TimeRank: a Random Walk approach for Community Discovery in Dynamic Networks

Ilias Sarantopoulos[1,2] Dimitrios Papatheodorou[2,4] Dimitrios Vogiatzis[1,3] Grigorios Tzortzis[1] Georgios Paliouras[1]

[1]National Centre for Scientific Research (NCSR) “Demokritos”, Athens, Greece,

[2]Athens University of Economics and Business, Athens, Greece

[3]The American College of Greece, Athens, Greece,

[4]Aalto University, Espoo, Finland

Table of contents

Introduction

The setting

- An undirected graph which represents a (social) network

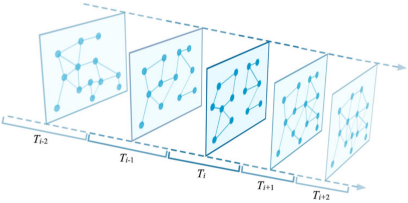

Figure 1: An example of a dynamic network consisting of five timeframes as shown in [1]

The setting

- An undirected graph which represents a (social) network
- Graph topology changes over time

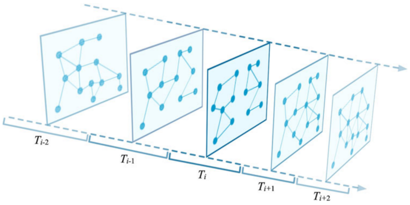

Figure 1: An example of a dynamic network consisting of five timeframes as shown in [1]

The setting

- An undirected graph which represents a (social) network
- Graph topology changes over time
- We use several discrete snapshots of the network and refer to them as timeframes

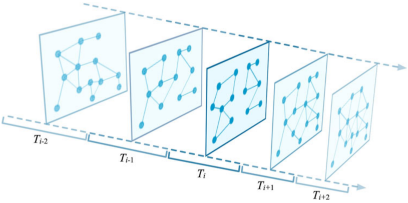

Figure 1: An example of a dynamic network consisting of five timeframes as shown in [1]

Dynamic Community Finding

- The communities (i.e. Clusters) in each timeframe (**Detection**)
- Track the communities across time (**Tracking**)

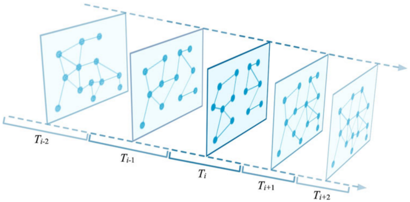

Figure 2: An example of a dynamic network consisting of five timeframes as shown in [1]

Common approaches

Two Step methods

1. 1st Step: Detect communities in each Timeframe graph (louvain, spectral etc.)
2. 2nd Step: Match communities between timeframes based on some similarity measure

One Step methods

Perform detection and tracking in one step (e.g. using Non-Negative Tensor Factorisation)

Background

MutuRank - Purpose

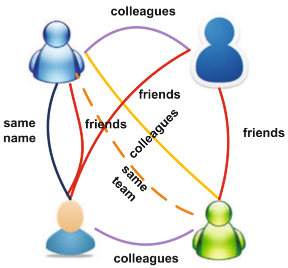

Figure 3: A multi relational network as described in [5]

- Perform Random walk on edges and relations
- Rank nodes and relations
- Transform Multi-relational network to Single relational
- Detect communities

MutuRank Algorithm

$$p_i^t = \alpha \sum_{j=1}^{n} \sum_{d=1}^{m} p_j^{t-1} \cdot o_{i,j,d} \cdot Prob^{t-1}[d|j] + (1-\alpha) p_i^*, \tag{1}$$

$$q_d^t = \beta \sum_{i=1}^{n} \sum_{j=1}^{n} p_j^{t-1} \cdot r_{i,j,d} \cdot Prob^{t-1}[i|j] + (1-\beta) q_d^* \tag{2}$$

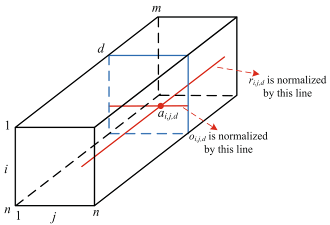

Figure 4: Depiction of the normalisation of tensors $\mathcal{O}$ and $\mathcal{R}$ in MutuRank and TimeRank (source [5]).

Transformation

Relations and nodes yield mutual influence.
Use **q** to transform to Single Relational Network (SRN)

$$w_{i,j} = \sum_{d=1}^{m} q_d \cdot a_{i,j,d}, \tag{3}$$

Then, detect communities using any algorithm

TimeRank

Adapting Muturank for community tracking

- Although communities evolve, they tend to have common structure.

Adapting Muturank for community tracking

- Although communities evolve, they tend to have common structure.
- Nodes and Communities in different timeframes are disconnected

Adapting Muturank for community tracking

- Although communities evolve, they tend to have common structure.
- Nodes and Communities in different timeframes are disconnected
- Most Community detection algorithms rely on close connectivity

Adapting Muturank for community tracking

N = # of Nodes
T = # of Timeframes

- Relations (Muturank) = Timeframes (Timerank)
- Adapt muturank representation to include time-varying nodes: $(N \times T) \times (N \times T) \times T$
- Add intra-timeframe edges (network edges)
- Add inter-timeframe edges: connect the same node with its image between timeframes.
- Apply muturank →temporal network with $N \times T$ nodes
- Perform clustering on this network and extract dynamic communities

Timerank - One to All Connection (AOC)

- 'One to All Connection' connects each node i_t with all its occurrences in other timeframes
- for each node i_t add the following pairs of edges in the network

$$\{(i_1, i_t), \ldots, (i_{t-1}, i_t), (i_{t+1}, i_t), \ldots, (i_T, i_t)\}$$

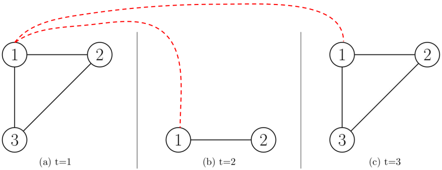

Figure 5: Sample dynamic network with 3 timeframes demonstrating AOC connections

Timerank - Next Occurrence Connection (NOC)

- 'Next Occurrence Connection' connects each node i_t with its images in the previous and next timeframes in which this node exists
- for each node i_t add the following pairs of edges in the network

$$\{(i_{t-1}, i_t), (i_{t+1}, i_t)\}$$

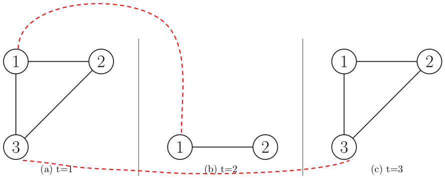

Figure 6: Sample dynamic network with 3 timeframes demonstrating NOC connections

TimeRank Algorithm

Algorithm 1 TimeRank Algorithm

1: Create $(N \times T)$ adjacency matrices for each timeframe
2: Add inter-time edges
3: Compose tensor $\mathcal{A} \in \mathbb{R}^{(N\times T)\times(N\times T)\times(T)}$
4: Apply MutuRank algorithm on $\mathcal{A}$ and get ranking of timeframes
5: Create time-weighted network with $(N \times T)$ nodes
6: Perform clustering on this network and extract dynamic communities

Datasets & Experiments

Datasets

Synthetic Datasets using Dynamic Benchmark Network Generator [3], which is based on [4].

Expand/Contract events

1. 1000 nodes / 5 timeframes
2. 32 communities
3. 10 expand, 10 contract
4. 25% expansion/contraction rate

Hide/Appear events

1. 1000 nodes / 5 timeframes
2. 32 communities
3. 10% of communities hide

Compared Methods (1/2)

Two step approach : Group Evolution Discovery (GED), Piotr Bródka, Stanislaw Saganowski, and Przemyslaw Kazienko

1. Run community detection algorithm on each timeframe (e.g. Louvain)
2. Match communities between sequential timeframes using a similarity measure

Compared Methods (2/2)

One step approach : Non-Negative Tensor Factorisation, Laetitia Gauvin, André Panisson, and Ciro Cattuto

Perform PARAFAC decomposition on 3-way tensor $T \in \mathbb{R}^{N \times N \times S}$, where N is the number of nodes of the network and S the number of network snapshots.

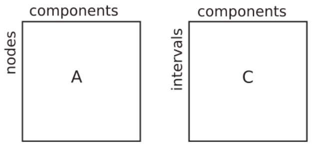

Figure 7: Schematic representation of the factorisation result for an undirected temporal network from [2].

Experimental Setting

1. GED
    - Ground truth communities as input
2. NNTF
    - Random restarts for initialisation of factors **A, B** and **C**.
3. TimeRank
    - AOC connection - uniform distribution for q
    - NOC connection - uniform distribution for q
    - AOC connection - run Muturank
    - NOC connection - run Muturank

Evaluation Metrics

- NMI for overlapping clusters
- Omega index (rand index expansion for overlapping clusters)
- BCubed

Results

	Expand/Contract					Hide/Appear				
Method	NMI	Omega	Prec	Rec	F1	NMI	Omega	Prec	Rec	F1
TR-AOC-U	0.866	0.874	0.905	0.882	0.893	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
TR-NOC-U	0.908	0.919	0.944	0.921	0.933	0.880	0.890	0.912	0.963	0.937
TR-AOC	0.849	0.864	0.890	0.883	0.886	**1.000**	**1.000**	**1.000**	**1.0000**	**1.000**
TR-NOC	**0.923**	**0.954**	**0.964**	**0.944**	**0.953**	0.910	0.918	0.935	0.963	0.949
NNTF	0.805	0.8445	0.842	0.864	0.853	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
GED	0.464	0.659	0.924	0.572	0.707	0.531	0.700	0.901	0.662	0.763

Table 1: Tables for Expand/Contract and Hide/Appear Datasets

	Expand/Contract					Hide/Appear				
Method	t_1	t_2	t_3	t_4	t_5	t_1	t_2	t_3	t_4	t_5
TR-AOC	0.212	0.196	0.198	0.196	0.199	0.214	0.189	0.199	0.191	0.207
TR-NOC	0.217	0.189	0.191	0.196	0.207	0.218	0.183	0.193	0.193	0.213

Table 2: Values for $\mathbf{q}$ distribution for the *Expand/Contract* and *Hide/Appear* datasets

Reddit

- Dynamic community : subreddit
- Timeframe: week
- Nodes: users
- Edges: replies

Reddit Experiment 1

	4 Timeframes					8 Timeframes				
Method	NMI	Omega	Prec	Rec	F1	NMI	Omega	Prec	Rec	F1
TR-AOC-U	0.385	0.316	0.557	0.56	0.558	0.319	0.243	0.547	0.543	0.545
TR-NOC-U	0.480	0.428	0.639	0.619	0.629	0.377	0.455	0.612	0.580	0.596
TR-AOC	0.390	0.373	0.576	0.605	0.591	0.295	0.217	0.531	0.521	0.526
TR-NOC	0.457	0.473	0.633	0.623	0.628	**0.435**	**0.537**	0.610	**0.654**	**0.631**
NNTF	0.447	0.496	0.627	**0.642**	0.634	0.395	0.480	0.590	**0.650**	0.619
GED-T	**0.584**	**0.776**	**1.000**	0.625	**0.769**	0.323	0.432	**1.000**	0.377	0.548

Table 3: Experiment 1 results for 4 and 8 timeframes

Reddit Experiment 2

	4 Timeframes					8 Timeframes				
Method	NMI	Omega	Prec	Rec	F1	NMI	Omega	Prec	Rec	F1
TR-AOC-U	0.050	0.068	0.480	0.544	0.510	0.028	-0.007	0.438	0.438	0.438
TR-NOC-U	0.380	0.487	0.692	**0.905**	0.784	0.032	0.032	0.435	0.536	0.480
TR-AOC	0.038	0.032	0.471	0.542	0.500	0.028	-0.007	0.438	0.439	0.439
TR-NOC	**0.456**	**0.604**	0.741	**0.951**	**0.833**	0.031	0.030	0.435	0.536	0.480
NNTF	0.276	0.389	0.723	0.652	0.686	**0.637**	**0.772**	0.849	**0.933**	**0.890**
GED-T	0.210	0.280	**1.000**	0.269	0.424	0.099	0.147	**1.000**	0.132	0.233

Table 4: Experiment 2 results for 4 and 8 timeframes

Future Work

Future Work

- Experiments on DBLP Data
- Mix Benchmark Data
- Use Multirank instead of Muturank
- Add Weights in inter-timeframe edges
- Scaling through parallelization

Questions?

Evaluation measures

BCubed

$$Precision(u,v) = \frac{Min(|T(u) \cap T(v)|, |C(u) \cap C(v)|)}{|C(u) \cap C(v)|}$$

$$Recall(u,v) = \frac{Min(|T(u) \cap T(v)|, |C(u) \cap C(v)|)}{|T(u) \cap T(v)|}$$

$$F_1 = 2 \cdot \frac{BCubed_{Precision} \cdot BCubed_{Recall}}{BCubed_{Precision} + BCubed_{Recall}}$$

P. Bródka, S. Saganowski, and P. Kazienko.
Ged: the method for group evolution discovery in social networks.
Social Network Analysis and Mining, 3(1):1–14, 2013.

L. Gauvin, A. Panisson, and C. Cattuto.
Detecting the community structure and activity patterns of temporal networks: a non-negative tensor factorization approach.
PloS one, 9(1):e86028, 2014.

D. Greene, D. Doyle, and P. Cunningham.
Tracking the evolution of communities in dynamic social networks.
In *Advances in social networks analysis and mining (ASONAM), 2010 international conference on*, pages 176–183. IEEE, 2010.

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

Z. Wu, J. Cao, G. Zhu, W. Yin, A. Cuzzocrea, and J. Shi.
Detecting overlapping communities in poly-relational networks.
World Wide Web, 18(5):1373–1390, 2015.