S_2 \ldots S_n$, combination of all the strategies of the individual players.

In a network setting, we can consider each such node as an agent that tries to maximize its own payoff, while joining a particular community formed within the network.

Corresponding strategies for node u include

- Join a particular community. $C \implies C \cup u$
- Leave a particular community $C \implies C - u$
- Switch communities C_1 and C_2 : $C_1 - u \implies C_2 \cup u$

As per Fig. 1 for a node i with degree d_i, the total number of possible actions would be $3 * d_i$ operations where at each branch there are 3 options to choose from.

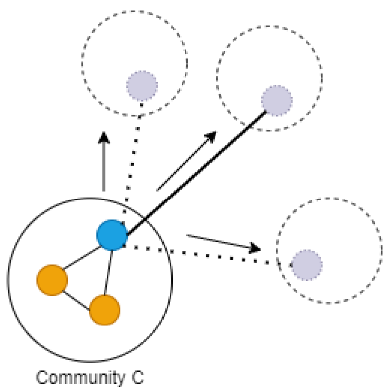


Figure 1: Strategy Set - The blue node can take any action along its degree/branches, *Arrow indicating joining external clusters*

Goal of such Strategies The goal for a node is thus to choose the maximum payoff strategy, thereby maximizing the modularity of the network, leading to a stable partition, in subsequent iterations.

Definition 1. ***Payoff function.*** *As per Marginal Payoff under Section 3 Definition 1. (McSweeney, Mehrotra, and Oh 2012) The payoff function of a node u for node v is given as:*

$$q(u, v, C) = \mu(u, C \cup v) - \mu(u, C - v)$$

where $\mu(i, C)$ is $\sum_{j \in C}(A_{ij} - \frac{k_i.k_j}{2m})\delta_{C,C_i}$

The payoff function indicates how useful a particular action will be for a given agent in a game setting. In the community formation domain, payoff indicates whether joining or leaving or switching a community is profitable. From here we use the notation for denoting the payoff function as utility function interchangeably.

Properties of Utility Function

Here we consider some of the desirable properties of payoff functions in general. We follow definitions **2** & **3** of Section *III* from (McSweeney, Mehrotra, and Oh 2012) regarding the following properties.

Definition 2. ***Symmetric*** *Let q be a payoff function and u and v be two different nodes. Then, q is symmetric iff*

$$\forall_{S \in N}(q(u, v, C) = q(v, u, C))$$

Definition 3. ***Aditively-Separable*** *q is additively-separable iff* $\mu(u, S) = \sum_{v \in S} \delta(u, v, S)$, *The property of additively-separable states that a node u's payoff for a community S is the sum of the marginal payoffs over the members of S.*

We now introduce here the concept of Nash equilibrium in the context of reaching a solution to finding optimal communities using the proposed payoff function. The motivation for proposing Nash equilibrium is due to the fact that at the end of each iteration of our community detection model we want to find an optimal community partition (each node is in the best community relative to other possible alternative communities). Since each of these nodes can take any of the strategies independently as self-rational agents, the stability of partitions formed is often decided whether the partitions have reached a Nash equilibria. We thus formally define the same

Definition 4. ***Nash Equilibria*** *A partition ϕ is a Nash Equilibrium for (N, μ) iff for all nodes u:*

$$q(u, \phi(u)) \geq q(u, C \cup u) \quad \forall\, C \in \phi, C \neq \phi(u)$$

which indicates for any partition, corresponding nodes' payoffs wont improve any further.

Proposition 1. *Every network will have at least one Nash equilibrium provided a symmetric and additively-separable payoff function is used.*

This proposition is proven in (Bogomolnaia, Jackson et al. 2002).

We thus show in further discussions how our proposed payoff function would follow the utility function properties, thus achieving Nash Equilibria.

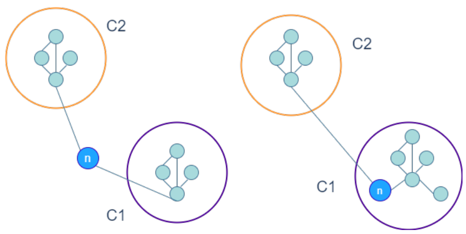
Proposed Methodology

Temporal Based Game Theoretic Community Detection

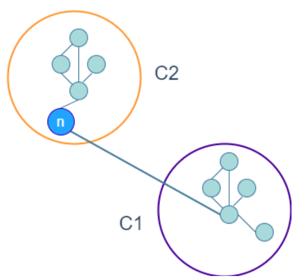
While an agent's decision from is based upon local neighborhood information in the network, despite being profitable at current timepoint, it might not be profitable later and a node might have to choose a community it initially rejected. We therefore consider the improvement of an agent's decision not only based on its local information but also through its temporal information via community retention, i.e. if a node has a higher chance of retention in the community it initially chose.

As a result, we aim at reducing the sampling space for the next iteration, by assigning a lower sampling probability to the node. We consider this since it has already made a wise decision, considering whether it would be retained in a community for long during its initial choice.

Defining Temporal Relation As per Fig: 2a Initially a node takes its action based on maximizing its utility/payoff function and decides to join a community, say C_1. However, after some iteration, it finds that the other communities its connected to via its branches are much more profitable, lets say C_2. So there is a corresponding switch action. Hence, as per the node takes a bad decision initially based on local motivations for joining C1 instead of C2, but later switches community.



(a) Node n sees joining C_1 is favourable in the first iteration



(b) Node regrets in later iterations, now chooses C_2 which seem more profitable

Figure 2: A case of switching communities

Push and Pull Strategies While the general utility function indicates how profitable an agent's action will be while joining a particular community at the current instant, it does not consider whether a node will regret the current choice based on local environment and have to switch later on. We therefore propose the idea of push and pull strategies.

Pull strategy involves the following. Suppose in 4b a node i decides to join a community C, where the corresponding community has nodes which has external neighbors as O_1, O_2. If for a particular node belonging to the external neighbors if there are a lot of connections to them, then there is a high probability of that node getting pulled into the community in some time later, thus increasing more number of internal connections in future, resulting in higher modularity of the community as a whole.

So if we consider an average pull which actually indicates the expected number of nodes that will be pulled into the community, then the average number of internal connections in the future timestamp will increase thereby forcing N_i to retain inside the community instead of switching.

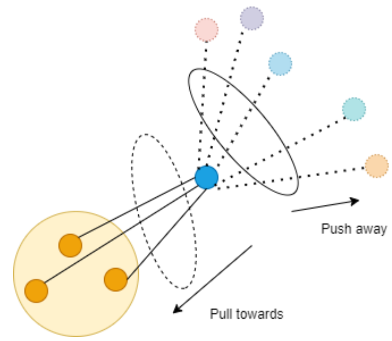
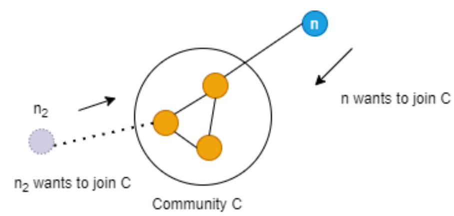


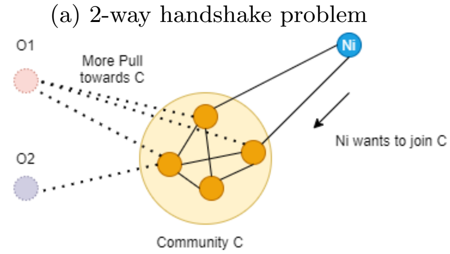
Figure 3: Community Retention : Push & Pull Strategy

The above claim can also be formally proved since is a 2-way handshake mechanism. Fig 4a

Suppose n decides to join C which has n_2 as an external neighbor. The claim we are making n should consider its decision based on the face that n_2 can be pulled into C implies that n_2 will join C among its other best decisions which in-fact is contributed to also if n joins C. Hence this becomes a good strategy.



(a) 2-way handshake problem



(b) Ni's decision based on if O_1 would join

Figure 4: Validity of Push-Pull Strategy

Push Strategy However taking a decision based on this can also be detrimental if there's a case where the external degree of such a node O_i is too high, indicating more external connections if considered in the community and hence more chance of escaping the community bond later. Hence nodes with high degree being entered through the pull are also prone to repulsion with the community.

Modified Utility Function The corresponding utility of the agent i to join a community C_t is given by follows.

$$\text{Payoff-Utility} = \mu(i, C_t) = \sum_{j \in C_t} (A_{ij} - \frac{k_i.k_j}{2m})\delta(C_t, C_i) \tag{1}$$

$$\text{Community-Retention} = \sum_{O_j \in O_{ext}} (A_{C_t,O_j} - \frac{d^{O_j}_{ext}}{2m}) \tag{2}$$

$$\lambda(\text{Payoff-Utility}) + (1-\lambda)(\text{Community-Retention}) \tag{3}$$

where $C = C_1, C_2, ..., C_{d_i}$, λ denotes the community retention rate and $\sum_j A_{C_t,O_j}$ includes all edge counts from community C_t to external node O_j.

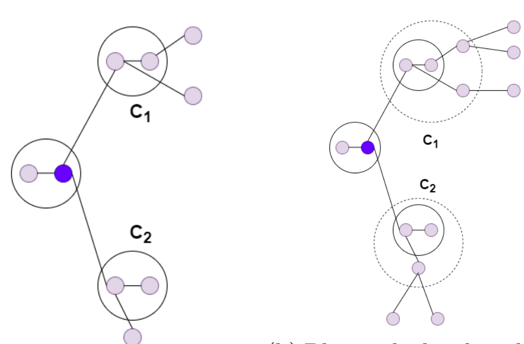
Sampling Strategy

Initially all the nodes in the graph have a uniform sampling probability P. After a node takes action based on eq.3 we assign the node a sampling probability given by eq. 4 for the next iteration.

$$Prob(N_i) = \frac{\lambda * P}{P_{N_1} + P_{N_2} + \cdots + \lambda * P + P_{N_n}} \tag{4}$$

By community retention mechanism, a particular node had previously chosen a community wisely such that there is less chance of switching to another community via its other branches later on. Hence the intuition is to give less chance to that node in the next iteration to be sampled, in the same proportionate amount as community retention rate. Also this guarantees that post community retention action, if a node gets chosen again and switches, its sampling probability gets factored down by λ successively.

Demonstration Example



(a) C_1: has 2 external nodes, C_2: has 1 external node

(b) Blue node decides whether to join C_1 or C_2 based on community retention

Figure 5: Community Retention & Sampling Strategy

In the above example, we show only how the community Retention can be calculated. The blue node chosen has an initial probability $= \frac{1}{14}$, total nodes n being 14 as per Fig 5b.

From Fig: 5b Total edges $m = 13$, Degree $d^{ext}_{C_1} = 3$, Degree $d^{ext}_{C_2} = 2$.

Community Retention for $C_1 = (6 - \frac{(d^{ext}_{C_1}*3 + d^{ext}_{C_1}*2)}{2m}) = 5.42$ since the external nodes (not considering blue node) for C_1 has degree 3 and 2 respectively. Community Retention for $C_2 = (4 - \frac{(d_{C_2}*3}{2m}) = 3.76$. Considering λ is 0.8, then the probability of the blue node for the next iteration is

$$\frac{(0.8) * P}{P_{N_1} + P_{N_2} + \cdots + (0.8) * P + P_{N_n}} = 0.0579 < P = \frac{1}{14}$$

where $P = P_{N_1} = P_{N-2} = ... = P_{N_n} = \frac{1}{14}$

Merging Strategy

Merging is done when a particular community C_1 wants to merge with another community C_2. Advantage of a merge strategy is that instead of serial node switchings all nodes from one community to its adjacent community in a single iteration. Considering a community C_x, we collapse it into one supernode SU_x with the outdegrees of the nodes within C_x represented as the outdegree of SU_x as in Fig 6 and the corresponding utility function is given by eq. 5. The maximum utility is then calculated as eq. 6

$$\mu_m(C_x, C_2) = \frac{1}{2m'} \sum (A_{C_x,C_2} - \frac{k_{C_x} k_{C_2}}{2m'}) \tag{5}$$

$$\text{utility}_{\text{comm}} = \max_{\forall j \in C}(\mu_m(C_i, C_j)) \tag{6}$$

Nodes' Unwillingness to merge However, some nodes in C_x might be reluctant to switch under such merge operation. So, intuitively their sampling probability should be higher in the next iterations, in order to get a higher chance of switching to better community.

Sampling Probability for merged nodes

For each node that got merged, we assign sampling Probability as eq 7 where P_m is the merging probability.

$$\text{Prob(n)} = \text{Prob(n)}/p_m \forall n \in C_x \tag{7}$$

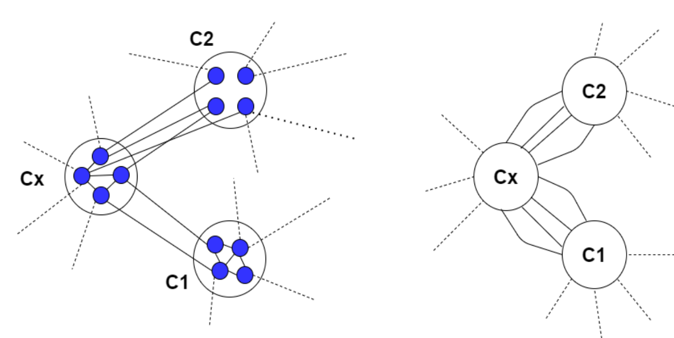


Figure 6: Whether Cx would merged with C1 or C2

Theoretical Proofs

Existence of Nash Equilibrium

Proposition 2. *The proposed utility function 3 follows additive-separable and symmetric properties.*

Proof : We show it only for the first term in eq.1 which can be shown for second term eq.2 also due to the same structure.

- **Symmetric Property** Suppose u joins community which has v. So, symmetric property holds for the first term a from 1, $A_{u,v} = A_{v,u}$. Also $\delta(C_u, C_v) = \delta(C_v, C_u) = 1$
- **Additively-Separable Property**

  By 1, $\delta(C_t, C_i) = 1$, since node i joins the C_t community.

  So $D(C, C_t) = \sum_{i \in C} \mu(i, C_t)$ indicates the utility for all nodes in C_i w.r.t nodes in C_t.

  Computing this for the entire partition set ϕ, yields, $\sum_{C \in \phi} D(C, C_t)$, which by average results in $\frac{1}{2m} \sum_{C \in \phi} \sum_{i \in C} \mu(i, S)$, i.e Q.

Algorithm 1 Temporal Community Detection using Game Theory

ProbabilityDist = $[p_1, p_2, ..., p_n], p_i = \frac{1}{N}$
π = InitialPartition(G)
$\pi_{prev} = \pi$
while true **do**
  node N_t = selectNode(NodeList, p = ProbabilityDist)
  π = CommunitySwitch(N_t, G, π)
  if N_t switches **then**
    $p_{N_t} = \lambda * p_{N_t}$
  end if
  P_m = random Number $s.t. \frac{1}{N} < P_m < 1$
  if $P_m <= \tau$ **then**
    π = MergeCommunity(G,π) .... *eq.*5 , *eq.*6 , *eq.*7
  end if
  if Terminate(π_{prev}, π) **then**
    BREAK
  end if
  $\pi_{prev} = \pi$
end while

Algorithm 2 Community Switch

Community Switch
for *each neighbor of node n* **do**
  totalUtility =λ(utilityJoin)+1-λ(communityRetention) *from equation* (3)
end for

Algorithm 3 Terminate

if $NMI(\pi_{prev}, \pi) >= \eta$ **then**
  Terminate algorithm
end if

Complexity Analysis

Proposition 3. *The* ***space complexity*** *provided by TCDG can be asymptotically represented as* $O(n^2) + O(n) + O(\phi) \approx O(n^2)$*, where* n *and* ϕ *indicate the number of nodes and number of partitions formed in the graph, respectively and* $n \geq \phi$*.*

Proof. Space Complexity

- *Storing Graph information as adjacency matrix* : $O(n^2)$
- *Storing dictionary of node and community labels* : $O(n)$ ex. $n_1 : C_5$, $n_2 : C_3$
- *Storing community dictionary* : $O(\phi)$ for ϕ = total community partitions formed e.x. $C_1 : (n_2, n_4), C_2 : (n_3, n_7)$

□

Proposition 4. *The* ***time complexity*** *provided by TCDG can be asymptotically represented as* $O(n * \phi) + O(\phi * (\phi - 1))$

Proof. Time Complexity

Considering ϕ as the number of partitions, the worst case complexity based on the utility function would be $O(n * \phi)$. For merging strategy the worst case complexity would be $O(\phi * (\phi - 1))$. □

Experiments and Results

We aim to answer the following questions:

Q1: **Community Structure** How closely does our method resemble traditional methods in terms of modularity and no. of communities: Table 3.

Q2: **Fast convergence** How does TGDC converges faster in less number of iterations.

Q3: **Real World Effectiveness** We have shown how our model performs in real world datasets including in recent covid research dataset. Table 2

Evaluation Metrics

Measurement of the Quality of Community *A. Normalized Mutual Information*

The measure of the mutual dependence between the communities is detected using Normalized Mutual Information (NMI) (Xie, Kelley, and Szymanski 2013). NMI measures the similarity between two partitions and denotes the quality of the partition. Let the two partitions or communities are C and C'. Then NMI can be denoted as

$$NMI(C, C') = \frac{2I(C, C')}{H(C) + H(C')} \tag{8}$$

Where $H(.)$ is the entropy function and the mutual information $I(C, C') = H(C) + H(C') - H(C, C')$. No similarity and maximum similarity of two communities manifest NMI value 0 and 1 respectively.

Experimental Setup

All experiments are carried out on a 2.4GHz Intel Core i9 processor, 32GB RAM, running OS Windows 10.0.18363. We ran the corresponding models TGDC, GCD and other baseline models in python using networkx library.

Datasets Used

We use the following datasets for evaluation purpose.

- *Amazon* is collected by crawling Amazon website. The vertices represent products; the edges indicate the frequently co-purchase relationships; the ground-truth communities are defined by the product categories in Amazon. This graph has 3,225 vertices and 10,262 edges
- *Zachary's karate club* (Zachary 1977) proposes a social network dataset of 34 members of a karate club at a US university in the 1970s.
- *American College football* (Girvan and Newman 2002a) consists of network of American football games between Division IA colleges during Fall 2000 season.
- *Dolphin social network* (Lusseau et al. 2003) consists of an undirected network of associations in dolphins in NZ.
- *Enron Email Dataset(Email)* proposes a communication network dataset of employees under Enron having around half million email conversations. For our experiments we sampled some instances as per Table 1.
- *ERDOS 992* proposes a pajek network dataset of 6.1K nodes and 7.5K edges.
- *Facebook Food Network* Data collected about Facebook pages (November 2017). These datasets represent blue verified Facebook page networks of different categories. Nodes represent the pages and edges are mutual likes among them.
- *Retweet Network* Nodes are twitter users and edges are retweets. These were collected from various social and political hashtags.
- *Facebook Politicians Network* Data collected about Facebook pages (November 2017). These datasets represent blue verified Facebook page networks of different categories. Nodes represent the pages and edges are mutual likes among them.

Baseline Models

- *MMSB* (Airoldi et al. 2008) uses dense subgraph extraction to detect overlapping communities in network graphs.

Network Name	Vertices	Edges	Degree
Amazon	3225	10262	12
Enron	143	623	10
Karate	34	78	1.2
Football	115	613	8.52
Dolphin	62	159	3
ERDOS992	6129	7591	2
Karate fb pages food	620	2102	6.78
retweet network	96	117	3.7
fb pages politician	5908	41729	14.12

Table 1: Networks under consideration

- *CPM* (Palla et al. 2005) uses k-clique information to generate corresponding communities.
- *Node2vec* (Grover and Leskovec 2016) uses vertex embeddings learned via biased random walk.
- *Fast-Unfold* (Blondel et al. 2008) is a community detection algorithm that tries to maximize the modularity using louvain heuristics.
- *Greedy MM* Newman et al (Clauset, Newman, and Moore 2004) starts with each node in its own community and tries to join pairs of communities until no such community pair is left.
- *GraphGAN* (Wang et al. 2018) uses adversarial training in a min-max game and combines generative and discriminative graph representation learning methods for finding communities.
- *GCD(Game-theoretic Community Detection)* We use (McSweeney, Mehrotra, and Oh 2012)'s basic community detection method using the node mechanism structure and aim to compare the convergence rate. This is also the case for $\lambda = 1$ in our model (i.e. without the community retention feature) and without any community merge.

As we see from the modularity values **TGDC** achieves a majority modularity in most cases.

Evaluation on Real world Datasets As seen from Table 2, it is clearly evident that our proposed Model TGDC performs well in comparison to the traditional community detection methods. All experiments corresponding to 2 were run over a period of 6 phases per model per dataset and the boundary values were decided based on the average of the 6 phases. Here we only showcase for larger datasets.

Evaluation on LFR Benchmark Datasets In Fig. 7 we also showcase our model using the Lancichinetti–Fortunato–Radicchi (LFR) benchmark dataset proposed by (Lancichinetti, Fortunato, and Radicchi 2008), using corresponding degree distributions and community size distributions. Here 4 graphs are generated for 4 different values of mixing parameter (μ) compared against their NMI values, where μ denotes the average ratio between the external connections of a node to its degree. We see that our proposed model

Network Name	TGDC(Our Model)	MMSB	CPM	Node2vec	Fast-unfold	Greedy MM	GraphGAN
Amazon	**0.77** ± **0.01**	0.63 ± 0.013	0.56 ± 0.007	0.59 ± 0.03	0.67 ± 0.02	0.62 ± 0.04	0.58 ± 0.01
Enron	**0.54** ± **0.01**	0.418 ± 0.008	0.380 ± 0.011	0.38 ± 0.026	0.42 ± 0.015	0.47 ± 0.02	0.51 ± 0.02
Karate	**0.42** ± **0.014**	0.418 ± 0.008	0.380 ± 0.011	0.38 ± 0.026	0.35 ± 0.13	0.39 ± 0.017	0.38 ± 0.021
Football	**0.582** ± **0.002**	**0.604** ± **0.02**	0.549 ± 0.064	0.573 ± 0.037	0.51 ±0.02	0.54 ±0.03	0.562 ±0.23
Dolphin	0.58 ± 0.016	0.55 ± 0.023	0.53 ± 0.036	0.57 ± 0.046	0.52 ± 0.04	0.51 ± 0.037	0.48 ± 0.12
ERDOS992	0.618 ± 0.002	0.604 ± 0.02	0.549 ± 0.064	0.573 ± 0.037	0.58 ± 0.02	0.571 ± 0.07	**0.63** ± **0.01**
FB-Pages-Food	0.61 ± 0.007	0.563 ± 0.014	0.503 ± 0.03	0.518 ± 0.011	0.55 ± 0.01	0.53 ± 0.03	**0.61** ± **0.04**
Retweet	0.51 ± 0.01	0.55 ± 0.023	0.53 ± 0.036	**0.57** ± **0.046**	0.48 ± 0.02	0.52 ± 0.03	0.57 ± 0.03
FB-Pages-Politician	**0.71** ± **0.02**	0.55 ± 0.023	0.53 ± 0.036	0.57 ± 0.046	0.64 ± 0.01	0.61 ± 0.003	0.704 ± 0.04

Table 2: Modularity Comparison for ($\lambda = 0.8$)

Network Name	TGDC(Our Model)	MMSB	CPM	Node2vec	Fast-unfold	Greedy MM	GraphGAN
Amazon	445	423	417	409	338	431	312
Enron	13	14	13	13	13	12	13
Karate	3	4	3	3	5	4	4
Football	8	10	6	7	13	11	8
Dolphin	9	11	8	10	8	8	9
ERDOS992	606	580	592	558	621	586	574
FB-Pages-Food	80	72	61	71	77	73	86
Retweet	32	27	28	30	35	34	30
FB-Pages-Politician	9	11	8	10	11	9	9

Table 3: No. of Communities Comparison for $\lambda = 0.8$

TGDC performs comparatively better than the baselines.

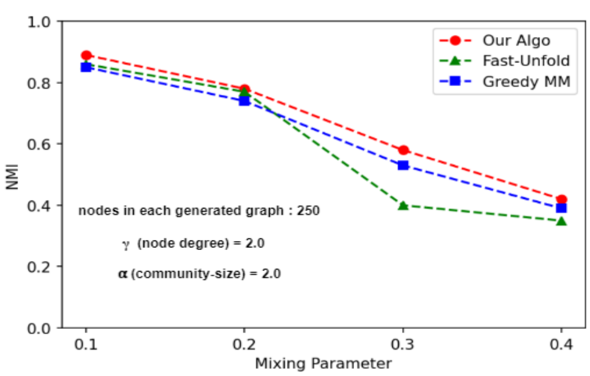
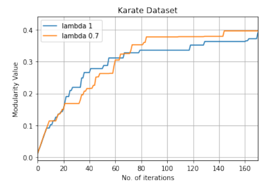


Figure 7: LFR benchmark Dataset

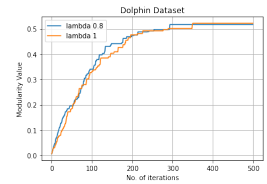
Fast convergence w.r.t GCD ($\lambda = 1$)

We mainly focus on convergence comparison of our model w.r.t game theoretic models like GCD, primarily due to the reason that other baseline models have different domain architecture and different definitions for iteration.

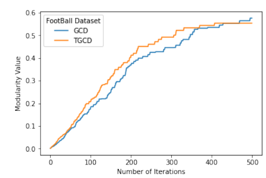
In Figure 8 we show how TGDC achieves almost the same modularity in less number of iterations as compared to the normal game theoretic model GCD . Here $\lambda = 1$ corresponds to the case of GCD baseline model and we use this notation interchangeably in the experiments section. In Fig. 8a, for Karate dataset,at iteration no. 80, value at $\lambda_{0.7}$ is higher than at λ_1. And this behaviour is continued till the maximum modularity is achieved. Similarly in Fig 8c for the Football Dataset, this behaviour is seen where our proposed model (TGDC) with $\lambda = 0.7$ achieves higher modularity in less iterations, thus can converge faster.



(a) **Karate Dataset** - TGDC($\lambda = 0.7$) reaches peak modularity 0.38



(b) **FB pages Food Dataset** - TGDC($\lambda = 0.8$) reaches peak modularity 0.58



(c) **FootBall Dataset** - From iteration no. 100 to 400, TGDC($\lambda = 0.8$) has higher modularity values compared to GCD($\lambda = 1$)

Figure 8: Faster Convergence for TGDC model

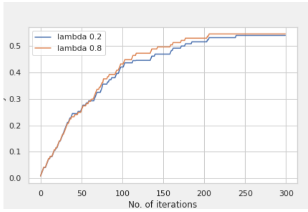
Choosing optimum value for lambda

High lambda indicates more focus on local environment based utility while low lambda indicates more focus on future community retention. As we see from 9 having very low lambda value results in more focus on future retention rather than taking decision based on local environment, hence a node might not join a profitable community, leading to slower convergence.

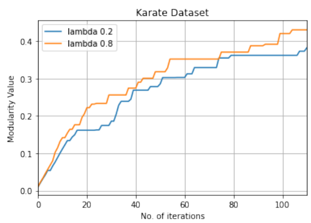
This is evident in Figure 9 where in 9a and 9b, lower lambda values perform poorly w.r.t higher lambda values. Example : In Fig . 9a, $\lambda_{0.8}$ value is considerably higher than $\lambda_{0.2}$ for iterations greater than 100.

Conclusion & Future work

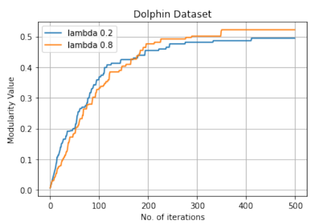
We see from our experiments and results on how our proposed approach can result in faster convergence with the optimum value of λ, at the same time achieving stable community partition. For applicability our model has also been tested on real world network datasets including Amazon Dataset to show how it performs as compared to traditional community detection models. In future we aim to propose a dynamic version of this approach where we propose to recompute communities



(a) Lower lambda value of 0.2 results in poor convergence - Retweet Dataset



(b) Lower lambda value of 0.2 results in poor convergence- Karate Dataset



(c) For Dolphin dataset, post iteration 200, lambda 0.2 achieves less modularity compared to lambda 0.8.

Figure 9: Deciding optimum lambda value - More iterations required for lower lambda values to achieve peak modularity

for specific nodes based on community retention history as opposed to recomputing the communities for each static snapshot.

References

Airoldi, E. M.; Blei, D. M.; Fienberg, S. E.; and Xing, E. P. 2008. Mixed membership stochastic blockmodels. *Journal of machine learning research* .

Arab, M. 2021. Providing an Approach to Predicting Customer Quality in E-Commerce Social Networks Based on Big Data and Unsupervised Learning Method. *arXiv preprint arXiv:2109.02080* .

Basu, S.; and Maulik, U. 2015. Community detection based on strong Nash stable graph partition. *Social Network Analysis and Mining* 5(1): 61.

Blondel, V. D.; Guillaume, J.-L.; Lambiotte, R.; and Lefebvre, E. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008(10): P10008.

Bogomolnaia, A.; Jackson, M. O.; et al. 2002. The stability of hedonic coalition structures. *Games and Economic Behavior* 38(2): 201–230.

Bu, Z.; Cao, J.; Li, H.-J.; Gao, G.; and Tao, H. 2018. GLEAM: A graph clustering framework based on potential game optimization for large-scale social networks. *Knowledge and Information Systems* 55(3): 741–770.

Cheng, Q.; Liu, Z.; Huang, J.; Zhu, C.; and Liu, Y. 2012. Hierarchical clustering based on hyper-edge similarity for community detection. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2012 IEEE/WIC/ACM International Conferences on*, volume 1, 238–242. IEEE.

Clauset, A.; Newman, M. E.; and Moore, C. 2004. Finding community structure in very large networks. *Physical review E* 70(6): 066111.

Feng, H.; Tian, J.; Wang, H. J.; and Li, M. 2015. Personalized recommendations based on time-weighted overlapping community detection. *Information & Management* 52(7): 789–800.

Gangopadhyay, A.; and Chen, S. 2016. Health care fraud detection with community detection algorithms. In *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*, 1–5. IEEE.

Girvan, M.; and Newman, M. E. 2002a. Community structure in social and biological networks. *Proceedings of the national academy of sciences* 99(12): 7821–7826.

Girvan, M.; and Newman, M. E. J. 2002b. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences* 99(12): 7821–7826. doi:10.1073/pnas.122653799.

Grover, A.; and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 855–864.

Ji, T.; Luo, C.; Guo, Y.; Wang, Q.; Yu, L.; and Li, P. 2020. Community detection in online social networks: a differentially private and parsimonious approach. *IEEE transactions on computational social systems* 7(1): 151–163.

Jiang, F.; and Xu, J. 2015. Dynamic community detection based on game theory in social networks. In *2015 IEEE international conference on big data (Big Data)*, 2368–2373. IEEE.

Lancichinetti, A.; Fortunato, S.; and Radicchi, F. 2008. Benchmark graphs for testing community detection algorithms. *Physical review E* 78(4): 046110.

Li, Z.; Hui, P.; Zhang, P.; Huang, J.; Wang, B.; Tian, L.; Zhang, J.; Gao, J.; and Tang, X. 2021. What Happens Behind the Scene? Towards Fraud Community Detection in E-Commerce from Online to Offline. In *Companion Proceedings of the Web Conference 2021*, 105–113.

Li, Z.; Liu, J.; and Wu, K. 2017. A multiobjective evolutionary algorithm based on structural and attribute similarities for community detection in attributed networks. *IEEE transactions on cybernetics* 48(7): 1963–1976.

Lusseau, D.; Schneider, K.; Boisseau, O. J.; Haase, P.; Slooten, E.; and Dawson, S. M. 2003. The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology* 54(4): 396–405.

McSweeney, P. J.; Mehrotra, K.; and Oh, J. C. 2012. A game theoretic framework for community detection. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 227–234. IEEE.

Newman, M. E. 2004. Fast algorithm for detecting community structure in networks. *Physical review E* 69(6): 066133.

Palla, G.; Derényi, I.; Farkas, I.; and Vicsek, T. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *nature* 435(7043): 814–818.

Wang, H.; Wang, J.; Wang, J.; Zhao, M.; Zhang, W.; Zhang, F.; Xie, X.; and Guo, M. 2018. Graphgan: Graph representation learning with generative adversarial nets. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Xie, J.; Kelley, S.; and Szymanski, B. K. 2013. Overlapping community detection in networks: The state-of-the-art and comparative study. *Acm computing surveys (csur)* 45(4): 1–35.

Ying, J.-C.; Shi, B.-N.; Tseng, V. S.; Tsai, H.-W.; Cheng, K. H.; and Lin, S.-C. 2013. Preference-aware community detection for item recommendation. In *2013 conference on technologies and applications of artificial intelligence*, 49–54. IEEE.

Zachary, W. W. 1977. An information flow model for conflict and fission in small groups. *Journal of anthropological research* 33(4): 452–473.