Topic – Dynamic Community detection

Community detection in temporal networks via a spreading process

Rohit Paul
1815032

Abstract - In this mini project, CollegeMsg dataset has been chosen which is a timestamped dataset. The edges are represented in the form (a, b, c) where there exists a directed edge from node a to b at time epoch time c. A modified version of Susceptible Infected Recovered spreading process has been implemented which detects communities in a particular timestamp taking into account the structure of communities in the previous timestamps. The dynamic community detection algorithm adopts a temporal tradeoff method for the detection of communities across the different timestamps dynamically.

1. Introduction

Community or subgraph structure is ubiquitous in many complex systems, such as human contact networks, social networks, and technological networks. Intuitively, a community's defining property is that nodes within the community are more tightly connected with each other compared to nodes outside the community. Operationalising this definition and subsequently identifying the community structure of a complex system often deepens the understanding of the system's functioning and improves the control of the system's dynamics. Consequently, community detection has become a topic of great importance and interest in network science. Heretofore, however, most algorithms for the purpose of community detection focused on static networks, whereas, in practice, system topology often evolves in time [9]. In social networks, for example, communities often merge or split because of the movement of individuals. Investigating community detection in such "temporal" networks is therefore an ongoing challenge for network scientists and a main concern of this manuscript.

1. Dynamic Community detection

Dynamic community detection is a more practical form of community detection when compared to static community detection because all real world networks exist in the form of dynamic networks.

In this type of community detection, the snapshots of communities can be treated as separate static networks and static community detection algorithms can be applied to each of them to find communities in the network. The main forms of dynamic community detection are-

- Instant optimal
- Temporal tradeoff
- Cross-time community detection

Instant optimal - in this type, the snapshots are treated as separate and static algorithms are applied on each snapshot separately and finally they are merged taking every adjacent pair of snapshots into account and the final result is achieved.

Temporal tradeoff - in this type whenever finding communities in a particular snapshot, the community structure of the previous snapshot is taken into consideration. There is a higher probability of two nodes being in the same community if they were in the same community in the previous snapshot.

Crostt-time community detection - it is the most complex method of community detection in the bunch. It considers the node of the entire network at the same time across all the snapshots and assigns them a community. The algorithm used in this technique is more complex than the ones used in the previous two.

2. SIR Model

The spreading model used in this project is the SIR model the details of which has been explained in the research paper thoroughly. It is used to create a similarity measure between similar nodes so that nodes can be partitioned into separate communities.

Given a set of time steps {1, 2,...,T }, a temporal network is a set of sequential graphs,

$$G = \{G[1], G[2],...,G[T] \},$$

where $G[t] = (V,E[t])$ and $V = \{v1, v2,...,vN \}$ is a set of nodes, while $E[t] \subseteq V \times V$ is a set of links between nodes at time t.

An alternative way of representing G[t] is by means of an $N \times N$ adjacency matrix, A[t] , where the matrix element $A[t]_{ij} = 1$ if $(vi, vj) \in E[t]$, and $A[t]_{ij} = 0$ otherwise.

3. Spreading

We adopted a traditional SIR model in which each individual might be in one of the following three states: susceptible S, infectious I, or recovered R. S individuals may get infected due to the physical contacts with their I neighbours, and this occurs with infection probability λ. Furthermore, I individuals may recover with probability μ. Given an initial number of I individuals, the spreading process has a natural end when there are no more I individuals left in the network. Our interest, however, is not necessarily in this final state, but rather states that best quantify node similarity. A Markov model for the SIR spreading process.

The details of the formulae used have been shown in the research paper.

4. A similarity measure has been created and now a partition matrix is formed for each time stamp. Now this partition matrix can be converted to a distance matrix by using Pearson correlation. This distance matrix is then clustered using Agglomerative clustering which is a form of hierarchical clustering.

5. Tools used

Programming language - Python programming language is used for the implementation of the algorithm.

Softwares used - google colab is used as the software which is an online platform for executing code. Jupyter lab for alternate method of executing the code.

Link to code - https://colab.research.google.com/drive/13eHhRkMCua1eJJaFvVDSC86--973WtGj?usp=sharing

Link to dataset used - https://snap.stanford.edu/data/CollegeMsg.html

Libraries used -

- Networkx
- Matplotlib
- Sklearn
- Math

Instructions to execution of code -

Method 1 -

- Unzip the folder
- Go to the link for code in google colab
- Upload the CollegeMsg.txt file to google colab
- Execute each block from top to bottom

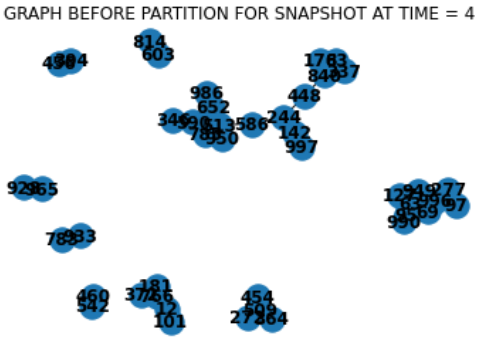
Method 2 -

- Unzip the folder
- Place the CollegeMsg.txt and Dynamic.ipynb files in the same directory
- Open terminal
- Type jupyter-lab
- Open the Dynamic.ipynb file
- Run each cell in the jupyter lab

6. Results

Three timestamps have been considered. The structures of the networks before partitioning are given below

Time stamp 1 :



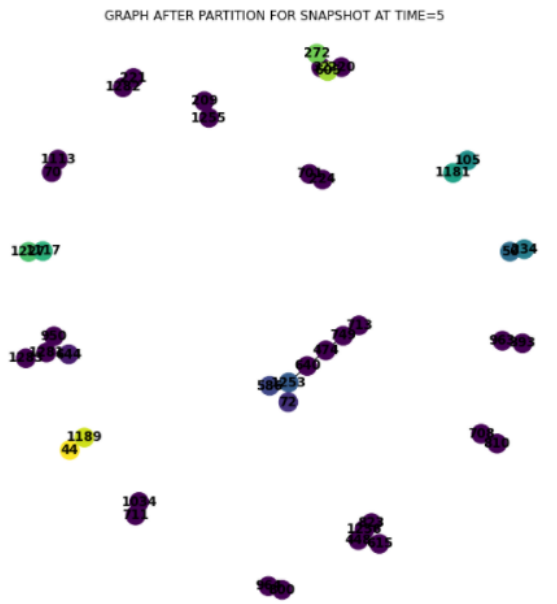
Time stamp 2 :

Time stamp 3 :

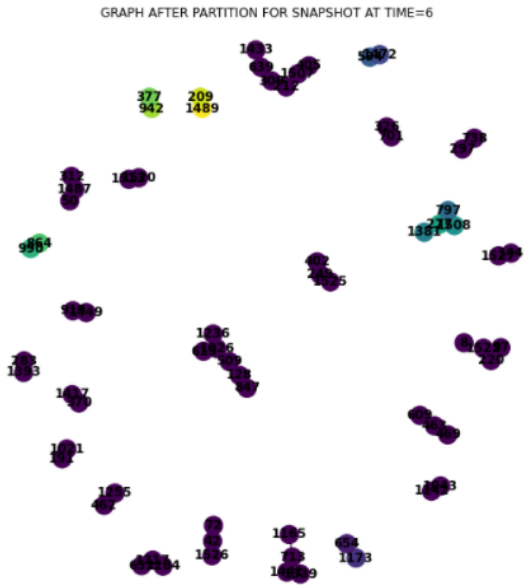
The graphs after clustering are given below

Time stamp 1 :

Time stamp 2 :



Time stamp 3 :



Similarity between the adjacent snapshots
Similarity between snapshot 1 and 2 =
69.04761904761905%
Similarity between snapshot 2 and 3 =
92.3076923076923%

```
Simmilarity between snapshot 1 and snapshot 2
69.04761904761905%
Simmilarity between snapshot 2 and snapshot 3
92.3076923076923%
```

7. Reference

- Community detection in temporal networks via a spreading process Peican Zhu, Xiangfeng Dai, Xuelong Li, Chao Gao4, Marko Jusup and Zhen Wang