UNCOVERING STRUCTURE IN SOCIAL NETWORKS

by

Matthew Bain Revelle
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Computer Science

Committee:

_____	Dr. Carlotta Domeniconi, Dissertation Director
_____	Dr. Daniel Barbará, Committee Member
_____	Dr. Kathryn B. Laskey, Committee Member
_____	Dr. Huzefa Rangwala, Committee Member
_____	Dr. David Rosenblum, Department Chair
_____	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering

Date: _____ Summer Semester 2021
George Mason University
Fairfax, VA

Uncovering Structure in Social Networks

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

By

Matthew Bain Revelle
Master of Science
George Mason University, 2009
Bachelor of Science
University of Mary Washington, 2003

Director: Dr. Carlotta Domeniconi, Professor
Department of Computer Science

Summer Semester 2021
George Mason University
Fairfax, VA

Dedication

I dedicate this dissertation to my family and friends.

To you all. Thank you for asking me "how much longer?", listening to me ramble about my research, and your words of encouragement. Special thanks to the Schultz family, the Hall family, Ben Bennett, Car Bauer, Ryan Donnelly, Ryan Kincade, Dr. Heidi Lawrence, Ryan Garner, and Joshua Lowder.

To my colleagues at Kudu Dynamics. I would not have been able to complete this milestone without your support.

To the professors at Mary Washington College and George Mason University that took time to mentor and challenge me. Especially Prof. Rita D'Arcangelis, Prof. Zoran Duric, Prof. Sean Luke, Prof. John Reynolds, and Prof. Marsha Zaidman.

To Billie. You were my one constant during this period of my life. The comfort and companionship you gave me helped me recharge and you always reminded me to go outside and get lost in the woods.

To my sibling, Garrett Revelle. Thank you for our random conversations of all kinds and allowing me to occasionally tag along in gaming matches when I needed a break.

To my parents, Christine McGowan and John Revelle. Thank you for encouraging me to learn and persevere. You provided me with an environment to explore and be curious.

To Shane. You care for me and help me become a better person. You are an inspiration to me in so many ways. Thank you for supporting me during the often stressful experience of completing my dissertation. I love you.

Acknowledgments

I would like to thank everyone who made the work of this dissertation possible.

I thank all the members of the Data Mining and Machine Learning (DMML) lab at George Mason University for their discussions, insights, and support. It was inspiring to work alongside so many engaged students and faculty.

I thank all my collaborators in the Volgenau School of Engineering, particularly Ben Gelman, Dr. Aditya Johri, Mack Sweeney, and Dr. Seungwon Yang.

I thank my dissertation committee members, Dr. Daniel Barbará, Dr. Kathryn B. Laskey, and Dr. Huzefa Rangwala. I appreciate the time taken to serve on the committee. I learned much from each of you through lectures, lab meetings, reading groups, and office visits. My only regret is that I did not work more closely with you during my time in the program.

My sincere thanks go to my dissertation director, Dr. Carlotta Domeniconi. Dr. Domeniconi's reputation as a rigorous mathematical thinker with high expectations for herself and her students is what caused me to seek her out as a research advisor. She has consistently pushed me to improve and perform all while permitting me to explore research areas new to both of us. Her patience and critiques, especially as I worked on the final portion of this dissertation, are greatly appreciated.

Table of Contents

List of Tables

List of Figures

Abstract

UNCOVERING STRUCTURE IN SOCIAL NETWORKS

Matthew Bain Revelle, PhD

George Mason University, 2021

Dissertation Director: Dr. Carlotta Domeniconi

Social networks are defined by relationships between people and permeate all aspects of human life. Improving our understanding of the structure and dynamics of networks enhances our knowledge of many human systems. In this dissertation, I present novel techniques and methodologies for the tasks of community detection, role discovery, and community evolution prediction as well as an analysis on temporal artifacts that may occur when constructing social network datasets.

Community detection in networks is a broad problem with many proposed solutions. Existing methods frequently make use of edge density and node attributes; however, the methods ultimately have different definitions of community and build strong assumptions about community features into their models. I propose a new method for community detection, which estimates both per-community feature distributions (topics) and per-node community membership. Communities are modeled as connected subgraphs with nodes sharing similar attributes. Nodes may join multiple communities and share common attributes with each. Communities have an associated probability distribution over attributes and node attributes are modeled as draws from a mixture distribution. The method includes two basic assumptions about community structure: communities are densely connected and have a small network diameter.

These assumptions inform the estimation of community topics and membership assignments without being too prescriptive.

There has been extensive research on social networks and methods for specific tasks such as: community detection, link prediction, and tracing information cascades; and a recent emphasis on using temporal dynamics of social networks to improve method performance. The underlying models are based on structural properties of the network, some of which we believe to be artifacts introduced from common misrepresentations of social networks. Specifically, representing a social network or series of social networks as an accumulation of network snapshots is problematic. I demonstrate how cumulative graphs differ from activity-based graphs and may introduce temporal artifacts.

Users in online social networks often have very different structural positions which may be attributed to a latent factor: roles. We analyze dynamic networks from two datasets (Facebook and Scratch) to find roles which define users' structural positions. Each dynamic network is partitioned into snapshots and we independently find roles for each network snapshot. I developed a role discovery methodology and investigate how roles differ between snapshots and datasets. Six persistent roles are found and user role membership, transitions between roles, and interaction preferences are analyzed and presented.

Communities in social networks evolve over time as nodes enter and leave the network and their activity behaviors shift. I present a novel technique for predicting community evolution events based on group-node attention. Group-node attention enables support for variable-sized inputs and learned representation of groups based on member and neighbor node features, including temporal information. Existing work on community evolution prediction has focused on the development of frameworks for defining events while using traditional classification methods to perform the actual prediction. It is my hope this work on a prediction model for community evolution events will prompt the development of additional novel prediction models.

Chapter 1: Introduction

1.1 Motivation

The use of data mining and machine learning in social networks is an active research area and encompasses many different tasks and approaches. In this dissertation, I present several models and analyses of social networks with an emphasis on social interaction networks, communities, persistent roles, and community dynamics. I have developed and contributed methods for community detection, role discovery, and community evolution prediction and presented findings on the effects of eschewing edge deactivation in social interaction networks and the persistence of network roles in the same.

Social interaction networks describe interactions between actors, usually people. These interactions may be phone calls, social media messages, or in-person conversations. In static network analysis, a single cumulative network is formed from a series of pairwise interactions that are accumulated as network edges. In the setting of static networks, I developed a community detection model which is presented in Chapter 4.

Community detection is the problem of finding related sets of nodes in a network. Generally, structural information is used to define these sets and we expect these nodes to be more densely connected to each other relative to the rest of the network. The problem is similar to clustering but in a network setting and also suffers from being an ill-posed problem. I developed a community detection model [1] which uses not only network structure but also node attributes. This model jointly learns communities along with a topic estimated by text associated with community members. My work on this model, and the comparative evaluation we performed, introduced me to other competitive models. A few related methods [2, 3] challenged the existing definition of communities as being subgraphs with high edge density [4, 5] and presented analyses indicating the overlap of multiple communities

are the densest subgraphs.

These models were based on the analyses conducted in [6, 7] that demonstrated the presences of densification—the super-linear growth of edges relative to nodes—and network diameter shrinking. For certain kinds of networks, where edges are only added and never removed, the finding is unsurprising. We might expect densification in information networks such as citation networks where paper authors serve as nodes and an edge represents one author citing another. However, due to social capacity this result is not expected for social interaction networks. In Chapter 5, I conduct an analysis on two series of network snapshots and show that both densification and diameter shrinking can be introduced into social networks by failing to include a mechanism for edge deactivation when constructing the networks.

While considering the need for edge deactivation when constructing network snapshots over time, I began to investigate the impact of network roles in social network dynamics. Roles can be considered latent factors which represent structural positions and node behaviors. Nodes can hold multiple roles simultaneously, and in a dynamic setting the roles of a node may change over time. In this way, roles capture patterns of behaviors that impact the evolution of a network. Extending the work of [8] and [9], I developed a methodology for identifying roles over temporal network snapshots and discovered the existence of persistent roles in two distinct datasets. The presence of shared persistent roles across both time and datasets was surprising and suggests that roles can serve as a basis for defining dynamic network structure. These results are presented in Chapter 6.

Role and community structure transitions are promising components both for general generative models of dynamic networks and specific prediction tasks such as community evolution prediction. While community evolution prediction is an active area of research, the literature has focused on providing alternative frameworks for labeling community evolution events across series of network snapshots. In effect, these various frameworks propose modifications to the definition of the community evolution prediction task in order to improve prediction performance when using standard classification models. To my knowledge,

there has been no prior work that introduces a classification model developed for community evolution prediction. In Chapter 7 I present a graph neural network (GNN) model that incorporates group-node attention and temporal information for community evolution prediction. This Group-Node Attention Network (GNAN) model is demonstrated to outperform baselines used for community evolution prediction.

1.2 Challenges

The tasks of community detection, role discovery, and community evolution prediction on social networks share several challenges. While repositories of network data [10, 11] are available there is often additional preprocessing required to construct a dataset for specific tasks. Ground truth for various machine learning and data mining tasks on networks is generally unavailable. There has been some effort in generating ground-truth communities which can be used to evaluate community detection methods. While this is generally considered an acceptable attempt for objective evaluation of community detection, there is evidence indicating that ground-truth communities defined according to metadata do not map to structural communities observed in networks [12, 13].

Similarly for role discovery and community evolution prediction, there are no ground-truth roles or community evolution events that can be used for evaluation. Evaluation of role discovery methods usually involves some form of qualitative analysis. In the case of community evolution prediction, since the various frameworks include a definition of community evolution events we can use these labels for evaluation. However, different network features are captured by the event labels depending on the labeling function used by the framework and the community evolution event labels from previous snapshots are used in training prediction models. In this way, community evolution prediction frameworks can be viewed as a form of feature engineering through obscured aggregation of features.

There are various types of networks and in [14] they are organized into four top-level categories: technological, information, biological, and social. As each type of network captures different relationships between different categories of objects, it is important to closely

examine the assumptions of models or findings. Assumptions made during the construction of a network dataset can introduce artifacts which alter the semantics of network nodes or edges. For example, consider social interaction networks without edge deactivation. Without edge deactivation a social interaction network becomes a history of all interactions rather than a temporal snapshot of a dynamic network.

1.3 Problem Statement

The overall hypothesis of this work is that node roles, communities, and their interactions form the building blocks of social networks; and these building blocks provide an effective framework to study and understand the evolution of network dynamics. The research areas of community detection, role discovery, and community evolution prediction are then each critical components to improving our understanding of network dynamics. Our understanding of network dynamics then relies on improvements in our ability to define communities, discover roles that characterize node behaviors, and learn models of structural change in both communities and roles. My work explores this hypothesis while developing methods that improve the state-of-the-art of community detection, role dynamics, and community evolution prediction.

The hypothesis in Chapter 4 is that groups of nodes with edge density are evidence of communities where community members have similar attributes within a subset of feature dimensions. Community detection can then be seen as jointly inferring community membership along with a feature distribution—or topic— associated with each community. Chapter 5 presents the hypothesis that a lack of edge deactivation when constructing networks from social interaction datasets can lead to the introduction of temporal artifacts. In Chapter 6, the hypothesis that node behaviors across temporal snapshots can be expressed through a set of roles and role transitions is explored. Finally, Chapter 7 considers the hypothesis that learning a group representation from individual node features according to group-node attention can improve performance of community evolution event prediction over the standard methods used in the literature.

1.4 Contributions

I developed the Seeded Estimation for Network Communities (SENC) method for joint discovery of community members and topics using both node attributes and graph structure [1]. Communities are modeled as connected subgraphs with nodes sharing similar attributes. Nodes may join multiple communities and share common attributes with each. Communities have an associated probability distribution over attributes and node attributes are modeled as draws from a mixture distribution. We make two basic assumptions about community structure: communities are densely connected and have a small network diameter. These assumptions inform the estimation of community topics and membership assignments without being too prescriptive.

I investigated the claims of densification and diameter shrinking in social interaction networks and provided evidence that densification will occur when edge deactivation is not included in the construction of social networks from social interaction data [15, 16]. In certain settings, densification and diameter shrinking can be viewed as temporal artifacts introduced by data preprocessing methodology. Determining the cause of topological effects such as densification is important as models developed for community detection, link prediction, and other network-related tasks are often designed with particular assumptions about network and community structure.

Users in online social networks often have very different structural positions which may be attributed to a latent factor: roles. I performed an analysis on dynamic networks from two datasets (Facebook and Scratch) to find roles which define users' structural positions [17]. Each dynamic network is partitioned into snapshots and I independently find roles for each network snapshot. I developed a role discovery methodology and investigate how roles differ between snapshots and datasets. Six persistent roles are found and user role membership, transitions between roles, and interaction preferences are analyzed and presented.

Existing work in community evolution prediction has focused on developing new frameworks that label events occurring between pairs of communities in consecutive network snapshots. Rather than propose an alternative framework, I developed the Group-Node

Attention Network (GNAN) model which uses group-node attention to learn a representation of a community that is optimized for the prediction of future community evolution events. I built several series of network snapshots and found the accompanying communities and their community evolution events. These datasets are used in a comparative evaluation that demonstrates the improved performance of GNAN over the standard baseline classifiers used for the community evolution prediction task.

The contributions of my dissertation are summarized as follows:

Seeded Estimation of Network Communities (SENC)

- A scalable probabilistic method for simultaneously finding highly interpretable community topics and node memberships.
- A flexible and intuitive method of influencing community estimation through the use of bounded seed groups.
- The introduction of several datasets with ground-truth communities used for comparative experiments with top-performing methods.

Temporal Artifacts from Edge Accumulation

- An analysis on the effects of edge accumulation on social interaction networks and the associated network artifacts that are introduced.
- Evidence which indicates densely overlapping communities are the result of unchecked edge accumulation.
- Findings that provide guidance for data preprocessing methodology and design of network models.

Persistent Roles

- A methodology for discovering and tracing persistent roles over a series of network snapshots.
- The discovery of six persistent roles: *popular*, *friendly*, *explorer*, *reciprocated*, *community member*, and *active-community member*.

- An analysis of role membership, transitions between roles, and interaction preferences.

Group-Node Attention Network (GNAN)

- A graph attention network model using group-node attention capable of learning a community representation.
- Construction of two series of network snapshots with communities and community evolution event labels for comparative evaluation.
- A comparative evaluation of GNAN against standard baseline methods used in the community evolution event prediction literature.
- An analysis of the effects caused by using increasingly older network snapshots for training.

Chapter 2: Background

2.1 Clustering

A substantial proportion of community detection techniques do not use node attributes to detect communities or provide per-community feature distributions as output. Many solely rely on graph structure [18] or independently group objects by topics and structure [19]. The state-of-the-art methods for community detection have introduced linking models, subspace clustering, topic models, and heterogeneous networks to improve performance and simultaneously estimate topics and membership.

The SENC method presented in Chapter 4 is most similar to subspace clustering and topic models—both are further discussed here. SENC assumes community members are similar across a subset of the feature space and considers node feature values to be drawn from per-community feature distributions.

The recent literature on linking models which incorporate node attributes [2, 20] shows promising results. I aim to perform competitively with those methods by taking a different approach which is probabilistic but allows the use of heuristics to select seed groups.

Other literature [21] has focused on topic models for heterogeneous information networks. While SENC is more general and does not require customization to support multiple types of nodes, extra information provided by those networks can be provided through additional node features or edges.

2.1.1 Subspace Clustering

Subspace clustering is used to find clusters of objects that occur when the objects are embedded in a subset of the feature space dimensions. A survey of subspace clustering methods is provided in [22] which categorizes various approaches. Subspace clustering is

frequently used on high-dimensional datasets and can be viewed as online feature selection for clustering [23].

A major challenge of subspace clustering is finding the optimal subspace clusters. A naive approach would exhaustively try every combination of features, but this is computationally infeasible for all but the smallest datasets. SENC is able to determine which features are relevant to each community by finding the maximal likelihood for the target community's feature distribution in the context of the mixture distribution which describes the node.

The work in Chapter 4 extends research on subspace clustering in networks by introducing the use of probability distributions to describe the observed features and to estimate community topics and memberships. We view communities as having feature distributions which represent a common interest of all members.

2.1.2 Topic Models

Topic models are probabilistic models used to find the semantic structure of documents [24] and frequently make assumptions about the relationships between topics, objects, and words. Some models support multiple topics per object or topic hierarchies, but the model is built with those assumptions. Topic models have been designed for networks which group related objects dependent on network structure [25].

The methods combining topic models with graph clustering tasks such as community detection are limiting. They either involve complex models which are only applicable to specific datasets or they independently find topics by treating vertices as documents and then attempt to fit the topics onto the graph to find clusters [19].

In Chapter 4, I represent node attributes as term-weight vectors and associate a topic with each community. Every cluster found is a community, and each community has a single feature distribution or topic. We can then estimate a node's membership to a community by finding mixture weights which best explain the node's feature values through community topics.

2.2 Social Network Analysis

2.2.1 Terminology

There are several terms which may be used interchangeably in this dissertation. A *network* is the same as a *graph*. *Graph* is more commonly used in formal definitions while *network* tends to be used when speaking of a graph in a data analysis context. *Vertex* and *node* are similarly associated with formal definitions and data analysis. *Group* and *community* both refer to a subset of vertices in a graph which exhibit community-like structure—for some definition of community-like—according to their corresponding edges.

2.2.2 Network Features

There are various statistics and measures used to characterize networks, nodes, and node subsets. I present a selection of such features here that are pertinent to this work. Additional information can be found in [14] and I use similar notation here. A network, or graph, $\mathcal{G}$ can be defined as a set of vertices $\mathcal{V}$ and edges $\mathcal{E}$. Edges are pairs of vertices (i, j) which may be unordered or ordered—ordered pairs are required to represent directed networks. Adjacency matrices provide an alternative representation of edges. A basic adjacency matrix $\mathbf{A}$ has elements A_{ij} where

$$A_{ij} = \begin{cases} 1 \text{ if vertices } i \text{ and } j \text{ are connected,} \\ 0 \text{ otherwise.} \end{cases} \tag{2.1}$$

Edge weights are commonly used to quantify the strength of an edge. The matrix $\mathbf{A}$ will be symmetric for undirected networks and can be extended to represent weighted edges by storing the edge weight as element A_{ij}. The *degree* of a vertex refers to the total number

of connected edges and can be defined as

$$k_i = \sum_{j=1}^{n} A_{ij} \,. \tag{2.2}$$

In a directed network, the definition of degree can be further extended to *in-degree* and *out-degree*, where each respectively refer to the number of incoming and outgoing edges. In a social network where edges correspond to a social interaction, the in-degree can be seen as a measure of popularity. Similarly, the out-degree is the total number of nodes interacted with and has a theoretical limit known as *social capacity* which is discussed in Section 2.2.3. When working with directed edges, a notion of *reciprocity* can be considered for an individual node in an unweighted, directed network by

$$\text{reciprocity}(i) = \frac{\sum_j A_{ji}}{\sum_j A_{ij}}, \forall j \mid A_{ij} = 1 \,. \tag{2.3}$$

A global definition of reciprocity may also be calculated as the ratio of all reciprocated edges over the total number of edges in the network.

A network *path* is a sequence of vertices where each vertex is connected to the next. A *shortest path*—or *geodesic path*—refers to a shortest path between two vertices. Multiple shortest paths may exist for a pair of vertices. The *shortest distance* (*geodesic distance*) refers to the length of the shortest path. The *diameter* of a network is the length of the longest shortest path.

Connected components are defined as the maximal subset of vertices such that there is a path between every pair of vertices in the component. Some networks may have only a single component. In social networks that consist of multiple components it is common to find a largest connected component that is much larger in number of nodes than all other components. The concept of network diameter can be applied to individual components in a network. A *clique* is a set of vertices where each vertex i is connected with every other

vertex j in the set and the size of a clique is the number of member vertices. A *maximal clique* is a clique such that there are no other vertices in the graph which can be added to create a larger clique. The term *k-clique* is used to refer to cliques with k vertices.[1]

There are additional group- and node-level features used in this dissertation to characterize network topology. We can measure group cohesion in terms of *group affinity* and *group density*. Group affinity captures the preference for interacting within a group compared to the rest of the network and is calculated by findng the ratio of the number of edges between group members over all the edges involving any group member

$$\frac{\sum_{ij} A_{ij} \mid i,j \in \mathcal{C}}{\sum_{ij} A_{ij} \mid i \in \mathcal{C}, j \notin \mathcal{C}}, \tag{2.4}$$

where $\mathcal{C}$ is the set of nodes in a group. Group density refers to the density of edges in a group and is calculated by finding the ratio of the number of edges between group members and the total possible number of edges in the group based on the number of group members $C = |\mathcal{C}|$ in

$$\frac{\sum_{ij} A_{ij} \mid i,j \in \mathcal{C}}{C(C-1)}. \tag{2.5}$$

Centrality measures are used to quantify the centrality of a node in a network. Two such measures are *PageRank* score and *betweenness centrality*. PageRank was introduced in [26] for calculating the relevance of web sites and improves upon previous centrality measures that incorporate node degree as a measure of importance for each node. The intuition is that web sites that are preferred sources of information will be linked to more often. Then links from one of these popular, or authoritative, web sites should contribute more to the centrality score. In a social network setting, PageRank will assign a higher score to a node if it is popular or a popular node interacts with it. However, the contribution of a popular node to its neighbors is proportional to its out-degree. The PageRank scores for vertices is

[1] An alternate definition of k-cliques which correspond to a different concept is given in [14].

defined as

$$\mathbf{x} = \mathbf{D}(\mathbf{D} - \alpha\mathbf{A})^{-1}\mathbf{1}\,, \tag{2.6}$$

where $\mathbf{D}$ is the diagonal matrix with elements having value $\max(d_i^{\text{out}}, 1)$, α is a constant less than 1, and $\mathbf{1}$ is a vector of ones. The term d_i^{out} is the out-degree of node i. Note that in the case of $d_i^{\text{out}} = 0$, we can use any non-zero value instead to prevent division by zero without changing the resulting value.

The betweeness centrality assigns higher scores to nodes which more often occur on the shortest paths between node pairs in the network. Betweenness centrality can be viewed as a proxy for measuring node influence. The score for node i is calculated by summing the proportion of shortest paths between all pairs of nodes that pass through node i,

$$x_i = \sum_{st} \frac{n_{st}^i}{g_{st}}\,, \tag{2.7}$$

where n_{st}^i is the number of shortest paths between nodes s and t that include node i and g_{st} is the total number of shortest paths between nodes s and t.

2.2.3 Network Dynamics

The concepts of *social capacity* [27] and *bursty communications* [28,29] have been considered separately and more recent literature [30–32] has attempted to measure and use these to determine the state of edges in a large social network.

Social capacity captures the maximum number of relationships one prefers to maintain at any given time and there is evidence that social capacity is conserved over time [30,33,34]. The term bursty is used to describe the temporal patterns of social interactions between pairs of nodes. That is, humans tend to interact in bursts and these patterns must be considered in order to correctly identify the activation/deactivation of edges.

The observation of social capacity and burstiness of human interaction in some networks suggests careful consideration is required to construct accurate static views of these

networks. In fact, accepted claims of graph evolution [35, 36] appear to fail when graph series are constructed based on timestamped interactions rather than accumulated without regard for edge deactivation.

Previous literature [37] introduced densification and diameter shrinking as common network characteristics and we briefly describe them here. Densification is the super-linear growth of edges relative to nodes and results in a network becoming denser over time. Diameter shrinking is the reported tendency for network diameters to decrease over time as more edges are accumulated. We can see both how densification contradicts the notion of social capacity and might account for diameter shrinking.

2.2.4 Network Roles

In [9], the role membership for a series of network snapshots are found and analyzed and the roles are used to construct a transition model of role memberships. Every node in every snapshot is represented as a mixed membership of roles. This mixed membership may change over time and a transition model captures the likelihood of transitioning between roles. Their method assumes roles are stationary and uses the same set of basis vectors (roles) for every snapshot rather than directly estimate roles from each snapshot. The authors suggest roles may generalize over time and across datasets, but do not provide support for this statement. To my knowledge, the work in Chapter 6 is the first to present evidence of common, persistent roles derived directly from data.

Some models which incorporate roles do not distinguish between node features derived from network structure and those external to the graph. In [38], a probabilistic model which incorporates node features as dependent on latent factors (roles) is introduced. While these features could be derived from network structure as described in [39], the experiments performed in [38] only include external features such as document terms and voting counts. The network topology is ignored.

Communities provide extra structural information which can benefit role discovery. In both [40] and [41], communities are simultaneously detected with roles. Roles are used as

latent factors of which node attributes are dependent.

Finally, [25] and [42] add roles to topic models where authors may take a role when generating a document and the topic of the document is dependent on the author's role.

An overview of role discovery approaches is provided in [39] which discusses graph-based, feature-based, and hybrid definitions of roles and methods for their discovery from graph and node-attribute data. They show that feature-based roles are more flexible and capable of capturing more complex roles. A framework for feature-based role discovery is introduced and discusses classes of approaches for role feature construction and role assignment.

The use of non-negative matrix factorization (NMF) for discovering node roles was introduced in [8]. In that paper, the authors use a method [43] to generate features which aggregate various per-node structural attributes. This node-attribute matrix is then decomposed using NMF and the resulting basis vectors correspond to node roles in the network. Later work adds additional constraints to NMF which can be used to specify expectations of sparsity or diversity of the roles [44]. The work in Chapter 6 differs as I discover persistent roles across datasets and time using independent decompositions of network snapshots.

Other work [45, 46] uses role-labeled nodes to identify the roles of unlabeled nodes. However, the roles in their work are not defined in terms of structural positions in the network but rather functional occupations in an organization (e.g., roles held in technology companies: research & development, executives, and human resources). That is, the roles are defined in terms of domain knowledge and non-structural node features. The authors then introduce a classifier for these functional roles which incorporates information derived from the network structure.

Aside from identifying patterns of structural positions, roles have also been used in the context of information cascades to identify groups of nodes which have similar influence and blockage attributes [47].

A feature-based approach for automatic detection of user roles in online forums is presented in [48]. Their method uses principal component analysis (PCA) and agglomerative

clustering of feature profile data to find roles; where each cluster corresponds to a role. Another feature-based approach using a mixture model of roles is presented in [49]. Nodes are first clustered using node features derived from the network structure and then a qualitative assignment of nodes to roles follows.

2.2.5 Community Detection

Community detection [50] is the task of clustering nodes in a network according to one or more criteria that indicate cohesion. Communities can be defined in terms of network structure alone [4] or include node attributes [1, 2, 51, 52]. Additionally, methods [53, 54] which operate on heterogeneous networks that include multiple kinds of nodes or edges have been proposed.

Communities can be partitions of a network where each node is a member of a single community, or communities may overlap. This choice is dependent on the type of network being processed, the limitations of the community detection method, and the real-world applications for the discovered communities.

While it was earlier proposed that high network modularity was a defining characteristic of communities [4,36] presented evidence that edge density is highest in regions of networks where communities overlap due to densification of the network. Since then, [15] showed that accounting for edge decay in social networks prevents densification. Additional information on this topic can be found in Chapter 5.

Earlier work in community detection focused on finding static communities in a single network. However, later work has considered how to extend community detection to a dynamic setting [55, 56]. The task of dynamic community detection expands community detection to include community tracking as a component of community detection. Rather than predicting future community evolution events, dynamic community detection methods attempt to find definitions of communities according to multiple criteria which includes adjusting previous community definitions according to additional information such as link activation and deactivation among nodes in the dynamic network. The introduction of the

temporal dimension to community detection requires not only finding communities in a network but also tracking the evolution of those communities over time. Nodes may enter and leave the network, new edges are created, and old edges are deactivated.

An alternative approach to dynamic community detection is to use a static community detection algorithm on a time series of network snapshots. The communities found in consecutive snapshots can be associated with one another based on community membership [57–60]. The tracking of communities over time and association of event labels with the changes in community membership is the setting for community evolution prediction discussed in Section 3.3.

2.3 Graph Neural Networks

Advancements in neural network architectures are being applied to various machine learning and data mining tasks on graphs and there are several surveys [61–63] which provide an overview of the various model architectures and their applications in graph classification [64], graph embedding [65], node embedding [64, 66], link prediction [67], graph generation [68, 69], and heterogeneous networks [70].

Complete surveys of GNNs can be found in [62] and [63], these surveys organize GNN architectures into categories according to model assumptions and purpose. The authors of [63] neatly define five top-level categories: *graph recurrent neural networks*, *graph convolutional networks*, *graph autoencoders*, *graph reinforcement learning*, and *graph adversarial methods*. The graph attention network model proposed in Chapter 7 falls under the graph convolutional networks category.

Graph convolutional networks (GCNs) generally accept both node attributes and the adjacency matrix as inputs and are often used for classification tasks, clustering, and graph reconstruction. There are many different approaches to performing convolution or aggregation operations on graphs and we can further categorize graph convolutional networks by

the type of convolution operation used: *spectral* or *spatial*. Models using spectral convolution [71, 72] transform node representations into the spectral domain. This transformation may result in node representations that include contributions from nodes that are distant in the input graph (spatial domain). There has been work to constrain spectral convolutions to account for locality in the spatial domain [71, 73] and generalize GCNs using a message-passing framework [74]. However, GCNs are still limited in expressive power due to the use of non-parametric weights used during aggregation instead of the the trainable parameters used to compute attention coefficients in graph attention networks [62].

The introduction of attention—through graph attention networks (GANs) [64]—has been especially useful for working with graph data due to supporting both variable-sized inputs and trainable, independent aggregation weights. Attention mechanisms enable a GNN to learn the significance of related graph objects when using aggregation to compute a hidden representation of a particular graph object. For example, attention can be used to learn graph node embeddings for each node, based on the attributes of adjacent nodes. Through the use of an appropriate loss function and backpropagation, the learned node embedding is optimized for the specific task of the neural network. While attention mechanisms are frequently used for *self-attention* where the attended objects and the object used to compute the attention coefficients are the same, attention can be used anywhere weighted aggregation is needed. In [75], self-attention is used at the node-level, but a *semantic attention* layer is used to learn the importance of distinct node embeddings that are associated with different views of a heterogeneous graph. Additional information about attention mechanisms is provided in Section 2.3.1.

As part of this dissertation, a graph neural network for community evolution prediction is presented in Chapter 7. This model, the Graph-Node Attention Network (GNAN), is a graph neural network that uses *group-node attention* to learn a group representation for predicting community evolution events. Group-node attention enables the representation of a community to be learned by using group-level features to attend to individual nodes associated with the community. While graph neural networks have been used for a variety of

tasks, to the best of my knowledge there is no existing work using graph neural networks for community evolution prediction or employing group-node attention. Additionally, GNAN utilizes the defined communities required for the community evolution prediction task, as well as a group-relative positional information, to provide a relaxed masking over nodes in the network.

2.3.1 Attention Mechanisms

Attention mechanisms in neural networks enable a model to learn which objects and features are relevant to a given task from a potentially variable-sized number of input objects. Attention can be viewed as a weighted mean where the weights (*attention coefficients*) are learned and the output of an attention layer is a representation of each input object according to an independent weighted mean of related objects.

Implementations of attention vary, but the formulation of attention from the Transformer model proposed in [76] is commonly used and includes a scaled dot-product attention represented in terms of query, key, and value matrices: $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$. *Self-attention* is a common use of attention where the representation of an object depends on the features of attended objects of the same type. This serves as an alternative to recurrent neural networks (RNNs) for learning representations of individual objects according to some context which includes potentially-related objects. Attention is frequently used in natural language processing to learn parts of speech and other semantic concepts useful for tasks such as natural language translation where each of the input objects is a word embedding.

In the self-attention setting, the $\mathbf{Q} \in \mathbb{R}^{n \times d_k}$, $\mathbf{K} \in \mathbb{R}^{n \times d_k}$, and $\mathbf{V} \in \mathbb{R}^{n \times d_v}$ matrices are projections of some input matrix $\mathbf{X} \in \mathbb{R}^{n \times d_{\text{model}}}$, where n is the number of objects, d_{model} is the length of the input, d_k is the key length, d_v is the value length, and each row of $\mathbf{X}$ is the feature vector of an individual object (e.g., sentence term or graph node). The $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ matrices are computed multiplying $\mathbf{X}$ by learned model parameter weight matrices

$\mathbf{W}^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $\mathbf{W}^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, and $\mathbf{W}^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$,

$$\mathbf{Q} = \mathbf{X}\mathbf{W}^Q, \tag{2.8}$$

$$\mathbf{K} = \mathbf{X}\mathbf{W}^K, \tag{2.9}$$

$$\mathbf{V} = \mathbf{X}\mathbf{W}^V. \tag{2.10}$$

The attention defined in terms of these matrices is

$$\boldsymbol{\alpha} = \mathsf{softmax}(\frac{\mathbf{Q}\mathbf{K}^\mathsf{T}}{\sqrt{d_k}}), \tag{2.11}$$

$$\textsc{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \boldsymbol{\alpha}\mathbf{V}, \tag{2.12}$$

where $\boldsymbol{\alpha}$ are the attention coefficients.

Attention layers can be stacked in self-attention; that is, the output of a previous attention layer is used as the input to the next attention layer and each attention layer has its own weight parameters. A common variant of attention is *multi-head attention*. An attention head corresponds to an individual attention function in multi-head attention and each head has its own weight parameters. The outputs of the individual attention heads need to be combined before a final layer output is produced. Concatenation and projection are used in [76] but other models [64] average the output of each head. Though attention is often used for self-attention, it is possible to use different types of objects for defining the query matrix $\mathbf{Q}$ than those used for defining the key matrix $\mathbf{K}$ and value matrix $\mathbf{V}$. The group-node attention defined in Chapter 7 is an instance of using attention with different types of objects.

Chapter 3: Related Work

3.1 Community Detection

There are many approaches to community detection and the state-of-the-art methods which use both network structure and node features are based on linking models [2,20,77], heuristic clustering [52, 78–80], topic models [25, 81], or graph neural networks [82–84]. Previous work [85] has also considered initialization with candidate communities. There has also been extensive work in community detection on attributed graphs [51]. Additionally, work has been done on avoiding community detection methods [86] and incorporating outlier detection in community detection [87].

Linking models estimate the probability of links and node attributes. They are similar to block models [88,89] with link probabilities dependent on node attributes and community membership. Recent implementations are efficient and competitively find communities, but treat node and community features as binary values [2]. This results in a poor representation of the community's shared interest or topic.

There have been attempts at extending clustering methods to support network data, such as subspace clustering [52, 78, 79]. In contrast to linking models, these methods do not model edge probability and instead use observed edges and node attributes to identify dense, connected subgraphs with similar node attributes over a subset of the feature space. These methods are not probabilistic and rely on heuristics for detecting nodes with similar attributes. Further, they find many duplicates of a single detected community and require a distinct post-processing step to identify the optimal detected communities.

Topic model approaches extend basic models such as LDA [90] to estimate latent factors and introduce a dependence of edges on the latent factors. These models are generative and

require a task-specific probabilistic graphical model. In the past they have been difficult to scale up for larger datasets due to the sampling methods on which they rely [91].

With the increased interest in deep learning, graph neural networks (GNNs) have been applied to the community detection problem [82–84]. These approaches learn embeddings for either nodes or edges and incorporate positional information, node attributes, or graph convolutions. Experiments indicate that these models can outperform existing methods, but due to architectural limitations several [83, 84] are only feasible for use with smaller network datasets.

3.2 Role Discovery

There is existing research on role discovery through matrix decomposition of node attributes [8, 9] and calculating the transition probability between roles over time in a consecutive series of graphs [9]. There are several problems with this work: a mixture of network types (e.g., computer, affiliation, social, and social interaction) are used for experiments, but there is little discussion on the relationship between discovered roles and network type; no explanation is provided for the length of the time windows used to construct the graph series; and, no analysis of features captured by roles and how these roles map to known *social roles* [92] which have structural definitions.

Each of those problems is remedied by my past and current work and will enable me to approach the problem from a theoretically sound position and focus on social roles within dynamic social interaction networks.

The term *role* in the context of [8,9] has a different definition than its classical usage in social network analysis [92]. In these recent works, roles are found through a decomposition of a matrix representing node attributes. The roles are then basis vectors which are used to reconstruct the original attribute matrix and do not necessarily correspond to social roles. The results of [8] do demonstrate that basis vectors can correspond to roles, but more investigation is needed to determine the universality of the findings and whether it

applies to social interaction networks.

3.3 Community Evolution Prediction

There has been substantial work on community evolution event prediction and community evolution in the last 10 years that have provided frameworks for tracking evolving communities across network snapshots [58, 93, 94].

The development of additional community tracking frameworks is often motivated by improving performance in predicting community evolution events associated with changes in community structure. Not all frameworks use the same set of evolution events, but the events present in [95]—one of the first community tracking frameworks used for community evolution prediction—can either be mapped to events in other frameworks or are explicitly dropped. This standard set of events include: *continuing*, *dissolving*, *growing*, *merging*, *shrinking*, and *splitting*.

Given a series of network snapshots over time, the general methodology of the frameworks involves performing community detection on each snapshot network, finding relationships between communities in adjacent snapshots, and then using the relationships and additional network features to define community evolution events.

I use the Group Evolution Discovery (GED) [95] framework in Chapter 7 to find community evolution events, but the model proposed there can be used with any community tracking framework. Selecting an appropriate community tracking framework is then analogous to feature engineering. Many traditional classification techniques (decision trees, SVM, etc.) have been used in previous works [57, 58, 96] which include experiments showing the relative performance of traditional classification techniques when applied to communities and events defined according to the community tracking framework being proposed in each paper. The community evolution prediction literature also often include experiments that evaluate the contribution of engineered features [96, 97]. However, these previous works do not introduce new classification models and instead alter the community evolution event labeling function. The redefining of the event labeling function acts as a form of feature

engineering where the definitions of community evolution events are crafted such that prediction performance is improved.

Chapter 4: Finding Community Topics and Memberships

4.1 Introduction

Given a graph of self-organizing objects, we wish to estimate the latent topics around which the objects organize and discover community membership. We hypothesize groups with high edge density in graphs are evidence of communities whose members have similar attributes within a subset of the feature dimensions. The work in this chapter has been published in [1].

In this chapter I present Seeded Estimation of Network Communities (SENC). SENC is a probabilistic method which uses both node attributes and graph structure to simultaneously estimate community feature distributions and members. We assume a community may exist around seed groups in the network. Many community detection methods build strong assumptions regarding community features into their models, which limits generalizability. SENC provides a flexible means of accounting for a variety of community structures through the use of configurable lower and upper bounds on discovered communities. The seed groups define the lower bounds, and they may in turn be defined by network structure or node and edge attributes. In the experiments presented in this chapter, we consider every maximal k-clique in the network to be the core of a partially defined community. The upper bounds provide an intuitive way to incorporate knowledge about the degree of clustering in the network. Nodes may be members of multiple communities and communities may overlap.

Communities are defined by the associated distribution (topic) and a set of member nodes. Every seed group corresponds to a community, and the initial feature distributions are a weighted average of the seed members' attributes. We use the features of nodes in each group to compute initial estimates for the community feature distributions (topics). We then find initial estimates of the membership weights given these estimated per-community

topics. After this initialization, membership weights and community feature distributions are iteratively updated. The feature distributions are updated by aggregating attributes of community members and finding the maximum likelihood of a mixture distribution where the parameters for all other communities are fixed.

4.2 Seeded Estimation of Network Communities

Network communities indicate interaction and attraction among members which is not shared by non-members. The nature of the interaction may be reflected in node attributes and we would expect for member nodes to be similar to one another. However, nodes may participate in multiple communities and the members of each community may be similar to each other in different ways.

To provide motivation for the proposed method, let us discuss an example using an unspecified online social network. This social network allows users to join discussion areas for topics such as "computer science" or "coffee." Suppose a user is interested in both CS and coffee and participates in both communities. We expect the user's posts to the CS community will be different from her posts to the coffee community. We also expect the user's post in the CS community will be more similar in content to other posts in the CS community than to most posts in the coffee community.

Now assume we do not have access to individual user posts. Instead we have aggregated word counts for each user and we do not know which post contained which words. We can model a user's word frequencies as a random variable drawn from a multinomial distribution. Since each user may belong to different communities or have different levels of involvement then it's necessary to use a different multinomial distribution for each user. As previously hinted, we expect posts within a single community to have similar word frequencies. If we knew those per-community word distributions we could then represent each user's word distribution as a mixture distribution. This is akin to standard topic models such as LDA [24].

The Seeded Estimation of Network Communities (SENC) method described here has an

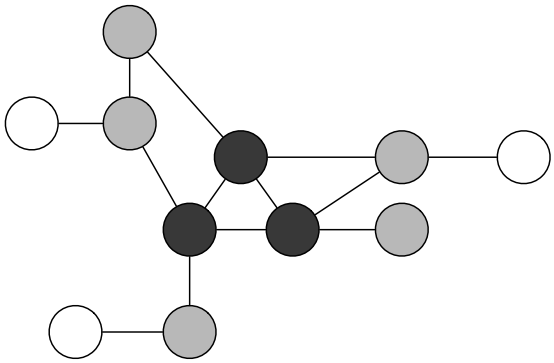


Figure 4.1: Lower and upper bounds for a seed community. The lower-bound nodes are black, upper-bound nodes are grey, and excluded nodes are white.

advantage over state-of-the-art community detection methods in its exploitation of network structure to regularize and guide estimation. This is possible through the use of *seed groups*. A seed group is a subgraph with properties which indicate the nodes are a subset of a community.

Each seed group is considered to be a lower bound of a community and its members are representative of this corresponding community. The lower-bound members, or *seed members*, influence estimation; the members' attributes are used as the initial estimate for the corresponding community's word distribution. The community topic is updated as additional member nodes beyond the lower bound are found.

Along with a lower bound, each seed community has a corresponding upper bound. The definition of this upper bound can be dependent on the network and its selection guided by simple network statistics such as the clustering coefficient. Figure 4.1 depicts an example of the bound sets for a seed community where the distance of a node from lower-bound members is used to define the upper-bound set. The bounds serve as a gentle bias to flexibly model assumptions regarding the shape of communities in a network.

Table 4.1: Definition of notation.

N	number of nodes
C	number of communities
D	number of feature dimensions
$\mathbf{\Phi}$	community topics, $C \times D$ matrix
$\mathbf{\Theta}$	community memberships, $N \times C$ matrix
$\mathcal{G}(\mathcal{V}, \mathcal{E})$	graph defined by vertices and edges
$\mathcal{S}_{c=1...C} \subseteq \mathcal{V}$	members of community c
$\mathbf{x}$	attributes of a node

4.2.1 Notation

Before continuing it is useful to introduce notation and additional terms for describing the proposed method. We use *topic* to refer to the characteristic features of a community as well as the associated probability distribution parameters for all C communities, $\mathbf{\Phi}$, where each row $\mathbf{\Phi_{c,*}}$ is a parameter for the categorical distribution associated with community $c = 1, \ldots, C$ with length D, the number of feature dimensions.

The node attribute vector $\mathbf{x}$ is a D-length vector of continuous, node feature values. A *membership weight vector* or *membership vector* is denoted as $\mathbf{\Theta}_{n,*}$ and refers to the probability weight vector associated with node n over all C communities. The individual membership vectors make up the N rows of the membership matrix $\mathbf{\Theta}$. The membership weights indicate the proportion of node features which are attributed to each community. For quick reference, basic notation used in the equations is available in Table 4.1.

4.2.2 SENC Model

SENC uses an EM algorithm to find the maximum-likelihood estimates for community topics and node memberships. Per-node community memberships are estimated as weighted counts of observed feature values given the community topics in the E-step and per-community topics are maximized in the M-step.

Node memberships for each node n participating in a seed group, $n \in \bigcup_{c=1...C} \mathcal{S}_c$, are estimated using the community topics. We represent the feature values of a node $\mathbf{x}$

as being drawn from a mixture distribution with per-node mixture weights $\mathbf{\Theta}_{n,*}$ over all community distributions $\mathbf{\Phi}$ using per-community topic distributions $\mathbf{\Phi}_{c,*}$. A single term for a node n is drawn by first selecting a community c with probabilities $\mathbf{\Theta}_{n,*}$ and then choosing a specific term with probabilities $\mathbf{\Phi}_{c,*}$. For the data discussed in this chapter, the community feature distributions are categorical distributions and node features $\mathbf{x}$ are generated by multiple trials of a mixture categorical distribution with proportions $\mathbf{\Theta}_{n,*}\mathbf{\Phi}$. A multinomial distribution is a categorical distribution with multiple, independent trials. We refer to the per-node feature distributions as multinomial distributions.

Nodes may be members of multiple communities and node features will then be characteristic of multiple community topics. In order to untangle the features characteristic of a community from those belonging to adjacent communities we define a mixture categorical likelihood function. This is the standard likelihood function but with the event probability vector $\mathbf{p}$ parameter computed as the matrix product of some $1 \times C$ mixture vector and $C \times D$ per-community topics matrix: $\boldsymbol{\theta}_{n,*}\mathbf{\Phi}$.

To improve readability we introduce $\boldsymbol{\gamma}$ as the sum of feature values from community members $\mathcal{S}_c$:

$$\boldsymbol{\gamma} = \sum_{n \in \mathcal{S}_c} \mathbf{x}\,. \tag{4.1}$$

When estimating $\mathbf{\Phi}_{c,*}$ using community members $\mathcal{S}_c$, the mixture vector $\boldsymbol{\theta}$ is a weighted average of membership vectors $\{\mathbf{\Theta}_{n,*} : n \in \mathcal{S}_c\}$ weighted by the proportional number of observations contributed by each node $n \in \mathcal{S}_c$

$$\boldsymbol{\theta} = \sum_{n \in \mathcal{S}_c} \frac{(\sum_d x_d)\mathbf{\Theta}_{n,*}}{\sum_d \gamma_d}\,. \tag{4.2}$$

Using $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$ we can now show how $\mathbf{\Phi}_{c,*}$ may be updated. The event probability vector

$\mathbf{p}$ is the parameter for a categorical distribution:

$$\mathbf{p} = \boldsymbol{\theta}\boldsymbol{\Phi}\,, \tag{4.3}$$

$$= \sum_{i=1}^{C} \theta_i \boldsymbol{\Phi}_{i,*}\,, \tag{4.4}$$

$$= \theta_c \boldsymbol{\Phi}_{c,*} + \sum_{i=1, i\neq c}^{C} \theta_i \boldsymbol{\Phi}_{i,*}\,. \tag{4.5}$$

We can use the factoring of $\mathbf{p}$ in Equation (4.5) with the multinomial expected value to find the maximum-likelihood value of $\boldsymbol{\Phi}_{\mathbf{c},*}$ given the community member observations $\boldsymbol{\gamma}$ from Equation (4.1).

The expected value for a single feature value i in random variable X drawn from $\text{Mult}(\mathbf{p}, n)$ is $E\{X_i\} = np_i$, where n is the number of trials and $\mathbf{p}$ is the event probability vector. If we replace the expected value of each feature dimension with the summation of community members' $\mathcal{S}_c$ attributes $\boldsymbol{\gamma}$ then we can substitute the expected value with the observed value γ_i for feature i and define

$$\gamma_i = (\sum_{d=1}^{D} \gamma_d)\boldsymbol{\theta}\boldsymbol{\Phi}_{*,i}\,. \tag{4.6}$$

If we replace the expected value of each feature dimension with the summation of community members' $\mathcal{S}_c$ attributes $\boldsymbol{\gamma}$ then we can define the maximum likelihood of $\boldsymbol{\Phi}_{c,*}$ as:

$$\boldsymbol{\gamma} = (\sum_{d} \gamma_d)\boldsymbol{\theta}\boldsymbol{\Phi}\,, \tag{4.7}$$

$$= (\sum_{d} \gamma_d)(\theta_c \boldsymbol{\Phi}_{c,*} + \sum_{i=1, i\neq c}^{C} \theta_i \boldsymbol{\Phi}_{i,*})\,, \tag{4.8}$$

$$\frac{\boldsymbol{\gamma}}{\sum_{d} \gamma_d} = \theta_c \boldsymbol{\Phi}_{c,*} + \sum_{i=1, i\neq c}^{C} \theta_i \boldsymbol{\Phi}_{i,*}\,, \tag{4.9}$$

$$\theta_c \boldsymbol{\Phi}_{c,*} = \frac{\boldsymbol{\gamma}}{\sum_{d} \gamma_d} - (\sum_{i=1, i\neq c}^{C} \theta_i \boldsymbol{\Phi}_{i,*})\,, \tag{4.10}$$

$$\boldsymbol{\Phi}_{c,*} = \frac{\frac{\boldsymbol{\gamma}}{\sum_{d} \gamma_d} - (\sum_{i=1, i\neq c}^{C} \theta_i \boldsymbol{\Phi}_{i,*})}{\theta_c}\,. \tag{4.11}$$

Using Equation (4.11) we can easily estimate community topics using node attributes, per-node community membership weights, and the latest topic estimates for other communities.

We use $\boldsymbol{\Phi}'$ to reference a modified version of $\boldsymbol{\Phi}$ with normalized columns, each summing to 1. The per-node community memberships are found by performing a weighted count of node attributes over the communities, where α denotes a normalization scalar

$$\boldsymbol{\Theta}_{n,*} = \alpha \boldsymbol{\Phi}' \mathbf{x}^T\,. \tag{4.12}$$

For each observed term, we assign a proportion of the count to each community according to the relative probability of that term occurring in each community. A community with a higher relative probability of a given term occurring will receive a larger proportion of the count than the others.

4.2.3 Algorithm

The SENC algorithm constructs per-community lower- and upper-bound matrices, initializes per-community topics $\mathbf{\Phi}$ and per-node community memberships $\mathbf{\Theta}$, and then performs expectation-maximization iterations until estimates stop improving or the maximum number of iterations is reached. The algorithm requires the $N \times N$ graph adjacency matrix and the $N \times D$ node attribute matrix as input. The lower-bound matrix is a $C \times N$ binary matrix of the seed members where 1-values indicate node n belongs to community c. The upper-bound matrix is a binary $N \times C$ matrix where 1-values indicate node n may belong to community c. This prevents nodes from distant communities being assigned to communities with a similar topic. The construction of lower- and upper-bound sets for each community is dependent on the network being processed. Two matrices are produced as output: a $C \times D$ community topic matrix and an $N \times C$ community membership matrix. A goal of the proposed method is to remove features representative of overlapping communities over EM iterations. The node membership vectors are estimated using the community topics to perform a weighted count over node attributes. These weighted counts are normalized to sum to one and used as membership weights.

Algorithm 1 Main Program: initialization, EM, termination.

Require: The *graph* and *node attributes*.
Ensure: The *community topics* and *membership*.
1: Construct *lower-bound* and *upper-bound* matrices;
2: Initialize community topics $\mathbf{\Phi}$ and memberships $\mathbf{\Theta}$;
3: **while** Not convergent or max iteration **do**
4: Call `E-step` to update membership $\mathbf{\Theta}$;
5: Call `M-step` to update community topics $\mathbf{\Phi}$;
6: Check for convergence;
7: **end while**
8: **return** Community topics $\mathbf{\Phi}$ and membership $\mathbf{\Theta}$;

After initial estimates are calculated, the algorithm alternatively updates the node memberships and community topics. The per-node and per-community iterations within the E- and M-step are independent and computation may be distributed across multiple threads. The E-step in Algorithm 2 updates the per-node community memberships for all nodes

Algorithm 2 E-step: update per-node community memberships.

Require: The *community topics*, *upper-bound matrix*, and *node attributes*.
Ensure: The updated *membership*.

```
for Each node n do
    Identify which communities influence node n;
    Select topics of influential communities;
    Compute weighted counts from selected topics with Equation (4.12);
    Assign normalized counts to membership vector Θ_{n,*};
end for
return Updated membership Θ;
```

Algorithm 3 M-step: update per-community topics.

Require: The *node attributes*, *membership*, *influence*, and *community topics* from the previous iteration.
Ensure: The updated *community topics*.

```
for Each seeded community c do
    Select all nodes with membership in c;
    Compute weighted average of selected nodes' membership by Equation (4.2);
    Estimate topic with Equation (4.11);
    Assign updated topic to parameter vector Φ_{c,*}, if likelihood improves;
end for
return Updated topics Φ;
```

given the community topics, influence matrix, and node attributes. This is done by computing the weighted counts of node attributes using the probability of each attribute for each community, as shown in Equation (4.12). The upper-bound complexity of the E-step is $O(N\mathcal{C}\mathcal{D})$, where $\mathcal{C}$ and $\mathcal{D}$ are the number of communities to which a node may belong and the number of dimensions relevant to those communities. In practice, $\mathcal{C}$ and $\mathcal{D}$ will be much smaller than C and D.

The M-step, shown in Algorithm 3, updates the per-community feature distributions. We find a new estimate for $\mathbf{\Phi}_{c,*}$ using Equation (4.11) and compare its log-likelihood to the previous iteration's estimate. The new estimate is used if it better explains the feature values of the member nodes. The M-step has computational complexity of $O(C(\mathcal{N}\mathcal{C}+\mathcal{N}\mathcal{D}+\mathcal{C}\mathcal{D}))$, where $\mathcal{N}$ is the number of nodes in the upper-bound set of a community, $\mathcal{C}$ is the number of communities associated with the $\mathcal{N}$ nodes, and $\mathcal{D}$ is the number of feature dimensions relevant to all $\mathcal{C}$ communities and $\mathcal{N}$ nodes. Again, $\mathcal{C}$, $\mathcal{N}$, and $\mathcal{D}$ are usually much smaller

than C, N, and D.

4.3 Experimental Evaluation

We evaluate the proposed model on networks with varying structure to determine whether SENC's results are consistently competitive with state-of-the-art methods. The networks considered are: an NSF research collaboration network, several DBLP citation networks [98], and a Scratch project collaboration network. For comparison, we evaluate the performance of four state-of-the-art community detection methods: CESNA [99], CoDA [3], EDCAR [100], and Link Clustering [101]. CESNA and EDCAR use network structure and node attributes to detect communities; however, the current implementations struggled to process networks with a large number of features. In order to evaluate more methods we elected to use smaller datasets. CoDA and Link Clustering only use network structure.

4.3.1 Datasets

We construct a research collaboration network from NSF awards granted by the Directorate for Computer and Information Science and Engineering (CISE) between January 1995 and August 2014. This is accomplished by forming undirected edges between the PI and co-PIs who received funding from the same award. The awards are associated with programs and we use the programs with at least three associated researchers as ground truth. We find 90% of researchers received funding from six or fewer programs; this suggests programs function well as ground-truth communities. There are a total of 768 programs in the CISE Directorate. NSF awards data is publicly available from the NSF website[1].

An online computer science bibliography, DBLP, contains entries for published papers with information about the authors, citations, and publication venues. The per-year DBLP citation networks were constructed from an existing citation dataset [98] by forming edges between authors who cited each other within that year. Papers are linked to a publication venue and these venues were used to define ground truth. Venues referenced only once

[1] http://www.nsf.gov/awardsearch/download.jsp

Table 4.2: Network statistics. N: number of nodes, E: number of edges, D: number of node attributes, MC: number of maximal cliques with 3+ members, GCC: global clustering coefficient, LCC: average local clustering coefficient, G: number of ground-truth communities.

Dataset	N	E	D	MC	GCC	LCC	G
NSF	8,168	38,212	43,445	3,331	0.590	0.683	429
DBLP 2010	32,961	130,420	58,007	37,120	0.422	0.440	2,288
DBLP 2011	32,614	131,921	56,166	39,955	0.421	0.438	2,215
DBLP 2012	33,576	135,883	54,269	42,443	0.381	0.397	1,861
Scratch	1,714	17,824	36,494	7,705	0.584	0.704	718

were removed from the dataset. Venues with three or more associated authors were used as ground truth.

Scratch [102] is an online community where users may write and share projects (programs) with other users. One way in which Scratch users may interact is by remixing projects. Remixing allows a user to create a copy of any existing project which they may then modify. We create a co-remix affiliation network from the MIT Scratch Team's dataset containing users, projects, and remixes. An edge is formed when two users remix the same project. To reduce the total number of edges we used co-remix edges where users had four or more projects in common. Users may create project galleries which are curated collections of projects. Galleries corresponding to three or more users were used as ground truth. The Scratch dataset used to construct the network may obtained from the MIT Media Lab website[2].

Several of the methods make use of node attributes and these were provided as tf-idf weighted values for EDCAR and SENC and binary values for CESNA. For the NSF CISE network, terms associated with each researcher were taken from NSF award titles and abstracts. The DBLP author terms were taken from titles and abstracts of papers they wrote. Scratch user terms were extracted from titles, descriptions, and tags of their projects. The term features in all networks had stop words removed and terms stemmed.

Multiple connected components were found in all networks and the smaller components

[2]https://llk.media.mit.edu/scratch-data

were removed as they may be trivially considered communities. Table 4.2 lists the network statistics for the largest component of each network used for experiments and analysis. All the networks used for experiments are undirected, but they vary in structure.

As shown in Table 4.2, the NSF and Scratch networks have higher clustering coefficients than the DBLP networks. This is unsurprising as the NSF and Scratch networks are affiliation networks (co-award and co-remix). These experiments show that while SENC is able to perform competitively across all the networks other methods tend to either perform better on networks with higher or lower clustering coefficients.

4.3.2 Methodology

The public implementations of CESNA, CoDA, EDCAR, and Link Clustering were used. CESNA and CoDA rely on an estimate of the number of communities. We provided the number of NSF programs, DBLP publication venues, and Scratch galleries as estimates. CoDA is designed for directed networks but can be used to find communities in undirected networks. It does this by processing the network twice, switching the direction of edges between runs. As a result, two sets of detected communities are generated. We combined both sets when evaluating the performance of CoDA. EDCAR requires 10 parameters and the suggested values from the implementation documentation were used. Link Clustering is parameter-less and only requires the edge list as input. Maximal cliques of size three and above were used as the lower-bound groups for SENC and the upper-bound groups were selected based on the clustering coefficient. The high clustering coefficients of the NSF and Scratch networks indicate tighter upper bounds should be used than with the DBLP networks. For the DBLP networks we extend the lower bounds by including all nodes adjacent to any lower-bound member. The upper bounds for the NSF and Scratch networks are simply the same maximal cliques.

Link Clustering and SENC require a post-processing step to define exact communities. The Link Clustering implementation includes a script to calculate the optimal dendrogram cut threshold and we use this to determine the communities for evaluation. SENC defines

Table 4.3: $F1$ scores for all methods and datasets.

Method	Attr.	NSF	DBLP10	DBLP11	DBLP12	Scratch	Avg.
CoDA	No	0.216	0.278	0.273	0.263	0.283	0.263
Link Clust.	No	0.303	0.266	0.265	0.258	**0.399**	0.298
CESNA	Yes	0.228	0.272	0.263	0.255	0.356	0.275
EDCAR	Yes	0.164	N/A	N/A	N/A	N/A	N/A
SENC	Yes	**0.346**	**0.301**	**0.297**	**0.298**	0.365	**0.321**

community membership with probabilities and does not perform a hard assignment of nodes to communities like the other evaluated methods. We account for this in the evaluation by filtering weaker memberships. For all nodes, we sort their memberships in descending order by weight and take all the assignments until the sum of weights reaches a minimum threshold value. An optimal threshold is used for each dataset.

We use the evaluation function described in [99, 103] and recited in Equation (4.13) to compute the $F1$ score and Jaccard similarity of detected communities against ground-truth communities. This function is especially useful when the numbers of detected communities and ground-truth communities differ as occurs with several of the methods in the experiments. In Equation (4.13), C^* denotes a set of ground-truth communities, C a set of detected communities, and $\delta(\cdot)$ is a similarity metric.

$$\frac{1}{2|C^*|}\sum_{C_i^* \in C^*} \max_{C_j \in C} \delta(C_i^*, C_j) + \frac{1}{2|C|}\sum_{C_j \in C} \max_{C_i^* \in C^*} \delta(C_i^*, C_j) \tag{4.13}$$

4.3.3 Results

Using the evaluation function defined in Equation (4.13) we find the $F1$ score and Jaccard similarity between the detected communities from all methods and the ground-truth communities.

The results are provided in Tables 4.3 and 4.4 and show SENC outperforms most other methods over all datasets and achieves the highest average performance. Unfortunately, the current implementation of EDCAR was unable to process most of the networks. We

Table 4.4: Jaccard index for all methods and datasets.

Method	Attr.	NSF	DBLP10	DBLP11	DBLP12	Scratch	Avg.
CoDA	No	0.132	0.172	0.168	0.162	0.174	0.162
Link Clust.	No	0.233	0.166	0.166	0.161	**0.265**	0.198
CESNA	Yes	0.139	0.167	0.161	0.156	0.228	0.170
EDCAR	Yes	0.112	N/A	N/A	N/A	N/A	N/A
SENC	Yes	**0.269**	**0.190**	**0.187**	**0.190**	0.235	**0.214**

Table 4.5: Top-5 researchers of the AMPLab and Computational Learning communities with corresponding membership weights.

AMPLab		Comp. Learning	
Peter Bartlett	0.5084	Laurent El Ghaoui	0.5884
Laurent El Ghaoui	0.4116	Peter Bartlett	0.4916
Michael Franklin	0.1346	Jesse Snedeker	0.4647
Michael Jordan	0.1049	Federico Girosi	0.4134
Alexandre Bayen	0.0996	Robert Berwick	0.2830

believe this is partly due to the large number of features.

We note the relative difference in performance of CoDA and CESNA to Link Clustering flips between the networks with higher and lower clustering coefficients. In the NSF and Scratch networks, Link Clustering outperforms CoDA and CESNA but performs worse than CESNA on the DBLP10 network and worse than CoDA on every DBLP network. This may indicate these other methods include a biased definition of communities which is not found in all social networks. SENC performs well across all the networks and avoids this problem through the use of its configurable bounds chosen based on network statistics such as clustering coefficients.

4.3.4 Interpretation of Detected Communities

We also perform a qualitative analysis on communities discovered by SENC to illustrate the interpretability of its results. Several communities relating to data mining and machine learning were found in the NSF CISE network.

We present the top-5 researchers and top-40 terms of two such groups in Table 4.5

Figure 4.2: Word clouds of the top-40 terms from the AMPLab community (left) and computational learning community (right).

and Figure 4.2. The first community is associated with Berkeley's AMPLab[3], which works on problems involving machine learning, cloud computing, and crowdsourcing. The top-5 researchers are all EECS faculty at Berkeley and Michael Franklin and Michael Jordan are both directors of AMPLab. Recall membership weights are normalized per-researcher and a lower membership weight indicates the researcher's work is also captured by other community topics. Most of the terms are self-explanatory, but the term *Alon* refers to Alon Halevy of University of Washington whose name appears in several award abstracts and has collaborated with Michael Franklin.

We find another community with 12 members in common with the AMPLab community. Its topic may be described as computational learning and its applications to computer vision and natural-language processing. The AMPLab and computational learning communities have 41 and 34 members respectively, with roughly about one-third being shared. These common members include: Michael Jordan, Michael Franklin, Peter Bartlett, and Tomaso Poggio. Figure 4.3 shows both the distinct and common members of both communities.

Although both communities are generally concerned with human-centric applications of machine learning, the AMPLab community is focused on computing architecture to solve

[3]https://amplab.cs.berkeley.edu

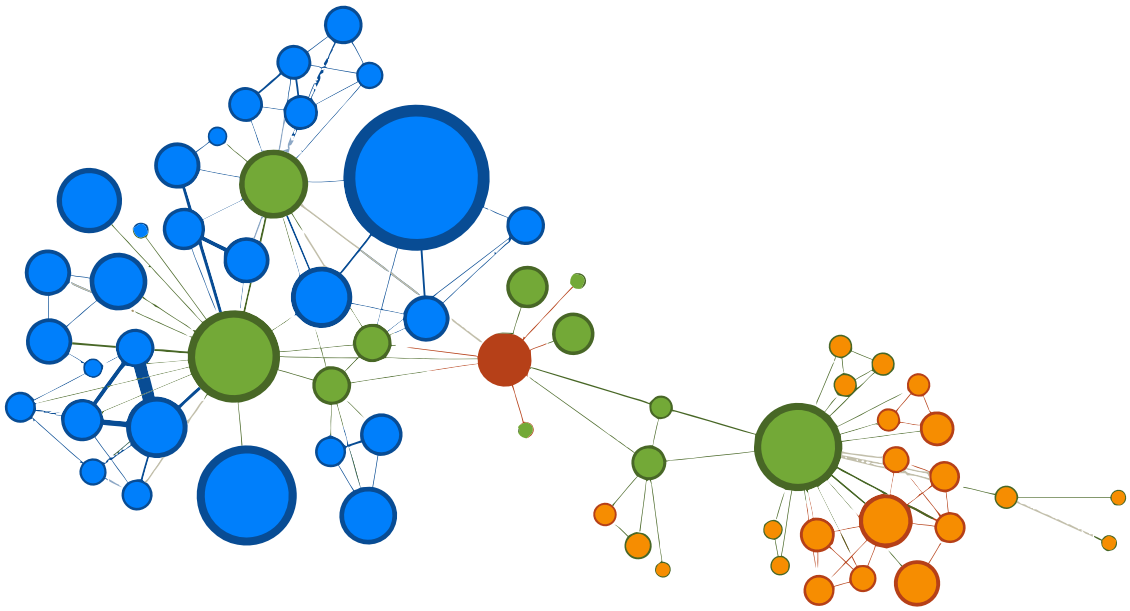


Figure 4.3: The AMPLab (blue) and computational learning (orange) communities with shared members (green). The red node represents Michael Jordan.

such problems, while the computational learning community is focused on understanding human vision and motor control. This discovery of overlapping communities with shared general interests but distinct features exemplifies an advantage of SENC's initialization by seed groups.

Chapter 5: Temporal Artifacts from Edge Accumulation

5.1 Introduction

The modeling of social networks is an expansive and active area of research. While models may incorporate other network features such as node attributes [55, 99, 100, 104], nearly all rely on network structure. Many methods are now also incorporating temporal dynamics [55, 56, 68, 105], but how the temporal information is integrated varies. There are various approaches [9, 55] to representing a dynamic social network as a series of networks, but until recently [30, 106] all have lacked theoretical foundation. The work presented in this chapter has been published in [15, 16].

Dynamic network representations which capture edge deactivation [55, 107] have shown to improve task-specific performance. However, many leading community detection methods [37, 104] are based on *cumulative graphs* and ignore edge deactivation. The findings presented in this chapter suggest that some existing models may be designed to accommodate temporal artifacts introduced by not including edge deactivation in the processing of network data.

There are two social network phenomena which motivate this analysis: *social capacity* [27] and *bursty events* [29]. Social capacity can be viewed as a per-node limit on the number of incident edges active at any given time and thus conflicts with the claim of densification and shrinking diameters in social networks [35, 36] unless additional conditions are met. For example, a network where every new node has a larger social capacity would lead to densification and shrinking diameters. While variation in social capacity based on demographics has been observed [30, 106] there has been no evidence presented that would indicate social capacity is a function of when a node joins the network.

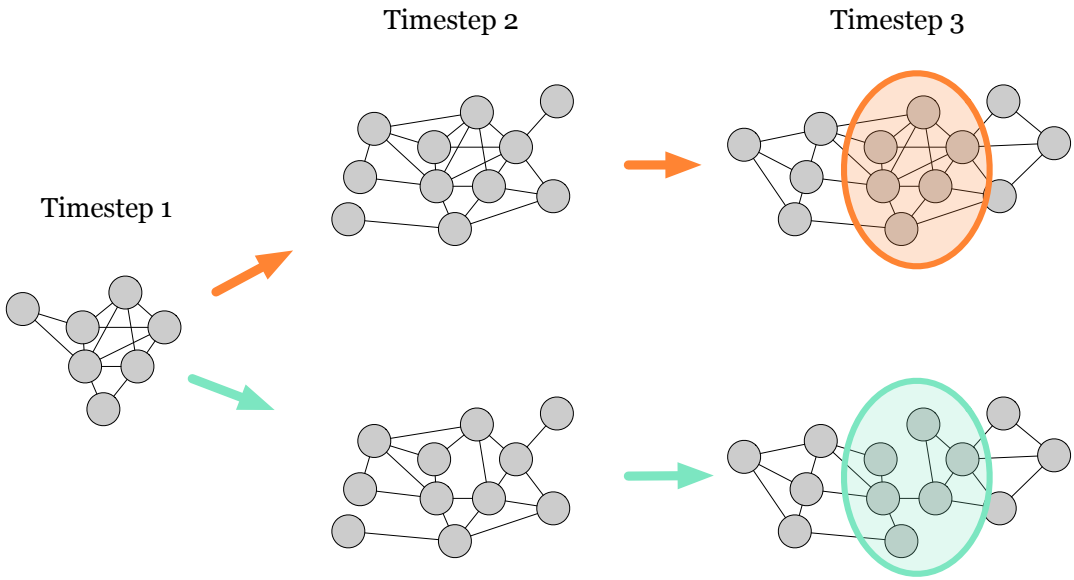


Figure 5.1: An example of divergent network structure between *cumulative graphs* (top, orange) and *activity graphs* (bottom, green).

In order to measure the existence of densification and shrinking diameters, we first must construct a series of network snapshots which more accurately captures network structure than simply accumulating all edges over time. We do this by using communication activity between nodes as evidence that an edge is active. The bursty dynamics of social communication are accounted for by measuring the inter-event times and selecting an observation window large enough to minimize incorrectly deactivating an active edge. Thus we are able to construct a series of *activity graphs* which provide a more accurate approximation of the network state at a given point in time. This method of graph construction has been used previously on a mobile phone network [30] to improve understanding of communication strategies. We can then measure and compare evidence of densification and shrinking diameters in both a cumulative graph series and an activity graph series. Figure 5.1 provides an example of how series of cumulative and activity network snapshots may diverge over time.

Densification and diameter shrinking are accepted as basic characteristics of dynamic

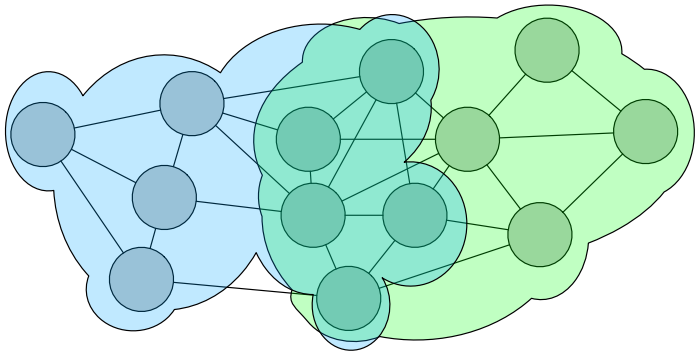


Figure 5.2: Dense community overlap when edges are never deactivated.

social networks. This would support the argument that the most dense regions of a network are not the centers of communities as previously shown [4] but instead the overlapping regions of communities [6] (Figure 5.2). However, this chapter presents results which contradict those findings. When edge deactivation is incorporated, we do not find evidence of densification and diameter shrinking appears to be dependent on the rate of new nodes entering the network; this may be an effect of social capacity.

5.2 Evidence of Temporal Artifacts

5.2.1 Datasets

A dataset with timestamped interactions is required to construct an accurate temporal series of networks. We use data from Scratch [102], an online community where users may write and share programming projects, and Facebook [108].

In Scratch, there are several ways by which users may interact: project comments, project remixes, gallery curation, and user following. More information about Scratch and these interactions may be found in [102]. We selected a single type of interaction to simplify analysis. Project comments are a natural choice as they are the most-frequent interaction between Scratch users and thus a better approximation of edge status (active/inactive). These project comments serve as a means for users to communicate within the context of

a project. The comments in the Scratch dataset are timestamped and thus we can create timestamped edges from comment authors to project authors.

The Scratch dataset spans over March 2007 to December 2011 and includes a large period of rapid growth in Scratch users, shown in Figure 5.3, which does not slow until towards the end of the dataset. There are a total of 7,788,000 project comment interactions between 164,205 users. There are many short-term interactions and we filter out directed interactions between pairs which only occur once or twice when measuring communication behavior. Such interactions have undefined or trivial inter-event statistics as there are zero or one inter-event observations when only one or two interactions are observed. There are a total of 1,799,050 of such interactions which were removed, leaving 5,988,950 interactions. The Scratch dataset used to construct the networks may be obtained from the MIT Media Lab website[1].

Facebook allows users to interact by posting on each other's wall and these posts are typically comments, photos, and web links. Each of these posts is recorded as an interaction with a source user (the post author), a destination user (the owner of the wall), and a timestamp.

The Facebook data includes wall post interactions between a subset of Facebook users over October 2004 to January 2008. There are a total number of 876,993 wall posts between 46,952 users. The Facebook networks were prepared similarly to the Scratch networks, edges are only formed between node pairs that have at least a total of three interactions over the entire dataset. The Facebook dataset used to construct the networks may be obtained from the KONECT website[2].

5.2.2 Methodology

As the relationships in the interaction networks are based on communication events between nodes, we check for evidence of bursty patterns. Bursty communication can be identified by the dispersion of inter-event times between node pairs. If communication is bursty then the

[1] https://llk.media.mit.edu/scratch-data

[2] http://konect.uni-koblenz.de/networks/facebook-wosn-wall

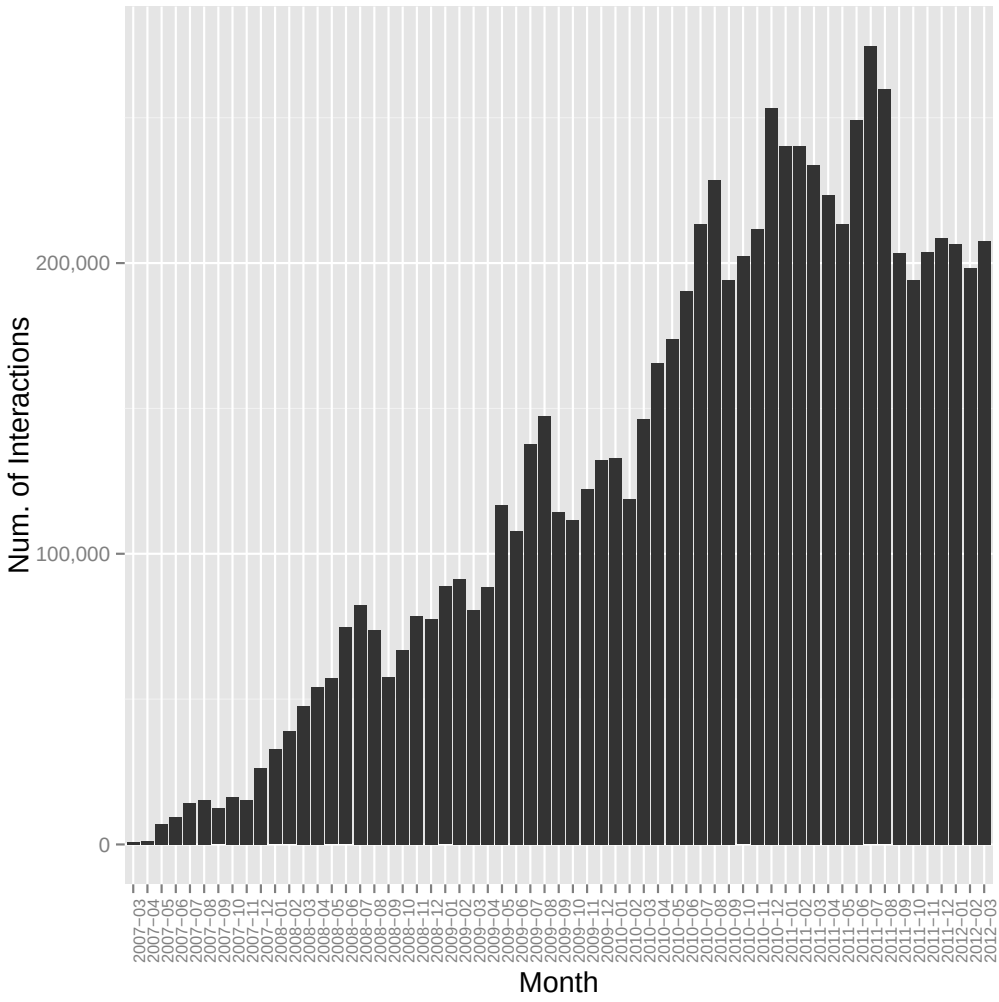


Figure 5.3: The number of interaction events occurring by month in the Scratch dataset.

standard deviation of inter-event time will be larger than the mean. The ratio of the mean and standard deviation of inter-event times is the coefficient of variation (cv) and used to measure dispersion. When $cv > 1$, there is evidence of bursty communication. The use of dispersion to identify burstiness is further discussed by Miritello et al.[32].

We hypothesize the observation of densification and diameter shrinking [35, 36] may be attributed to the inclusion of deactivated edges in a network. To test this we construct two graph series from each dataset. Each network in all the series captures network activity over consecutive and non-overlapping periods. The size of the observation windows are based on the inter-event times of the datasets. A three-month observation window was selected for the Scratch series because it is large enough to account for the majority of inter-event times (97% of inter-event times are < 62 days) and conveniently maps to annual quarters. The first series is a *cumulative graph series* where new nodes and edges are added at each consecutive snapshot to the previous network in the series. The second series is based on node interaction activity and we refer to it as the *activity graph series*. The two Facebook series are based on a six-month observation window that was calculated similarly.

Edge activity is determined by tracking the activation and deactivation of edges between consecutive observation windows. A similar approach has been used in previous literature [32, 106]. An edge is considered to activate if it is not present in the preceding observation window but an interaction event occurs in the current observation window. Similarly, an edge is deactivated if an event occurs in the current observation window but not in succeeding period. Only edges active in each observation window are used to construct the activity graph series of both datasets.

The edge-node ratio ($\frac{num.of edges}{num.of nodes}$) is calculated for each graph in all series and used to measure densification. If densification is present, we expect the number of edges to grow super-linearly in the number of nodes [36]. We also measure the effective diameter of every graph in both series to determine whether diameter shrinking is observed. The effective diameter is the smoothed count of the smallest number of hops at which at least 90% of all connected pairs of nodes can be reached [36]. We prefer the effective diameter over the

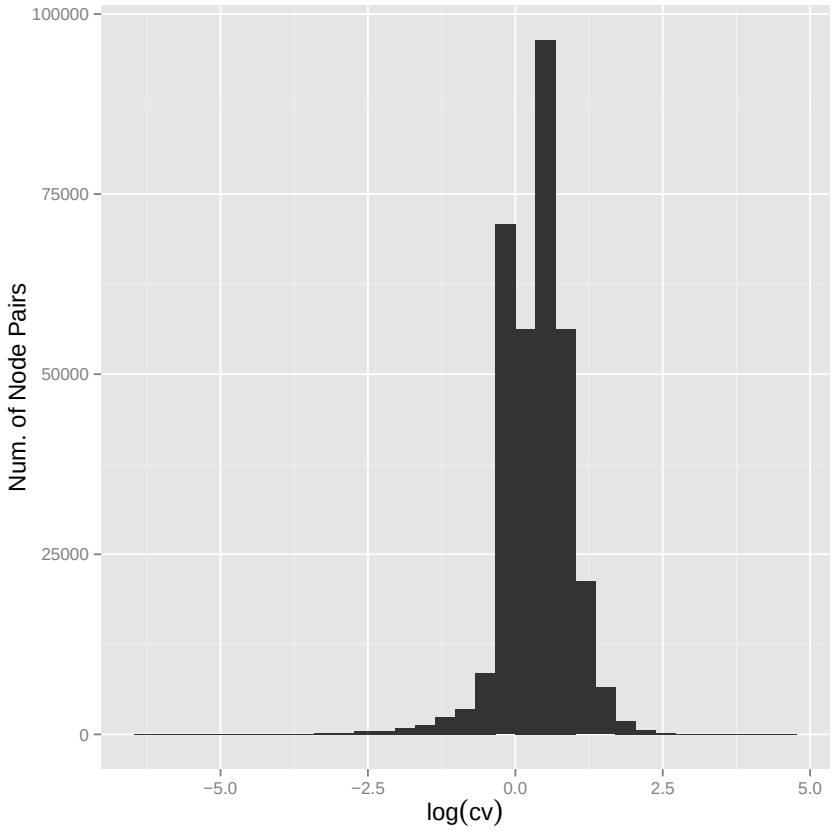


Figure 5.4: The $log(cv)$ for node pairs in the Scratch interaction network with at least three events. A small number of node pairs (1,038) were removed for this plot as they had a cv of zero and thus were undefined.

standard diameter because it was used in [36] to report diameter shrinking and is more robust to degenerative graph structures.

5.2.3 Results

As shown in Figure 5.4, bursty communication patterns are observed in the Scratch dataset as the cv values are frequently greater than 1 ($log(cv) > 0$). Bursty communication patterns are also observed in the Facebook dataset, though somewhat less frequently with just under half of interacting node pairs having cv values > 1. The reduced frequency of bursty communication patterns in Facebook may be due to node pairs having either a single burst

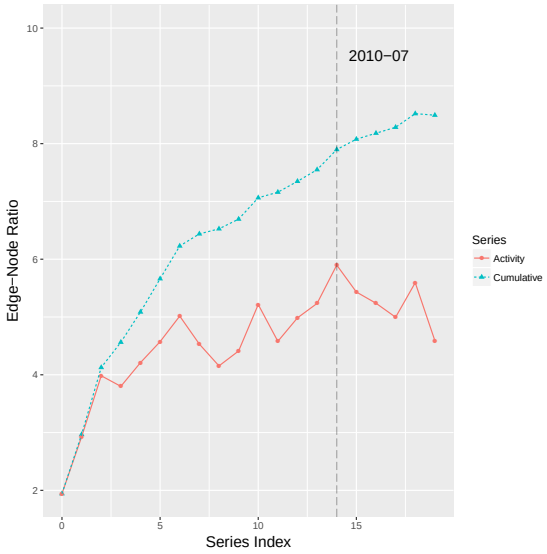


Figure 5.5: The edge-node ratio over time in the Scratch cumulative and activity graph series.

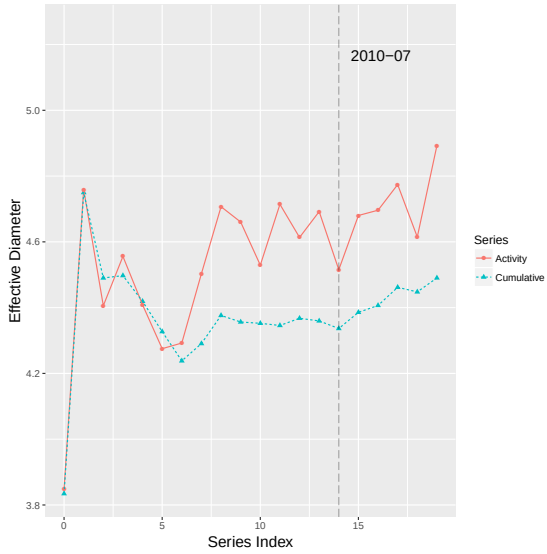


Figure 5.6: The effective diameter over time in the Scratch cumulative and activity graph series.

of interactions or interactions around a regular event (e.g., posting on a friend's Facebook wall for their birthday).

For Scratch, we see evidence of densification in the cumulative series but not in the activity series—shown in Figure 5.5. The accumulation of edges, without removal of deactivated edges, appears to introduce densification as a temporal artifact in the Scratch interaction network. This is especially clear when the number of interactions stops growing around July 2010, denoted by dashed vertical line in both Figures 5.5 and 5.6.

An overall trend of diameter shrinking is not clearly observed in either Scratch network series. After an initial decrease, Figure 5.6 shows a generally increasing diameter for both series and a larger variance in diameter for the activity series. The lack of diameter shrinking may be due to the growth of the Scratch website during the period of time covered in the dataset. However, the effective diameter of the cumulative series is generally smaller than that of the activity series. This is unsurprising given that the cumulative snapshots contain additional edges which would reduce the distances between pairs of nodes.

In the Facebook network series we see trends similar to that of the Scratch network

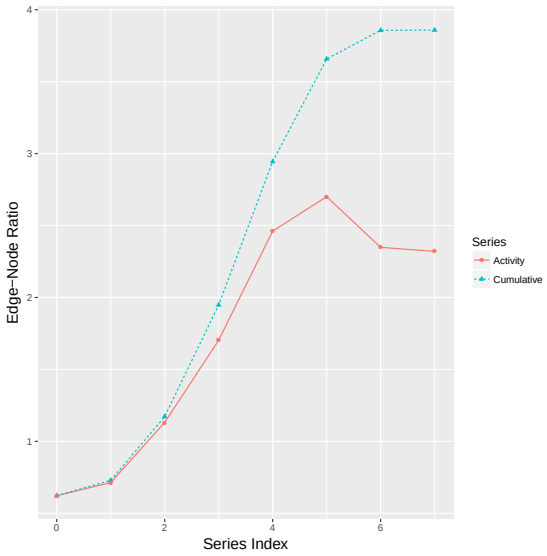


Figure 5.7: The edge-node ratio over time in the Facebook cumulative and activity graph series.

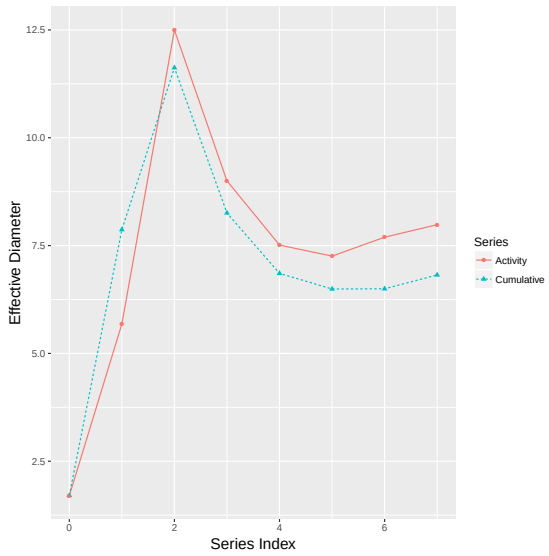


Figure 5.8: The effective diameter over time in the Facebook cumulative and activity graph series.

series. As in Scratch, Figure 5.7 shows that the edge density continues to increase in the cumulative series but not in the activity series. While an initial decrease in the diameter is observed in both Facebook network series, Figure 5.8 depicts the diameter of both series slowly increasing in the later network snapshots.

These findings are not unexpected but they are contrary to previous literature [35, 36] which has served as the basis for leading network models. The edge-node ratio in the cumulative graphs is monotonically increasing over time and social capacity is ignored. In contrast, the edge-node ratio in activity graphs may decrease or stabilize as inactive edges are detected and removed.

Figures 5.9 and 5.10 reveal another interesting artifact caused by accumulating inactive edges. In Figure 5.9, we see that the number of nodes in the cumulative series is increasing at a much larger rate than the activity series. The total number of nodes, along with edge density, are exaggerated by not removing inactive edges. While this is the case for the Scratch dataset, the snapshots in both of the Facebook network series have a similar number of nodes.

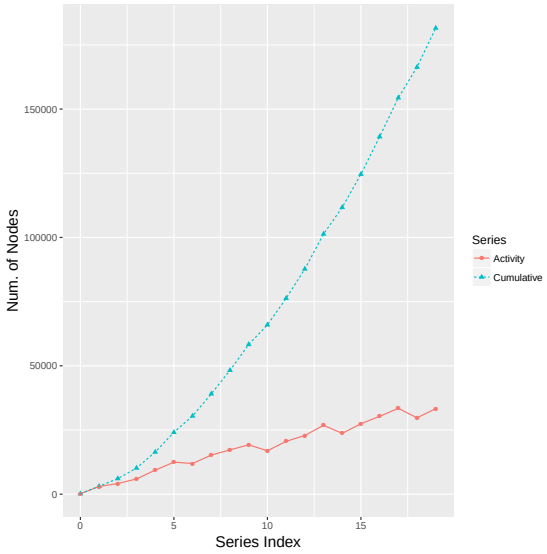


Figure 5.9: The node counts over time in the Scratch cumulative and activity graph series.

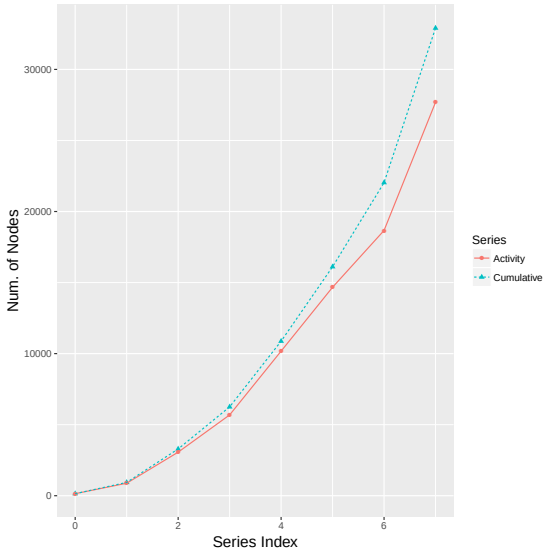


Figure 5.10: The node counts over time in the Facebook cumulative and activity graph series.

This contrast between the Scratch and Facebook datasets indicates many Scratch users do not remain active while most users in the Facebook dataset do stay active. The short membership of some nodes in the Scratch network is likely due to Scratch being used as a teaching aid in classrooms. Students will create an account to participate in class and then become inactive at the end of a school semester or year. Ignoring edge deactivation may mask the actual node interaction patterns and contribute to mistaken findings which appear reasonable, as demonstrated by the difference in node lifetimes of the Scratch and Facebook datasets. These findings suggest more accurate social networks may be derived from ongoing dyadic interactions rather than one-time events such as "following" or "friending."

Chapter 6: Persistent Roles

6.1 Introduction

Online networks rely extensively on user contributions and participation for their vibrancy. This requires that users perform certain activities and take on specific roles within the network. In this chapter we take a distinct approach to identify latent role behaviors which persist over time by examining interaction patterns and structural positions of users. This approach provides a novel way of understanding latent mechanisms that underlie the structure and processes of dynamic networks. The work presented in this chapter has been published in [17].

Role discovery has been applied to many networks [9, 109] and incorporated into static network models [49]. Despite the prevalence of role discovery methods and applications, no experiments have been presented that show the existence of persistent roles derived directly from data. While network-specific roles are useful for many purposes, identifying a set of roles which commonly occur in online social networks enables new methods for comparative analysis which emphasize relationships between roles.

In this chapter, I present a methodology for discovering and tracing persistent roles over time. We discover roles for 26 network snapshots of online social networks from two datasets (Facebook and Scratch). These roles are found to persist both within and between the network snapshots from both datasets. We then conduct a summary analysis to demonstrate how roles may help interpret network structure by considering role membership, transitions between roles, and interaction preferences.

In these experiments, we discover six roles from the networks and show these roles are both distinct from one another and occur in every network from both datasets. These roles are: *popular*, *friendly*, *explorer*, *reciprocated*, *community member* and *active-community*

member. While the discovered roles are common to both datasets and persist over time, we find the relationships between roles may differ. These findings suggest common roles shared among social interaction networks are useful for modeling and comparing networks.

6.2 Discovering Persistent Roles

We aim to find roles which best characterize the nodes in a network. The network datasets we consider in this chapter are dynamic networks which include timestamped, directed *interactions* between node pairs. Each interaction represents a single action such as one user messaging another. As the primary goal is to identify persistent roles over time, we will partition the dynamic network $\mathcal{D} = (\mathcal{N}, \mathcal{E})$ into snapshots, $\mathcal{S}_t$ for each timestep t. The original edges $\mathcal{E}$ are timestamped, directed interactions between node pairs and only edges occurring at timestep t, $\mathcal{E}_t$, are included in snapshot $\mathcal{S}_t = (\mathcal{N}_t, \mathcal{E}_t)$. The edges in $\mathcal{E}_t$ are converted from individual interactions to directed, weighted edges, where the edge weight is the total number of directed interactions occurring between the nodes in $\mathcal{S}_t$. Nodes $\mathcal{N}$ are derived from the edges $\mathcal{E}$ and all nodes present at timestep t, $\mathcal{N}_t$, participate in at least one edge in $\mathcal{E}_t$.

6.2.1 Temporal Network Snapshots

The snapshots we construct are non-overlapping and each snapshot $\mathcal{S}_t$ spans the same length of time, known as the observation window Ω. The structure of network snapshots are defined by the activity which occurred within the observation window, thus there is no accumulation of inactive edges. The observation window Ω is calculated so that most time deltas δt_{ij} between interactions of any two nodes i and j are smaller than Ω. Specifically, we find the average time deltas $\langle \delta t_{ij} \rangle$ for each interacting node pair. The 90^{th} percentile of all average time deltas is then used as Ω. We assume most connected pairs do not continually disconnect and reconnect and thus choosing an Ω which preserves most edges is appropriate. This methodology is described with more detail in [16, 31].

Table 6.1: Node features

	Name	Description
1	*In-degree*	Count of incoming edges
2	*Out-degree*	Count of outgoing edges
3	*Weighted in-degree*	Count of incoming interactions
4	*Weighted out-degree*	Count of outgoing interactions
5	*Reciprocity*	Ratio of reciprocated edges over all outgoing edges
6	*New activity count*	Count of new outgoing edges
7	*Social strategy*	Ratio of new outgoing edges over all outgoing edges [31]
8	*Betweenness centrality*	Number of all shortest paths which pass through the node
9	*PageRank*	PageRank measure of centrality [26]
10	*Weighted PageRank*	Weighted variant of PageRank
11	*Transitivity*	Probability any two neighbor nodes are connected (local clustering coefficient) [110]
12	*Weighted transitivity*	Weighted variant of transitivity [111]

6.2.2 Role Feature Selection

From the network snapshots we find D structural and behavioral features ($D = 12$ for these experiments) for all $n \in \mathcal{N}_t$ nodes and construct a matrix of node attributes $\mathbf{X}_t \in \mathbb{R}^{D \times N_t}$. The complete list of features used is shown in Table 6.1. Most of the features listed in Table 6.1 have common definitions, a few do not. The *new activity count* is computed for each node as the difference of the set of nodes reached from outgoing edges at the current snapshot $\mathcal{S}_t$ and the set of nodes reached from outgoing edges at the previous snapshot $\mathcal{S}_{t-1}$. Similarly, *social strategy* is a ratio of the count of new outgoing edges (outgoing edges at snapshot $\mathcal{S}_t$) that did not exist at the previous snapshot over the total number of outgoing edges for the given node at snapshot $\mathcal{S}_t$, $\frac{\text{num. of new outgoing edges}}{\text{num. of all outgoing edges}}$. Users with a higher social strategy value tend to prefer making new connections (*social explorer*, or simply *explorer*) rather than preserve older connections (*social keeper*) [31].

These features were selected to enable the representation of the unique structural and behavioral patterns which may exist in online social networks which include individual,

timestamped interactions. For example, while in-degree (count of incoming edges) captures popularity, the weighted in-degree (count of incoming interactions, e.g., in Facebook, number of incoming wall comments) captures the overall level of incoming activity for the target node. Features such as transitivity encode information about a node's neighborhood while betweenness centrality and PageRank capture global information about the node's position in the network. The reciprocity, new activity count, and social strategy pertain to interaction behaviors.

6.2.3 Role Discovery and Membership

To find roles, a decomposition of a node-attribute matrix is performed and the resulting basis vectors are the discovered roles. We use non-negative matrix factorization (NMF) [112] for this task. The role vectors contain values corresponding to each feature which can be used to characterize the role — features with higher values are more characteristic of the role. For example, a role with a large in-degree might be labeled as popular.

NMF decomposes a matrix $\mathbf{X} \in \mathbb{R}^{D \times N}$ into a basis matrix $\mathbf{U} \in \mathbb{R}^{D \times L}$ and a coefficient matrix $\mathbf{V} \in \mathbb{R}^{L \times N}$, where L is the factorization rank of the decomposition $\mathbf{X} \approx \mathbf{UV}$. Each of the L columns of the basis matrix $\mathbf{U}$ are the basis vectors or factors (roles) and the N columns of the coefficient matrix $\mathbf{V}$ are the coefficient (weight) vectors which explain how each observation $\mathbf{x}_i$ is represented as a mixture of roles.

NMF is independently run on the matrix of node attributes for each snapshot $\mathbf{X}_t$ with the same parameters. We use the standard Euclidean update equation and Frobenius cost function. We use non-negative double singular value decomposition (NNDSVD) [113] to initialize NMF. This helps NMF converge faster and introduces a bias for sparse factors (roles). We do not expect roles will have non-zero values for all features as we assume roles are a parts-based representation [114] of node attributes. Each role is characterized by a subset of all available features.

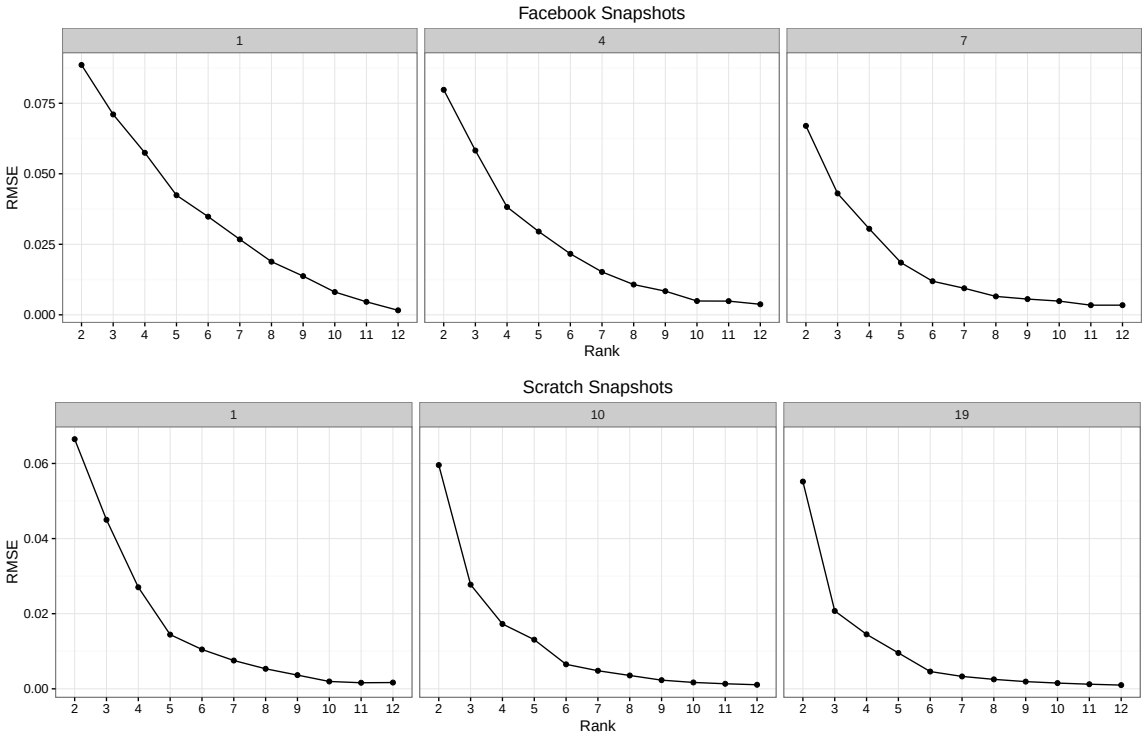


Figure 6.1: Error curves for the first, mid, and final network snapshots in Facebook (top) and Scratch (bottom).

6.2.4 Model Selection

A critical parameter of NMF is the factorization rank L. The common methods for selecting the rank value include: MDL [115], AIC [116], and error curves [117]. We initially tried to use MDL but found model size dominated the description length and resulted in the selection of low-performing models.

Recent existing work on role discovery with NMF [8, 9] used MDL and we attempted to use the same MDL function definition. Unfortunately, it appears the function does not appropriately balance between the model size and error for these datasets. We found that in all cases, the model with the lowest MDL had the smallest rank possible (for NMF with NNDSVD), $L = 2$.

We inspected the error curves, shown in Figure 6.1, and found that $L = 2$ results in a relatively large error. These curves were computed by calculating the root-mean-square

error (RMSE) between the actual data $\mathbf{X}$ and corresponding NMF approximation $\mathbf{UV}$. Instead of MDL, we elected to use the knee of the error curve to estimate the rank. As shown in Figure 6.1, networks across both datasets had a similar error curve. Ranks $L = 5$ and $L = 6$ correspond to the knee point for most of the curves, and therefore are appropriate choices. Rank $L = 6$ is used for the factorization of all networks in these experiments.

6.2.5 Tracking Roles

Given T snapshots and node-attribute matrices for each snapshot $\mathbf{X}_t$, $t = 0 \dots T-1$, NMF is used to perform the approximate decomposition $\mathbf{X}_t \approx \mathbf{U}_t\mathbf{V}_t$. Recall the basis matrix $\mathbf{U}_t$ corresponds to role features and the coefficient matrix $\mathbf{V}_t$ corresponds to role membership weights for each user. We hypothesize that roles may persist over time and need to verify whether the same roles do occur in consecutive basis matrices; i.e., do roles from $\mathbf{U}_t$ appear in $\mathbf{U}_{t+1}$.

This role tracking is performed by measuring the similarity of every pair of role vectors between consecutive snapshots $\{\mathbf{u}_t^i \times \mathbf{u}_{t+1}^j \mid i, j \in 1 \dots L\}$. We use cosine similarity to evaluate the pairs and ensure that each role in snapshot t maps to only one role in snapshot $t+1$ (the mapping is injective). We use a threshold value (0.75) to determine whether a pair matches. That is, if $\text{sim}(\mathbf{u}_t^i, \mathbf{u}_{t+1}^j) > 0.75$ then the pair of role vectors match. In practice, we find most matching pairs in this data have a cosine similarity greater than 0.9.

6.3 Experimental Analysis

6.3.1 Datasets

We use two datasets of timestamped, directed interactions to construct dynamic networks and 26 network snapshots. The first dataset is a collection of Facebook wall posts [108] available from KONECT[1]. In Facebook, users may post on each other's wall and these posts are typically comments, photos, and web links. Each of these posts is recorded as an

[1] http://konect.uni-koblenz.de/networks/facebook-wosn-wall

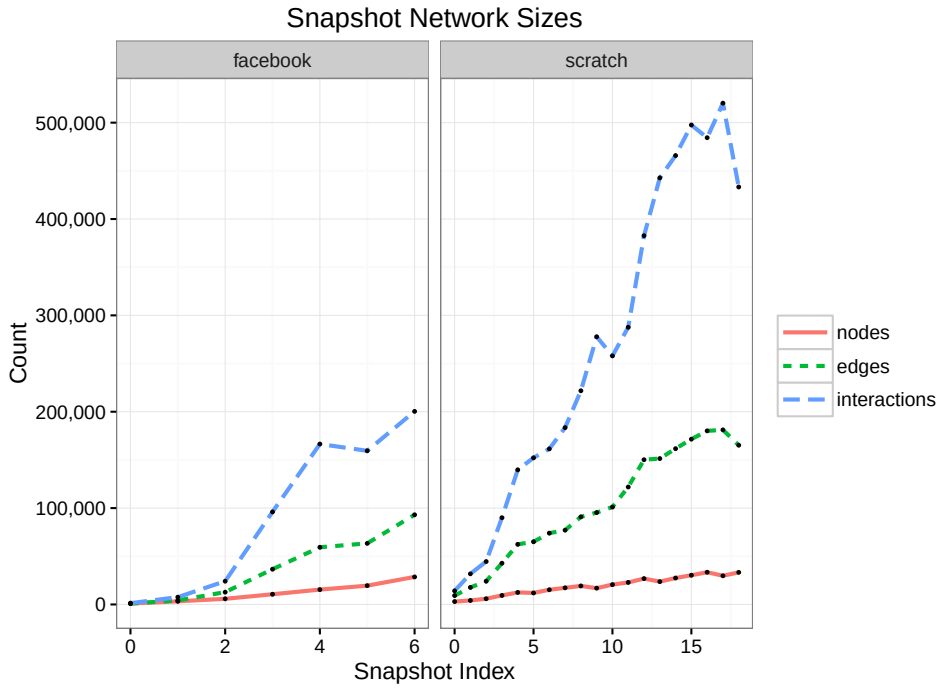


Figure 6.2: Number of nodes, edges, and interactions over time in the Facebook and Scratch networks.

interaction with a source user (the post author), a destination user (the owner of the wall), and a timestamp.

The second dataset is a collection of Scratch project comments [102] extracted from a general Scratch dataset available from the MIT Media Lab website[2]. Scratch is an online social network and web application for writing and sharing software programs. Programming education is the primary objective of Scratch and many users are children and young adults. Scratch users write and share projects; comments may be made on each other's projects. Similar to Facebook walls, project comments in Scratch serve the purpose of public communication between users.

In both datasets, the interactions are used to construct a dynamic network and then network snapshots. The snapshots are constructed using the methodology discussed in Section 6.2.1. Figures 6.2, 6.3, and 6.4 show how the size and clustering of the snapshots

[2]https://llk.media.mit.edu/scratch-data

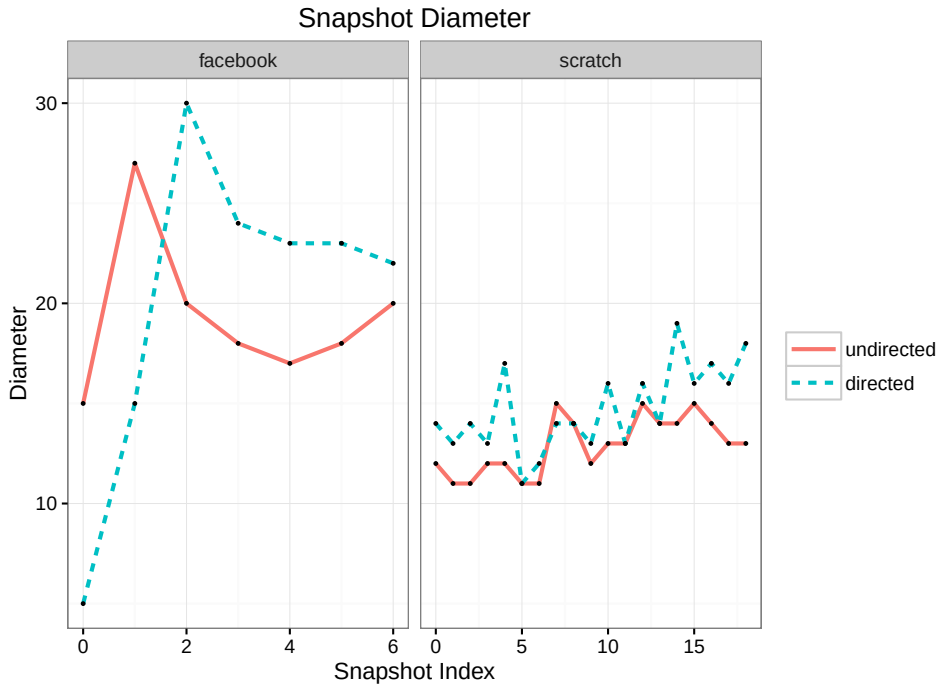


Figure 6.3: Network diameter over time in the Facebook and Scratch networks.

from both datasets vary over time. Note that both the Facebook and Scratch interaction networks are growing over time.

A node-attribute matrix is created for each network snapshot using the features described in Section 6.2.2. Attributes are normalized by min-max normalization with all values belonging to the interval $[0, 1]$.

6.3.2 Methodology

We use the roles found by decomposing the per-snapshot, node-attribute matrix $\mathbf{X}_t$ to answer the proposed research questions. First we demonstrate that a common set of six persistent roles are found in the series of network snapshots from both datasets. While the feature proportions of the roles is similar across datasets and over snapshots, the magnitudes of the vectors change. Correspondingly, the magnitudes of the coefficient vectors (role membership weights) differ between snapshots.

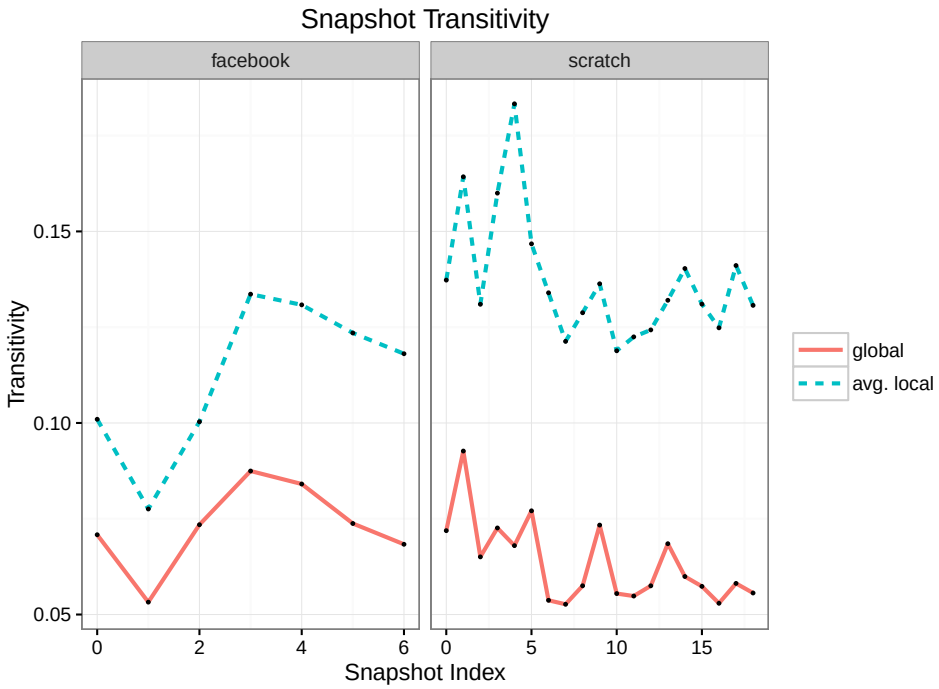


Figure 6.4: Global and average local transitivity (clustering coefficient) over time in the Facebook and Scratch networks.

We resolve this issue by averaging the basis vectors (roles) across all snapshots and then using non-negative least squares (NNLS) [118] to find the optimal coefficient matrix for the data, given the averaged basis matrix. This normalizes the role memberships between snapshots and these membership values are used in the rest of the analysis. Note that since the original basis vectors for all network snapshots had high cosine similarity, the averaged basis vectors also have a high cosine similarity with every original basis vector.

6.3.3 Persistent Roles

We use the methodology discussed in Section 6.2.3 to find roles in each network snapshot from both datasets. Then we follow the methodology described in Section 6.2.5 to determine whether the discovered roles occur in all snapshots from each dataset. We find six roles in both datasets which persist over time and perform a pairwise comparison of the sets of roles from each dataset. There is a one-to-one correspondence (bijection) of the two sets of six roles, using the same cosine similarity test as was used for testing the persistence of roles across consecutive snapshots. That is, the same set of six roles persist over time in both datasets. We note that several roles are dominated by a single feature which is not shared with any other role, this suggests a parts-based factorization of node attributes.

Figure 6.5 shows the discovered roles and their feature weights. The role names were selected according to the distinguishing features of the roles and we describe them here. The *popular* role is defined by the in-degree and centrality features while the *friendly* role has larger proportions in out-degree, weighted out-degree, and the number of new outgoing edges.

The *reciprocated* role is dominated by the reciprocity feature and captures the proportion of a node's outgoing edges which are reciprocated by the receiver node. A node with perfect reciprocity would have a high membership weight in this role. The *explorer* role is dominated by the social strategy feature which indicates whether a node prefers to interact with new nodes rather than maintain existing relationships. We have observed that many nodes start as explorers when they first join the network.

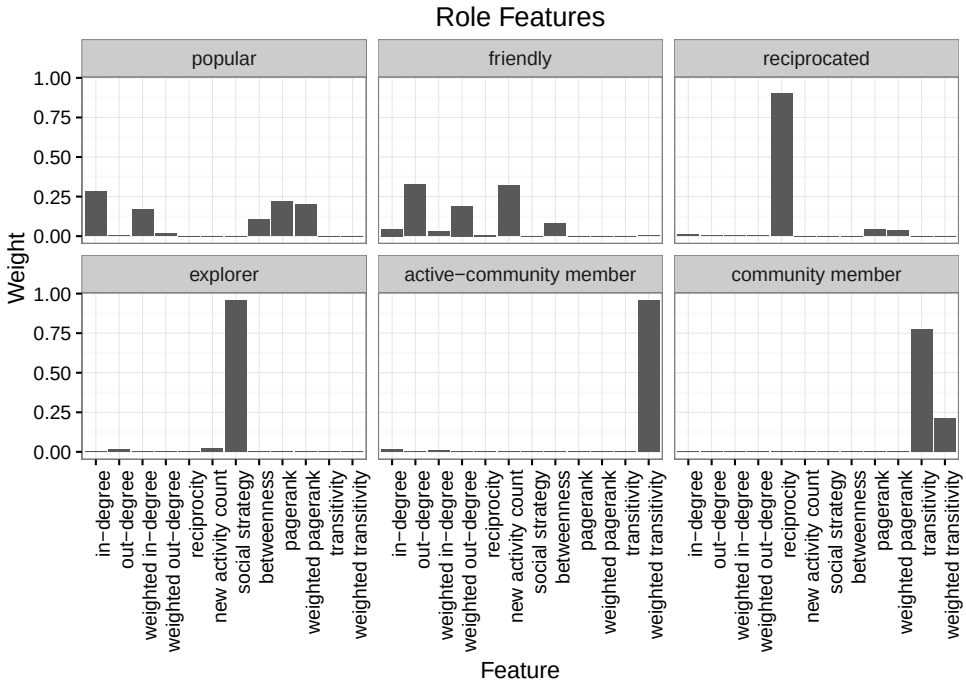


Figure 6.5: Features for all roles, computed as average of role basis vectors from all network snapshots.

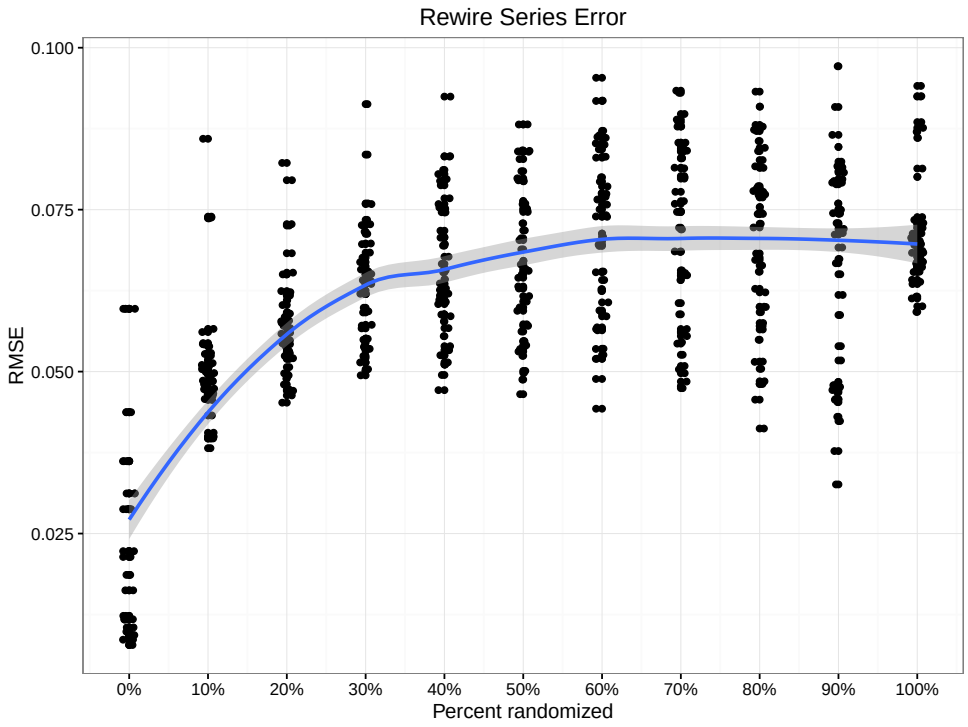


Figure 6.6: Errors plotted for 50 series of randomly rewired networks.

The final two roles, *active-community member* and *community member*, capture the clustering of nodes. Active-community member is dominated by weighted transitivity which is similar to standard transitivity (local clustering coefficient) but accounts for the strength of the edge when calculating the coefficient. As we defined edge weight as the number of directed interactions between a pair of source and destination nodes, a node with a high weighted transitivity coefficient is involved in an active community. In contrast, a node with a high unweighted transitivity coefficient simply participates in a densely-connected community and we cannot say anything about the activity of the community without further information.

6.3.4 Evidence of Role Dependence on Network Structure

We conduct an experiment with synthetic data to demonstrate the discovered roles capture patterns particular to the datasets. Fifty series of rewired networks were generated from

networks in the original datasets. For each series, one of the snapshot networks was randomly selected. An increasing percentage of interactions in the network were removed and replaced with the same number of random interactions. Non-negative least squares (NNLS) is used to find the optimal role memberships (coefficient matrix) for each of the rewired networks.

The root-mean-square error between the actual data and the optimal approximation is calculated and Figure 6.6 shows the error increases as more interactions are randomly rewired. Thus this analysis supports the fact that the discovered roles reflect an intrinsic property of both social interaction networks, and not an artifact of the methodology used.

6.3.5 Role Membership

Using the persistent roles, we compare their distributions of role membership weights and check for correlations between roles. The role membership correlations (Spearman's coefficients) were calculated for every snapshot network, however due to space constraints only the results for the final snapshot from Scratch is shown in Figure 6.7.

The role membership correlations tend to be similar between all network snapshots in each dataset with one notable exception. Several correlations in early Facebook snapshots (popular and friendly, community member and friendly) shifted from having a negative correlation to a positive correlation. This change in Facebook may be due to the growth and sudden increase of activity after the first few snapshots.

6.3.6 Role Transitions

Nodes may be members of multiple roles and their role memberships may change over time. We visualize these transitions in Figure 6.9 for both the Facebook and Scratch datasets by identifying the top-5% nodes of each role for each network snapshot and draw a line between the roles of subsequent snapshots if nodes transition from one role to the other between those two snapshots. We select the nodes with the highest role membership weights as we expect them to be exemplary representatives of the roles. The height of the bars corresponds to

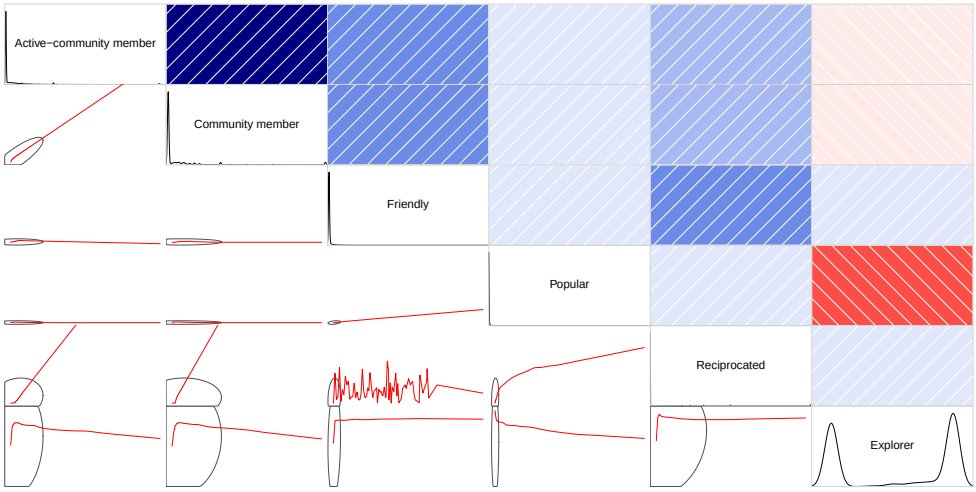


Figure 6.7: Role correlations for the final snapshot from the Scratch dataset. The upper panels are colored to correspond to positive (blue) and negative (red) correlation. Darker shaded panels indicate larger correlation. The diagonal panels show the distribution of role membership weights. The lower panels show a confidence ellipse and smoothed line of the correlation.

the number of nodes with the role. A line is drawn between two roles if at least 10 users transitioned between the roles. The transition lines are sized according to the logarithm of the number of transitioning users. Since a user may share multiple roles, some transition lines merge and show users with multiple roles in common transitioning to a role in the next timestep. Figure 6.8 helps explain how to interpret the transition lines.

As shown in [9], role membership of nodes may change over time and understanding these transitions allows us to construct predictive models. In this work, since a set of common roles has been identified, we can also perform comparative analysis of role transitions between the two datasets.

In both datasets, we see there are many transitions between *popular* and *friendly* roles as well as both community member roles. This is unsurprising as membership correlation is high for both pairs of roles. Further we note that neither *popular* nor *friendly* nodes ever transition to the *explorer* role. In contrast, users do transition from *explorer* to *popular* and

Figure 6.8: A transition line from the *red* role to the *blue* role (left). A combined transition line from the *red* and *green* roles to the *blue* role (right). A combined line corresponds to transitioning users who belong in the top-5% of multiple roles in a single timestep.

friendly. This suggests that the most-popular users are less inclined to form new connections at the same rate as the top-5% *explorer* users.

There are also differences in the role transitions between the two datasets. In Facebook, we observe some community member nodes transition to the *explorer* role but this does not occur in Scratch. We hypothesize this may be attributed to the different uses of the social networks. While Facebook is a general social network, Scratch is used for teaching programming by schools and it is common for students in those classes to primarily only interact with other classmates.

6.3.7 Role Affinity

In this section we determine whether the persistent roles affect user link preferences. As the networks used in this study are directed, we consider both how roles impact the selection of nodes to interact with (outgoing) as well as how roles affect the attractiveness of some nodes (incoming). All nodes are assigned their primary role (the role with highest membership weight) for the role affinity analysis.

In Figure 6.10, we have colored nodes according to role and highlighted a subgraph for demonstration purposes. A standard force-directed layout algorithm was used to position the nodes. Note that while nodes with a higher in-degree tend to be either popular (magenta) or friendly (black), the friendly nodes have more outgoing interactions (larger outgoing edges). While friendly and popular roles reside in the core of the subgraph, explorer (green) and reciprocated (yellow) nodes appear on the periphery.

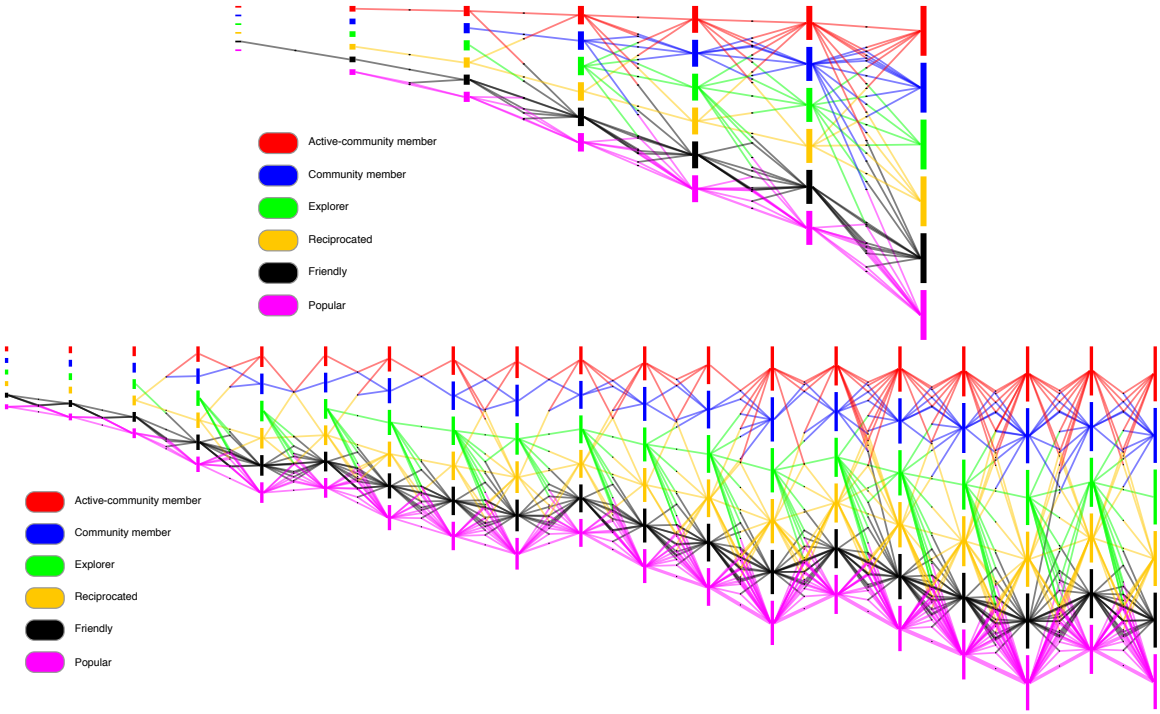


Figure 6.9: The role transitions for the top-5% users in each role over all snapshots for Facebook (top) and Scratch (bottom).

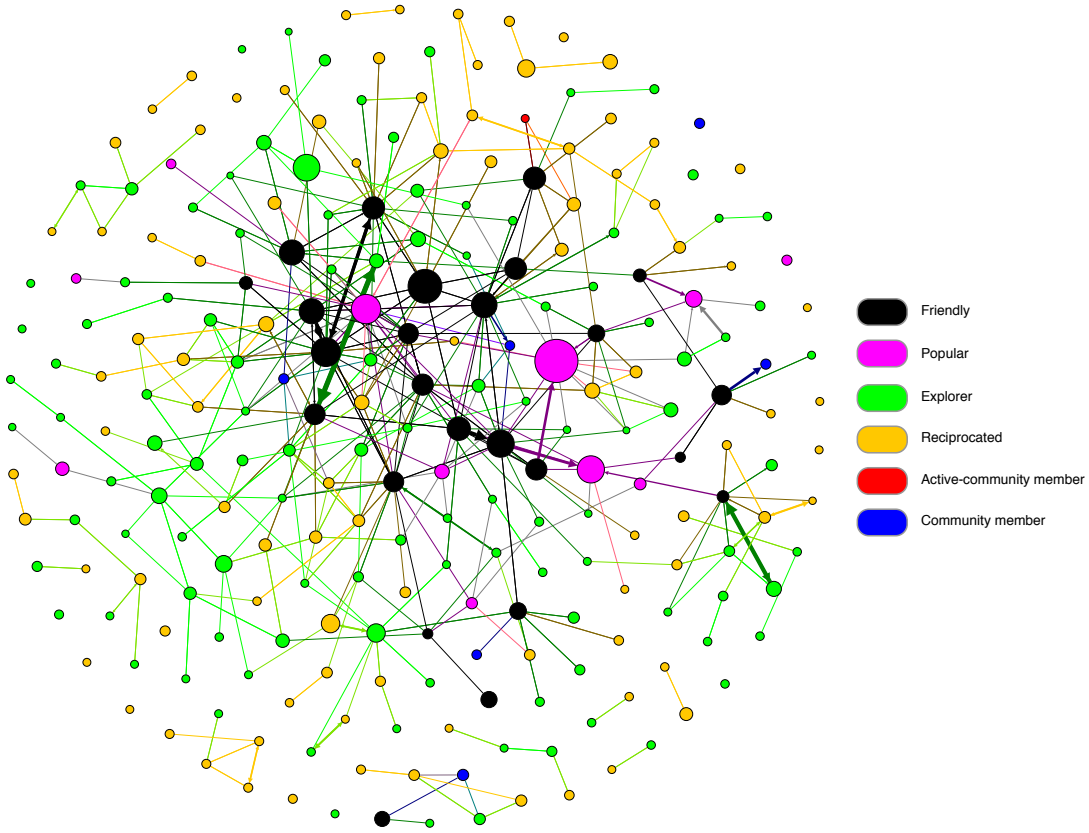


Figure 6.10: A subgraph from a Facebook snapshot network. Nodes are colored by their primary role and sized according to their in-degree. Edges are sized according to the number of interactions they represent.

We augment the network visualization with Figure 6.11 to present the exact counts of edges between roles. We note the lack of incoming edges to explorer nodes; evidence of this is also visible in the network of Figure 6.10.

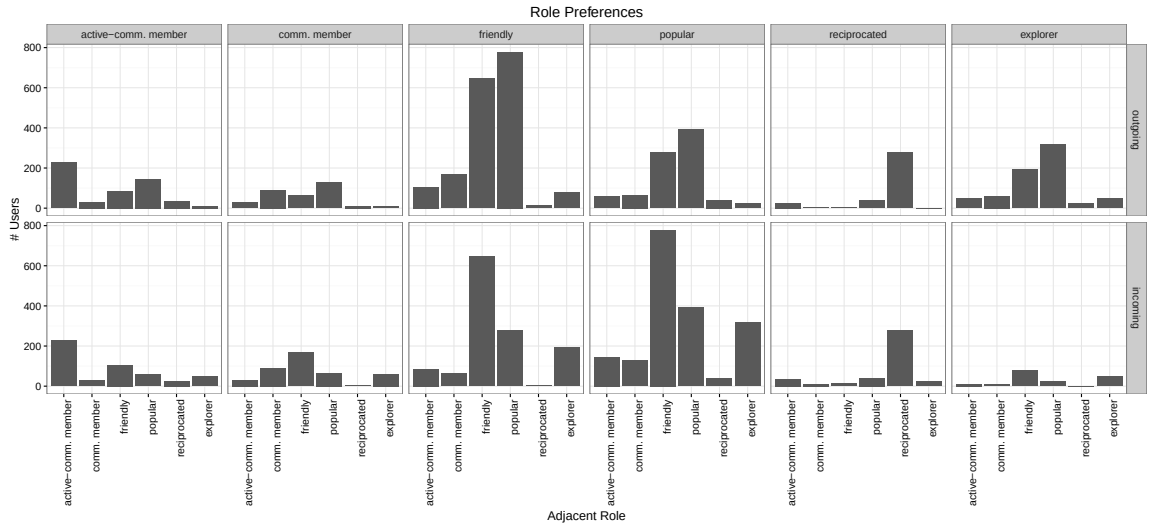


Figure 6.11: The number of users with a primary role linked to/from other user roles. The column labels refer to the source node roles (for outgoing edges) and destination node roles (for incoming edges). The roles on the x-axis refer to the adjacent nodes.

Chapter 7: Group-Node Attention for Community Evolution Prediction

7.1 Introduction

Social networks can change dramatically over time and predicting those changes is important in various real-world setting such as supporting education in massive open online courses (MOOCs) and online communities of creators (OCOCs) [119, 120] or disrupting criminal group persistence [121].

Existing work in community evolution prediction has focused on developing new frameworks that label events occurring between pairs of communities in consecutive network snapshots. A main goal of these frameworks is to define community evolution events in such a way that prediction of those events by standard classification models improves relative to other community evolution prediction frameworks. Rather than attempt to prescribe another event prediction framework, I propose a new prediction model inspired by recent work in graph attention networks that relaxes graph attention by leveraging community membership information.

In this work, I propose a graph neural network model, the Group-Node Attention Network (GNAN), which is capable of attending to community members and neighbors to construct an embedding for groups used to predict the occurrence of one or more community-level events in the next snapshot. In Chapter 6, we have seen evidence of persistent social roles that are uncovered directly from network data. A motivation for group-node attention is to enable the learning of group representations from group-related roles which may be included in the learned node embeddings. To the best of my knowledge, GNAN is the first use of graph neural networks for the community evolution prediction task.

Table 7.1: Definition of notation.

Symbol	Definition
$\{\mathcal{G}^1, \ldots, \mathcal{G}^T\}$	Series of T snapshot networks.
$\mathcal{G}^t = (\mathcal{V}^t, \mathcal{E}^t)$	The set of nodes and edges for network $\mathcal{G}^t$.
$\{\mathcal{C}^1, \ldots, \mathcal{C}^T\}$	Series of community subgraph sets from T snapshot networks.
$\mathcal{C}_i^t$	The subgraph for group i of snapshot t.
$\mathsf{M}(\mathcal{C}_i)$	The set of member nodes for subgraph $\mathcal{C}_i$.
$\mathsf{N}(\mathcal{C}_i)$	The set of nodes adjacent to subgraph $\mathcal{C}_i$.
N	Number of nodes associated with a group.
E	Number of event labels.
D_n, D_g	Number of attributes in a single snapshot per node/vertex and group.
D_m	Number of hidden dimensions used for the model
D_q, D_k, D_v	The query, key, and value sizes used in group-node attention.
H	Number of attention heads.
P	Number of previous snapshots used to construct node attribute vectors.
$\mathbf{x}_u^t$	$D_n P$ node attribute vector for node u at snapshot t that includes attributes from previous snapshots.
$\mathbf{X}_i^t$	$N \times D_n P$ node attribute matrix for group i in snapshot t.
$\mathbf{g}_i^t$	D_g group attribute vector for group i in snapshot t.
$\mathbf{m}_i^t$	N group-relative position vector for all nodes in group i in snapshot t.
$\mathbf{h}_X$	Hidden representation of the node attributes.
$\mathbf{h}_g$	Hidden representation of the group attributes.

I define a neural network architecture which incorporates changes of time, group-relative node positions, and basic network features and demonstrate it is capable of outperforming baseline methods frequently used in the community evolution prediction literature. Two series of network snapshots are constructed from social interaction networks in Facebook and Scratch and we use the clique percolation method (CPM) [4] and the GED [95] framework to find communities and their associated events. Along with [58], these series of network snapshots contain an order of magnitude more communities than most previous work in community evolution prediction.

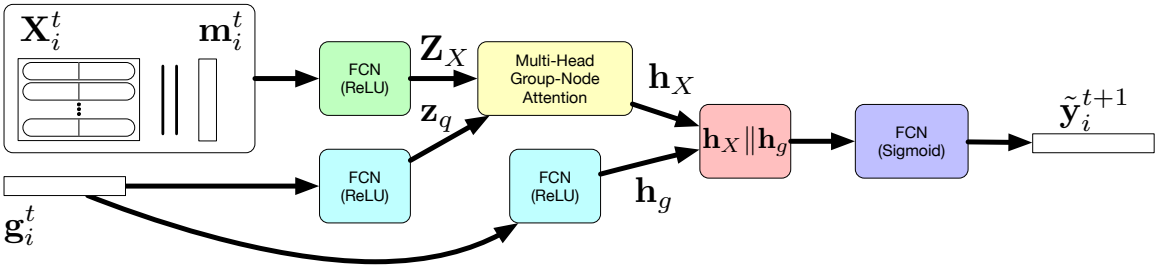


Figure 7.1: Diagram of model architecture showing the computation of event predictions of group i at snapshot t. The $\|$ symbol represents concatenation.

7.2 Group-Node Attention Network

I propose a classification model (Figure 7.1) that learns to use the features of related nodes to find a representation of a group optimized for predicting evolution events in dynamic networks based on *group-node attention*. Group-node attention is performed by using group features to determine the relevance of node features and the contributions of group members and group-adjacent nodes. Node and group features can be derived from the network topology (e.g., network degree and group density) or from external sources such as text documents associated with nodes.

A community can be simultaneously involved in multiple events. Given consecutive network snapshots $\{\mathcal{G}^{t-1}, \mathcal{G}^t\}$ and sets of communities found in each snapshot $\{\mathcal{C}^{t-1}, \mathcal{C}^t\}$, we can train a model that outputs a vector where each element corresponds to an event class. The values of the output vector are used with a decision threshold to predict whether a group is involved with an event of each of the various types.

7.2.1 Spatial and Temporal Mixing

A major intuition behind the proposed model is to incorporate the features of group members and neighbors (spatial information) and changes of those features over time (temporal information). Because we do not require that community identity be known across snapshots, we must rely on node identity for tracking changes over time. Figure 7.2 depicts a

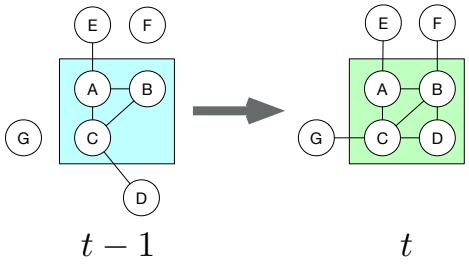


Figure 7.2: The attributes of group members and group-adjacent nodes in snapshot t are used to define the node attributes associated with the group.

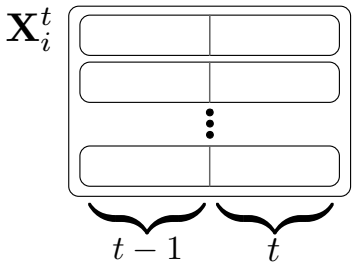


Figure 7.3: Concatenation of node attributes for a node at snapshot t and including historical attributes. In this figure, $P = 2$ with a total of two snapshots used to form the node attribute matrix from the current snapshot t and the previous $t-1$ snapshot.

group and neighboring nodes at snapshot t along with the configuration of those nodes at snapshot $t-1$. Given a node u which exists in both snapshots $\mathcal{G}^{t-1}$ and $\mathcal{G}t$ we can form a vector $\mathbf{x}_u^t$ that is a concatenation of the attributes of node u at snapshots $t-1$ and t, as shown in Figure 7.3.

The input matrix $\mathbf{X}_i^t$ for group i at snapshot t is constructed through row concatenation of all node feature vectors $\mathbf{x}_u^t$ for each node u that is a member or neighbor of group i: $\{u \in \mathsf{M}(\mathcal{C}_i^t) \cup \mathsf{N}(\mathcal{C}_i^t)\}$. An additional dimension, represented as vector $\mathbf{m}_i^t$ in Figure 7.1, is used to specify whether the node is a member or group-adjacent (one-hop neighbor of the group).

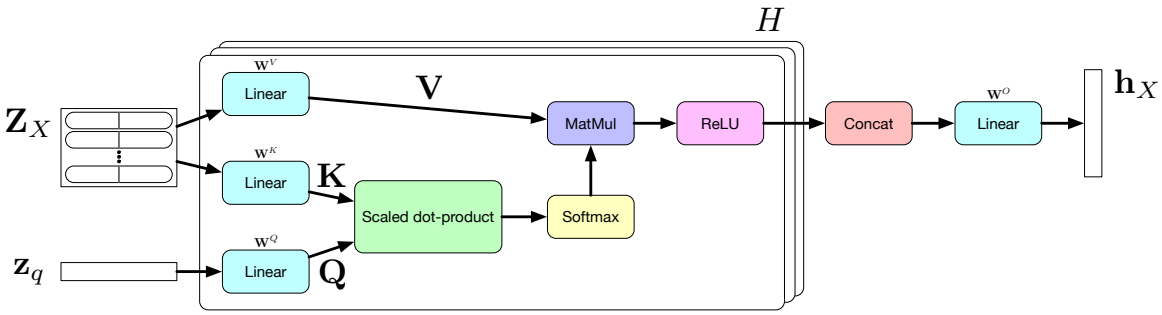


Figure 7.4: A diagram of the multi-head group-node attention layer used in the model with H attention heads.

7.2.2 Group-Node Attention

In order to generate the aggregated hidden group vector $\mathbf{h}_X$, we use the result of the spatial and temporal mixing of the nodes associated with the input group along with transformed group features to construct query, key, and value matrices $(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ using learned weight matrices $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V$ as in [76]. Figure 7.4 provides a diagram of the multi-head group-node attention portion of the model where we see the separate linear layers associated with each of the weight matrices.

Attention applied to graphs is most often used in the form of self-attention. In that situation, a node embedding is computed based on the representation of adjacent nodes in earlier layers. In the proposed model, we are using group-node attention rather than self-attention. That is, the hidden representation of a group depends on a learned transform of group and group-adjacent nodes that is conditioned on group features by way of the $\mathbf{Q}$ matrix used to compute the attention coefficients. The output of the attention layer is then an aggregation of the node feature vectors $\mathbf{X}_i$ associated with group i into a single hidden representation $\mathbf{h}_X$ dependent on the group features $\mathbf{g}_i$.

There are three fully-connected layers used for input transformations: FCN_X, FCN_q, and FCN_g. Each performs a linear transform of its input and then a non-linear activation. For example,

$$\text{FCN}_g(\mathbf{g}_i^t) = \mathsf{ReLU}(\mathbf{W}_g \mathbf{g}_i^t + \mathbf{b}_g)\,, \tag{7.1}$$

where $\mathbf{W}_g \in \mathbb{R}^{D_g \times D_m}$ is a weight matrix and $\mathbf{b}_g$ is a bias vector. These transforms are used to reshape the input to match the appropriate dimensions used by the model and each has its own weight and bias parameters.

To simplify notation we will introduce several intermediate variables which correspond to the output of these transformations:

$$\mathbf{z}_q = \mathrm{FCN}_q(\mathbf{g}_i^t)\,, \tag{7.2}$$

$$\mathbf{Z}_X = \mathrm{FCN}_X(\mathbf{X}_i^t \,\|\, \mathbf{m}_i^t)\,, \tag{7.3}$$

$$\mathbf{h}_g = \mathrm{FCN}_g(\mathbf{g}_i^t)\,. \tag{7.4}$$

For readability we will drop the i and t annotations when referring to the intermediate variables. An individual input to the model is for a single group i at a snapshot t.

We use $\mathbf{h}_g$ directly to predict classification labels, but $\mathbf{Z}_X$ and $\mathbf{z}_q$ are used for the group-node attention layer that learns a representation of groups from node features. The $\mathbf{z}_q$ vector is used to construct the query matrix $\mathbf{Q}$ and $\mathbf{Z}_X$ to construct the key and value matrices $\mathbf{K}$ and $\mathbf{V}$.

The group-node attention layer uses multi-head attention in order to potentially learn multiple group representations. The output of each head is provided by $\textsc{GNAttHead}(\cdot)$ defined as

$$\textsc{GNAttHead}(\mathbf{Z}_X, \mathbf{z}_q) = \boldsymbol{\alpha}\mathbf{V}, \tag{7.5}$$

where $\boldsymbol{\alpha}$ is a attention coefficient vector and $\mathbf{V}$ is a value matrix.

We use scaled dot-product attention where the attention coefficients are

$$\boldsymbol{\alpha} = \mathsf{softmax}(\frac{\mathbf{Q}\mathbf{K}^\mathsf{T}}{\sqrt{D_m}}). \tag{7.6}$$

The query $\mathbf{Q}$, key $\mathbf{K}$, and $\mathbf{V}$ matrices are defined as

$$\mathbf{Q} = \mathbf{z}_q \mathbf{W}^Q \,, \tag{7.7}$$

$$\mathbf{K} = \mathbf{Z}_X \mathbf{W}^K \,, \tag{7.8}$$

$$\mathbf{V} = \mathbf{Z}_X \mathbf{W}^V \tag{7.9}$$

where $\mathbf{W}^Q$, $\mathbf{W}^K$, and $\mathbf{W}^V$ are learned weight parameters; and $\mathbf{Q} \in \mathbb{R}^{1 \times D_k}$, $\mathbf{K} \in \mathbb{R}^{N \times D_k}$, and $\mathbf{V} \in \mathbb{R}^{N \times D_m}$. Since we use the transformed group feature vector $\mathbf{z}_q \in \mathbb{R}^{D_q}$ to construct the attention coefficients $\boldsymbol{\alpha}$ we are calculating a weighted average over the node embeddings of the group members and group neighbors

$$\text{GNAttHead}(\mathbf{Z}_X, \mathbf{z}_q) = \mathsf{softmax}(\frac{\mathbf{z}_q \mathbf{W}^Q (\mathbf{Z}_X \mathbf{W}^K)^\mathsf{T}}{\sqrt{D_m}}) \mathbf{Z}_X \mathbf{W}^V . \tag{7.10}$$

The output of each head is passed through a ReLU activation and then all are concatenated ($\|$) together and passed through a final transformation,

$$\text{GNAtt}(\mathbf{Z}_X, \mathbf{z}_q) = (\|_h^H \, \mathsf{ReLU}(\text{GNAttHead}_h(\mathbf{Z}_X, \mathbf{z}_q))) \mathbf{W}^O \,, \tag{7.11}$$

where $\mathbf{W}^O \in \mathbb{R}^{HD_v \times D_m}$ is a weight matrix parameter that mixes the results of the group heads.

Finally, the output of GNAtt is concatenated with $\mathbf{h}_g$ and passed through an output layer

$$\mathbf{h}_X = \text{GNAtt}(\mathbf{Z}_X, \mathbf{z}_q) \,, \tag{7.12}$$

$$\tilde{\mathbf{y}} = \text{FCN}_{\mathsf{out}}(\mathbf{h}_X \,\|\, \mathbf{h}_g) \,, \tag{7.13}$$

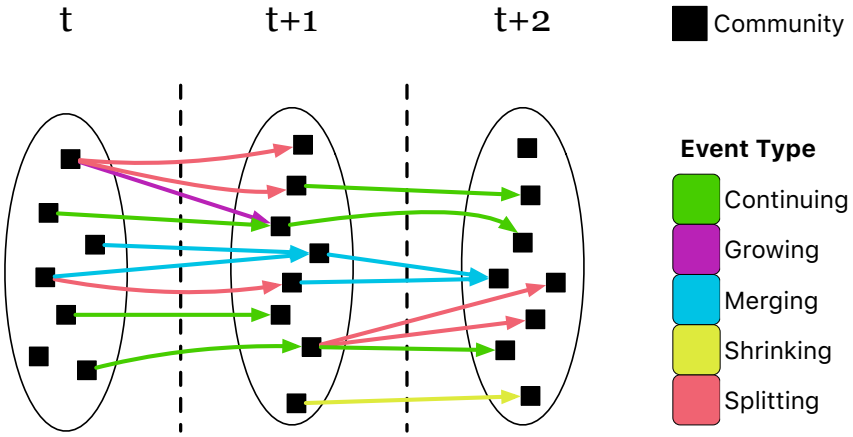
where $\text{FCN}_{\mathsf{out}}$ uses a sigmoid activation function to scale the predicted class label probabilities and $\tilde{\mathbf{y}} \in \mathbb{R}^E$.

7.3 Experiments

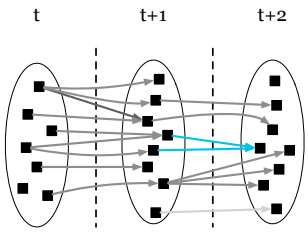
7.3.1 Community Detection and Tracking

The model presented in this chapter predicts community evolution events for communities defined over a series of network snapshots. This requires both a method for detecting communities and tracking the same communities across snapshots. As outlined in Section 3.3, there are many existing community tracking frameworks. The proposed model does not depend on any specific framework and we select GED [95] for our experiments as it is well-established in the literature and supports overlapping communities. GED uses two parameters—α and β— for labeling events, we use the standard values of 0.5 for both parameters. We use the clique percolation method (CPM) [4] on clique graphs as described in [122] to define the communities for each network snapshot. The used implementation of CPM constructs a clique graph and then merges cliques that share a majority of members. CPM supports overlapping communities and uses cliques as primitives for constructing communities—this matches the intuition of the proposed model and expected structure of social interaction networks where communities are dense, overlapping graph regions [4].

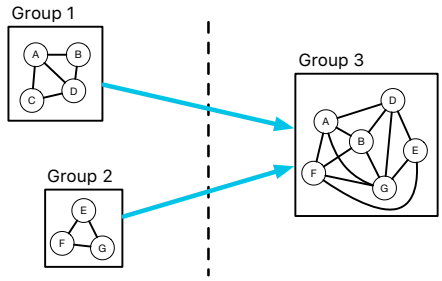
Figure 7.5 shows an example of labeling community evolution events for the community evolution prediction task. In particular, we construct snapshots with a fixed temporal window length and identify community evolution events as relationships between communities in successive snapshots. The network snapshots are constructed from social interaction data, and we adopt a methodology used by [30] to determine a maximum edge deactivation threshold to prevent densification (Chapter 5) [15]. Edges are removed if they are inactive for longer than the edge deactivation threshold. Any future activity will cause the edge to be reformed.



(a) Labeling community evolution events between pairs of communities in consecutive network snapshots.



(b) An instance of communities merging.



(c) A closer look at two merging communities.

Figure 7.5: Community evolution events across three network snapshots.

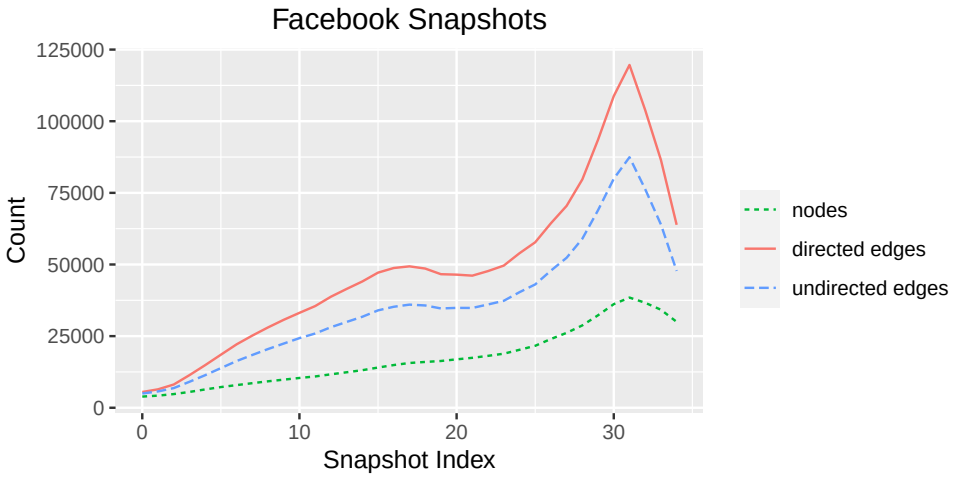


Figure 7.6: Facebook snapshot network size statistics.

7.3.2 Datasets

We use several datasets that have been previously used for community tracking and community evolution prediction: the Facebook WOSN [108] and Scratch [102] datasets. Following the methodology in [30], we found maximum edge deactivation thresholds of six months (165 days) and three months (62 days) for the Facebook and Scratch datasets. Due to the size of the Scratch networks and computational requirements of pre-processing, we elected to use a smaller threshold of four weeks. The Facebook dataset extends from July 2006 until April 2009 and each network snapshot includes activity for a month. The Scratch dataset includes data from May 2010 until May 2011 and each snapshot includes two weeks worth of activity. For both datasets we include all communities with four or more members as we found this reduces noise in the data caused by groups of three forming and dissolving. Larger groups are more likely to participate in various kinds of community evolution events.

Figures 7.6 and 7.7 show the number of nodes, directed edges, and undirected edges for the series of network snapshots for both Facebook and Scratch. We note that while the number of nodes and edges for both series change over time, the Scratch snapshots seem to be more stable, especially with respect to the number of nodes. This may also be due to

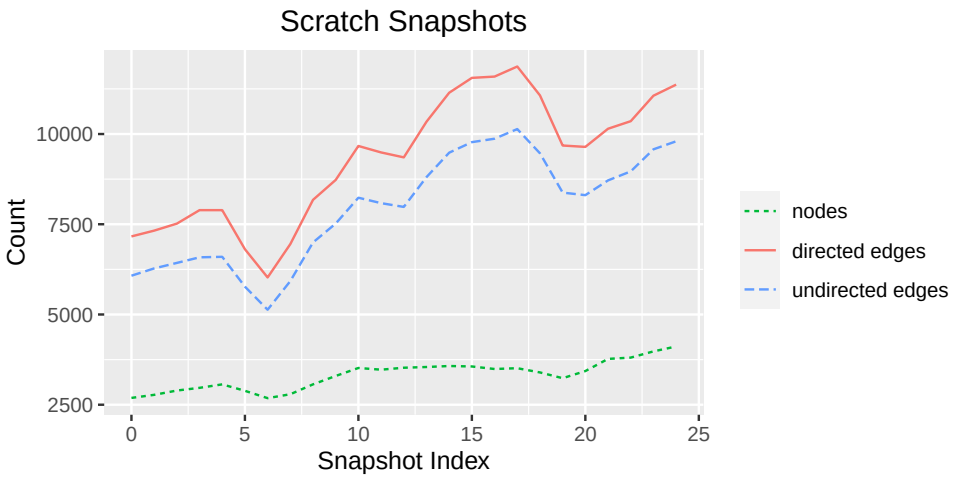
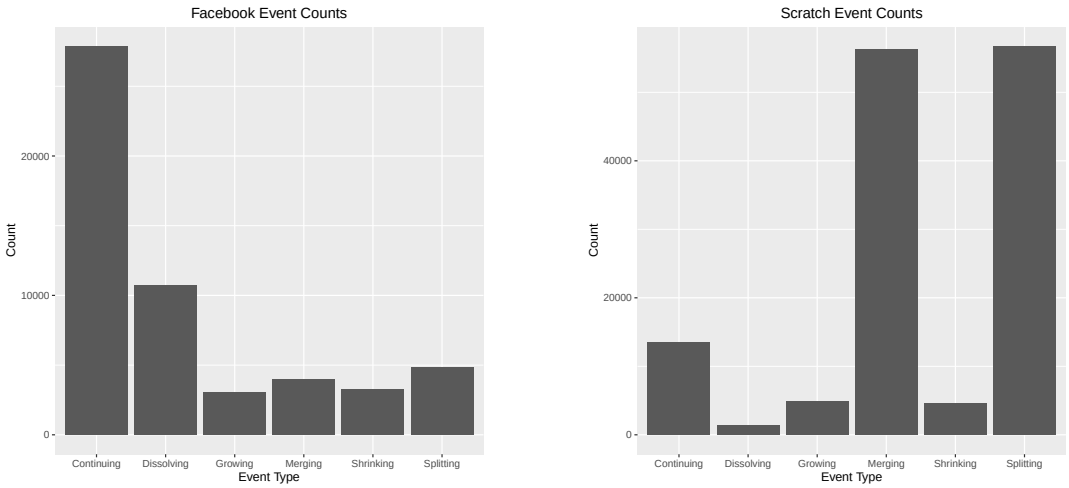


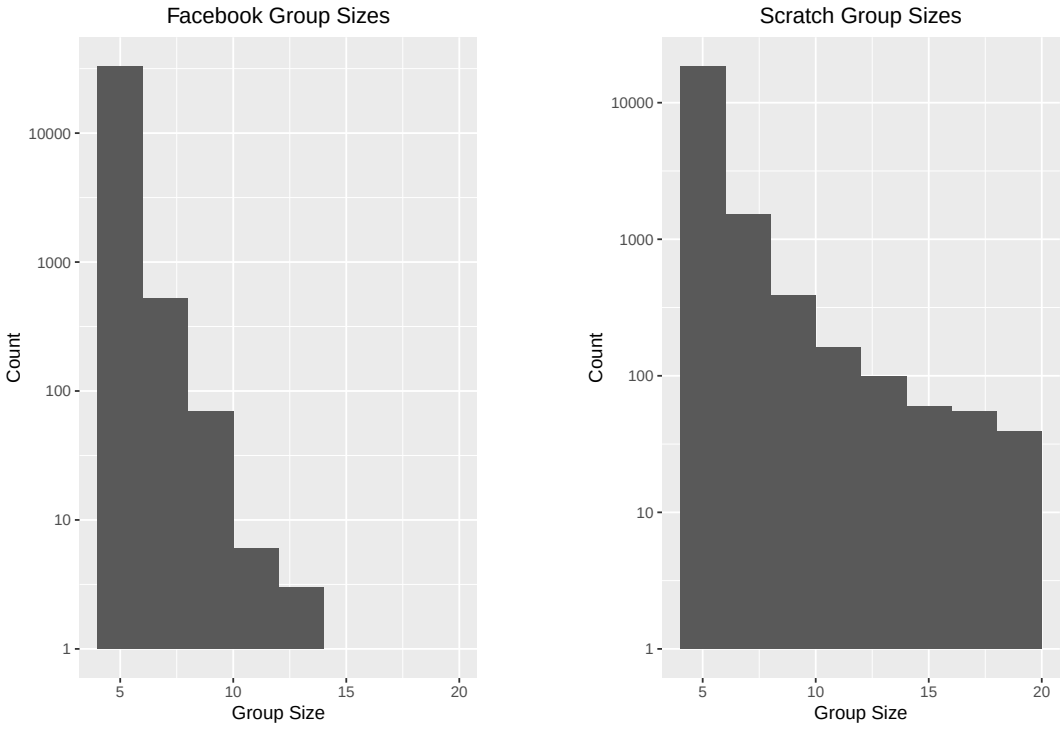
Figure 7.7: Scratch snapshot network size statistics.



(a) Facebook event counts

(b) Scratch event counts

Figure 7.8: Community evolution event counts for the datasets.



(a) Facebook group sizes

(b) Scratch group sizes

Figure 7.9: Distribution of group sizes in datasets. There are an additional 139 groups with size greater than 20 and 20 groups with size greater than 100 in the Scratch network snapshots. The largest group has size 687. The largest group in the Facebook network snapshots has 14 members.

the additional pre-processing steps to reduce the total number of edges and groups when generating the Scratch snapshots. We see that *continuing* and *dissolving* events are the most frequent in the Facebook dataset, while *merging* and *splitting* are the most common in the Scratch dataset. This may be due to the nature of the networks—Facebook is primarily used to connect with people already known by a user, but Scratch encourages creating new relationships through project collaboration. The distribution of events for both datasets can be found in Figure 7.8.

As shown in Figure 7.9, most groups in both datasets are smaller. Because of this we expect the use of group-relative position as a performance optimization—rather than the masked attention based on the adjacency matrix—should have minimal effect on the learned

Table 7.2: Node features

	Name	Description
1	*In-degree*	Count of incoming edges at snapshot t
2	*Out-degree*	Count of outgoing edges at snapshot t
3	*Previous in-degree*	Count of incoming edges at snapshot $t-1$
4	*Previous out-degree*	Count of outgoing edges at snapshot $t-1$

Table 7.3: Group features

	Name	Description
1	*Density*	The ratio of actual edges to potential edges among group members
2	*Group affinity*	The ratio of edges between group members over all edges of all group members
3	*Size*	Number of member nodes
4	*Event counts*	Counts of the incoming event types from the previous snapshot

group representation since all members of a smaller group are within two hops of each other.

7.3.3 Methodology

For comparison to GNAN, we trained instances for four baseline methods selected because of their common use for community evolution prediction and for their diversity. The baseline methods used are CART decision trees, logistic regression, multilayer perceptron (MLP), and SVM with a radial basis function (RBF) kernel. The implementations from [123] are used for CART, logistic regression, and SVM. Those implementations support a class weights parameter which was configured to *balanced* to adjust for the class imbalance in the dataset. None of these three baselines directly support multi-label classification so an instance of the classifier was trained for each community evolution event label. We performed a sweep over the regularization parameter (C) for SVM using the values: 0.01, 0.1, 1, 10, and 100. The GNAN and MLP models are both implemented with [124], use a model size of 16 dimensions for embedding layers, and were optimized using AdamW with a learning rate of 0.001 and weight decay of 0.01. The MLP models have three layers. The binary cross-entropy loss function was used and training stops after five consecutive

Table 7.4: Mean AUC scores for events in the Facebook dataset. Highest values are in bold. The $^{\bullet}/^{\circ}$ annotation indicates whether GNAN is statistically superior/inferior to the other method. A two-sided Wilcoxon signed-rank test was used at 95% significance level.

Method	Cont.	Dis.	Grow	Merge	Shrink	Split	Macro Avg.
CART	$0.526^{\bullet}$	$0.515^{\bullet}$	$0.508^{\bullet}$	$0.532^{\bullet}$	$0.570^{\bullet}$	$0.584^{\bullet}$	$0.539^{\bullet}$
Log. Reg.	$0.581^{\bullet}$	$0.590^{\bullet}$	$0.552^{\bullet}$	$0.695^{\bullet}$	$0.923^{\bullet}$	$0.892^{\bullet}$	$0.706^{\bullet}$
MLP	0.620	$0.596^{\bullet}$	$0.571^{\bullet}$	0.718	**0.939**	$0.962^{\bullet}$	$0.734^{\bullet}$
SVM	$0.585^{\bullet}$	$0.590^{\bullet}$	$0.556^{\bullet}$	$0.693^{\bullet}$	$0.916^{\bullet}$	$0.884^{\bullet}$	$0.704^{\bullet}$
GNAN	**0.636**	**0.617**	**0.603**	**0.757**	**0.939**	**0.966**	**0.753**

epochs without a decrease in total loss when tested against the validation data. Since none of the baselines support variable-sized input, the features for group member and neighbor nodes were provided as two separate vectors—one for group members and one for group neighbors—containing the mean average of the feature values. Those two vectors were concatenated with the group features to form a single vector input.

The Holdout method with random splits is used for comparing the proposed method with the standard models used in community evolution prediction. We perform 30 random splits of the network snapshots into training, validation, and test sets. All network snapshots before the split are used for training and validation, the remaining future network snapshots can be used for testing. The training and validation sets are randomly split from all the examples in all the snapshots that occur before the randomly-selected split. The random splits are selected from the interval $[5, T]$ where T is the final snapshot. This guarantees a minimum number of training examples are made available to the models. In order to address the non-determinism and sensitivity to parameter values for some of the models included in this experiment, we train five instances of each model for each random split. We select one of the five models for each split based on the macro-averaged AUC to use for evaluation with the test data. We only use the groups from the snapshot after the split for evaluation.

Table 7.5: Mean AUC scores for events in the Scratch dataset. Highest values are in bold.

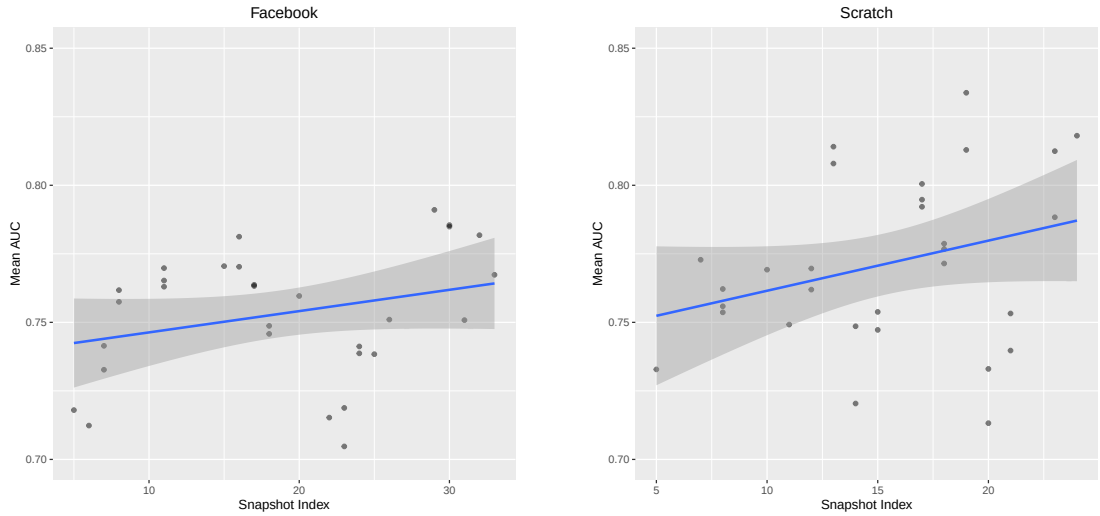
Method	Cont.	Dis.	Grow	Merge	Shrink	Split	Macro Avg.
CART	0.533$^{\bullet}$	0.590$^{\bullet}$	0.517$^{\bullet}$	0.642$^{\bullet}$	0.570$^{\bullet}$	0.670$^{\bullet}$	0.586$^{\bullet}$
Log. Reg.	0.588$^{\bullet}$	0.784	0.584$^{\bullet}$	0.800	0.681$^{\bullet}$	0.828$^{\bullet}$	0.710$^{\bullet}$
MLP	0.621	0.814	0.610	0.819	0.778	**0.908**	0.756
SVM	0.588$^{\bullet}$	0.784	0.569$^{\bullet}$	0.749$^{\bullet}$	0.751$^{\bullet}$	0.824$^{\bullet}$	0.710$^{\bullet}$
GNAN	**0.631**	**0.827**	**0.636**	**0.845**	**0.782**	0.907	**0.769**

7.3.4 Comparative Results

The results of the evaluation of GNAN against the baselines are shown in Tables 7.4 and 7.5. The mean AUC scores are used for the evaluation as they capture the overall comparative performance. We see that GNAN generally outperforms all baselines on both datasets with the exception of the shrinking event in Facebook and the splitting event in Scratch. The most competitive baseline is the multilayer perceptron (MLP).

In the Facebook snapshot series, we see that all methods other than CART perform well for predicting shrinking and splitting events. This is likely due to group size and previous snapshot event counts being good indicators of shrinking and splitting. We notice the same is true for the splitting event in the Scratch dataset, but prediction of the shrinking event seems to be more challenging.

The Scratch networks were constructed from social interactions and we required there be at least four interactions between a pair of nodes before including the edge in the networks. This was done in order to improve the performance of community detection with CPM; however, we expect that including that missing structural information would further improve the relative performance of GNAN over the baselines for the Scratch networks. Similarly, the Scratch networks do contain some larger groups and extending GNAN with self-attention for nodes may also increase model performance that by capturing substructure within those large communities.



(a) Facebook mean AUC

(b) Scratch mean AUC

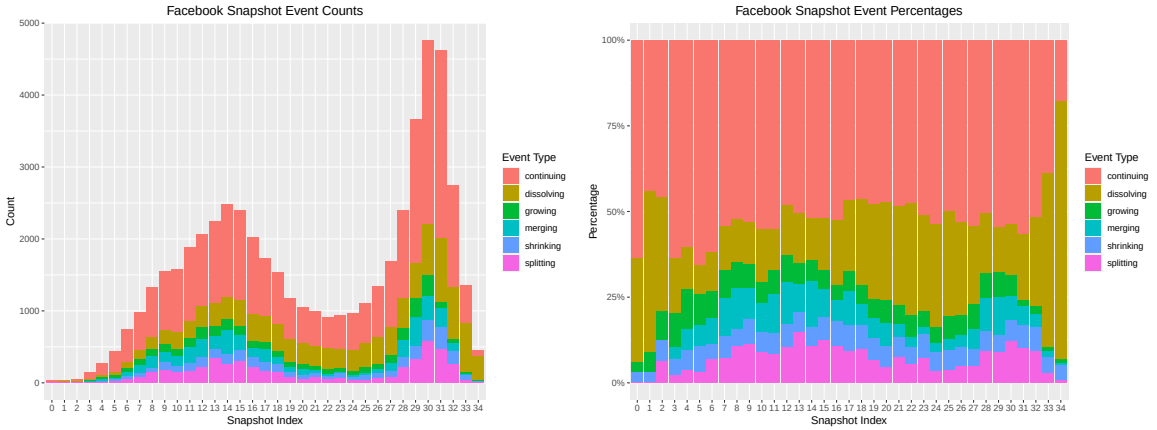
Figure 7.10: The mean AUC scores for GNAN on both the Facebook and Scratch network snapshots.

7.3.5 Temporal Effects

In addition to the comparative experiment, analyses were performed to evaluate the performance of GNAN across snapshots and as additional, earlier network snapshots were added.

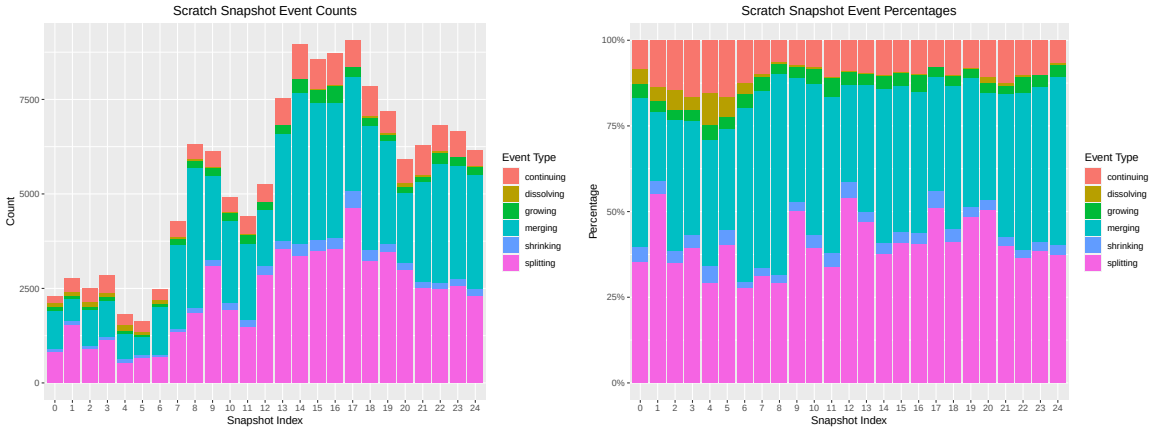
In Figure 7.10, we see the mean AUC scores for the GNAN model instances trained for the comparative analysis in Section 7.3.4 on both the Facebook and Scratch network snapshots. We see a general trend of higher scores in later snapshots for both datasets, but there are also decreases in score values for particular ranges of snapshots. The general increasing trend of scores might be attributed to having more training data. If we consider the changes in network activity over the snapshots in Figures 7.6 and 7.7 we can see that GNAN performance appears to correlate with network activity. Using the number of undirected edges as an indicator of network activity, we calculate Spearman's rank correlation coefficient and find that the mean AUC is slightly correlated with network activity for the Facebook snapshots, $\rho = .2956,\ p = 0.1$, and for the Scratch snapshots, $\rho = .3501,\ p = .06$.

A decrease in prediction performance during periods of reduced network activity may indicate that the model is missing additional information useful for predicting community



(a) Facebook event counts

(b) Facebook event percentages



(c) Scratch event counts

(d) Scratch event percentages

Figure 7.11: The event counts and percentages over network snapshots for both the Facebook and Scratch datasets.

evolution events. Notably, events external to the network—such as start/stop of academic semesters or holidays—may impact node behavior and result in structural changes.

The evolution event counts in Figure 7.11 show a positive correlation between community evolution event counts and network activity when considered along with the network activity figures (Figures 7.6 and 7.7). Additionally, the event percentage plots in Figure 7.11 reveal that the distribution of events changes over time. This is most obvious in the Facebook network snapshot series where we see the proportion of dissolving events is negatively correlated with increased network activity, while continuing, growing, merging, and

splitting events all appear to be positively correlated with increased network activity.

While all previous snapshots were used in training GNAN and the baseline models for the comparative evaluation in Section 7.3.4, additional GNAN model instances were trained with incrementally larger snapshot intervals to determine the performance impact of adding additional historic information to the model. Snapshots for evaluation were selected by starting at the final snapshot of each dataset and adding every fifth snapshot index. For each evaluation snapshot index, training data was provided in increasingly larger intervals with a stride of five. When fewer than five snapshots remain then all additional snapshots are added to the final interval of training snapshots. For example, for the evaluation snapshot index of 14, there would be three training intervals: $[9, 13]$, $[4, 13]$, $[1, 13]$. Five instances of GNAN are trained for each of these three intervals of training data and then validation is used to select the best model for each training interval—that model is used for evaluation.

The results provided in Figures 7.12 and 7.13 show that while generally more training data improves model performance, including data from earlier snapshots can negatively impact prediction performance for certain evolution events. The network activity (Figures 7.6 and 7.7) of Facebook and Scratch network snapshots and the distributions of evolution events (Figure 7.11) change over time and this can affect model training.

For Facebook, the prediction performance of GNAN on shrinking and splitting events appears to be consistent across all evaluation snapshots. While not as tightly grouped as the Facebook AUC scores, the shrinking and splitting events also have the lowest variance in score across snapshots for the Scratch dataset. The prediction performance for all other events in both datasets appear to be more dependent on the evaluation snapshot used.

Consider the prediction of growing events for the Facebook evaluation snapshot at index 34. We see that using only snapshots 29-33 improves performance compared to using snapshots 24-33. However, adding the five next earlier snapshots such that all snapshots 19-33 are used increases performance again. According to Figure 7.11a, there are growing training examples gained by including all of the earlier snapshots. From Figure 7.6, we see that while the number of edges decline for a period over snapshots 29-33 and snapshots 19-23

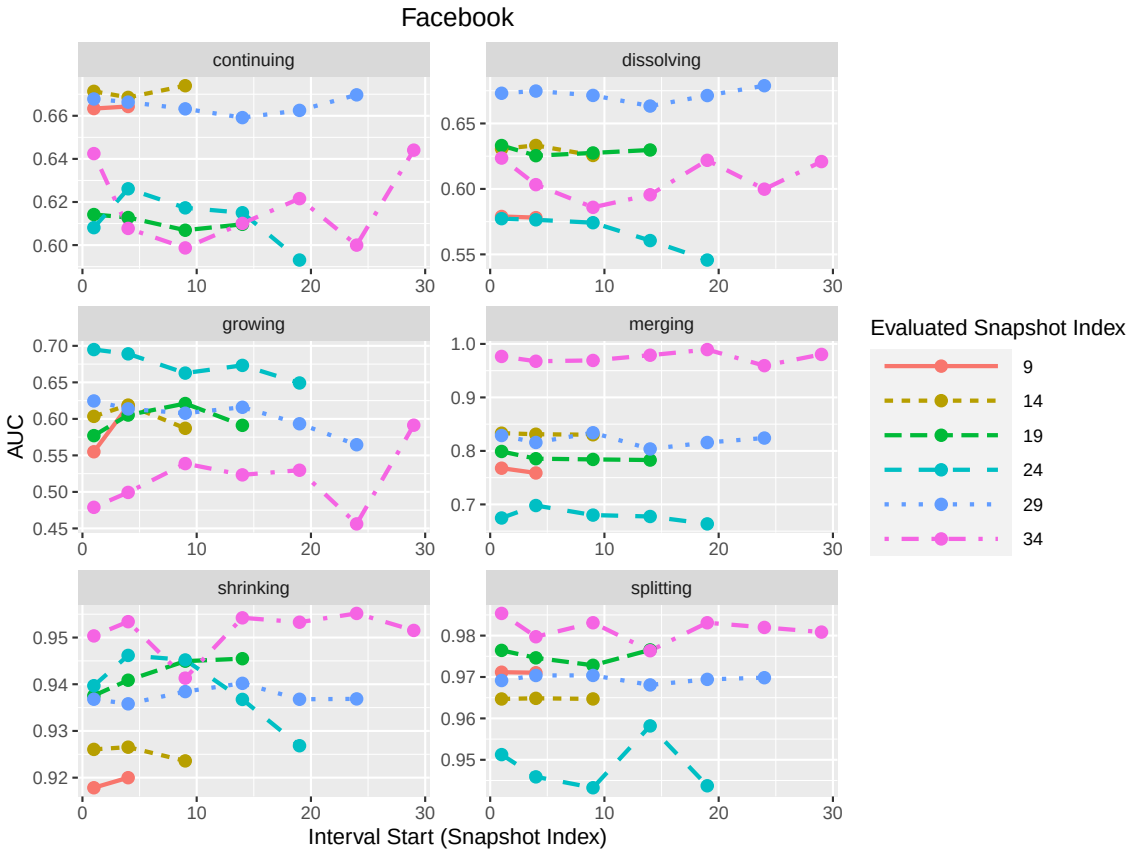


Figure 7.12: GNAN model performance as earlier training data is introduced for the Facebook dataset.

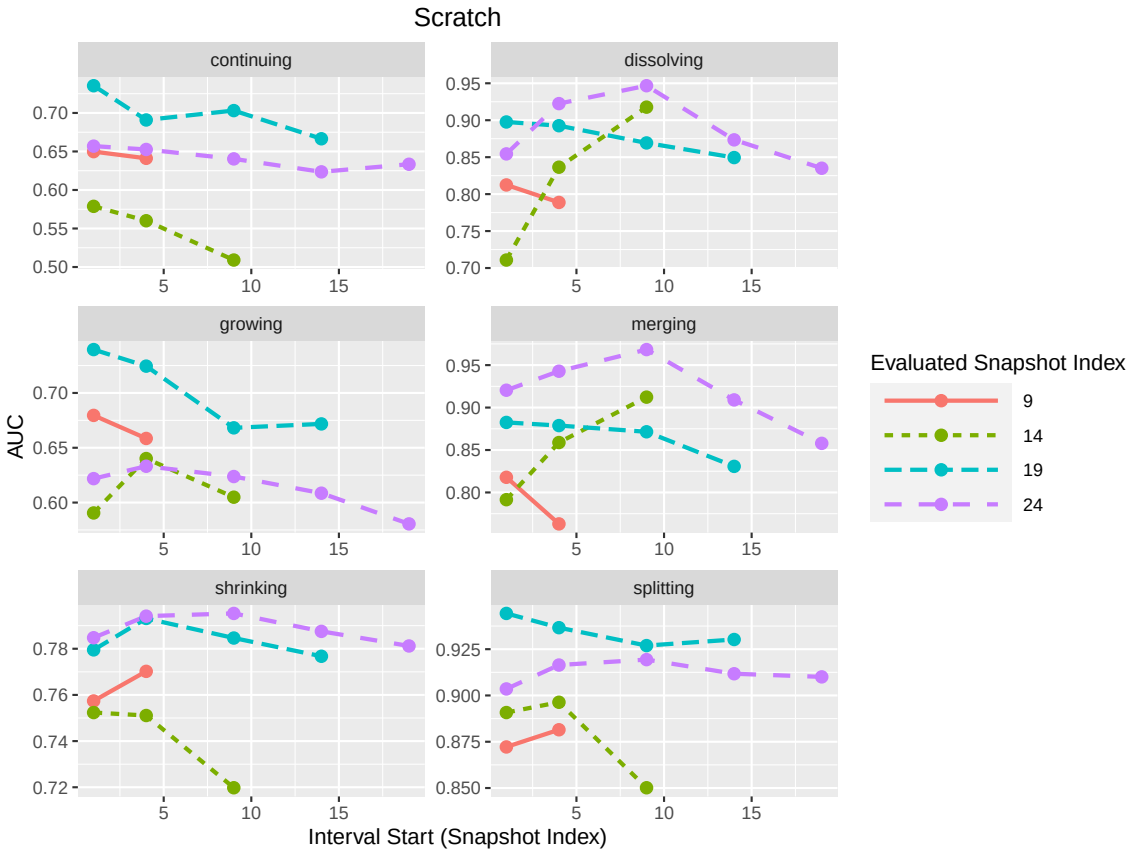


Figure 7.13: GNAN model performance as earlier training data is introduced for the Scratch dataset.

there is only a growth of the number of edges in the snapshots 24-29. The reason for the drop of performance may then be due to the predictors for growing events being different between periods of increased network activity and periods of decreased network activity. It appears that more training examples for the growing event only improve prediction performance when those examples are taken from snapshots with network activity similar to that of the evaluation snapshot. This relationship between network activity in the training snapshots and the evaluation snapshot suggests that model performance may be improved by selecting training snapshots that capture similar network trends as the evaluation snapshot.

Chapter 8: Conclusion and Future Research

8.1 Conclusion

Social networks of all kinds continue to affect our daily lives in a myriad of ways and inspire an active research community. Identifying communities, uncovering the roles held by individuals, and predicting network dynamics are all aspects of uncovering structure in social networks. The ultimate impact of research in this area can be significant as it informs us how to nurture learning in MOOCs [119, 120], interrupt radicalization in at-risk populations [125], dismantle criminal organizations [121], and improve civic discourse by identifying and exposing misinformation campaigns [126].

In this dissertation, I have presented novel methods for community detection, persistent role discovery, and community evolution prediction. I have also shown that previous assumptions about community structure and network topology for social interaction networks may be attributed to data preprocessing methodology rather than being an innate characteristic of social networks based on social interactions.

A probabilistic approach to community detection that outputs node memberships and community topics (SENC) was introduced. The bounded seed groups enable SENC to account for differences in underlying community structure across many networks. The output produced by SENC is highly interpretable. We can understand the nature of a discovered community by examining its topic distribution. We can also review a node's relative community involvement through its membership weights.

Evidence was presented that temporal artifacts may be introduced in social networks when the relationships represented by edges require allocation of inelastic resource such as time or attention. Our findings suggest more accurate social networks may be derived from ongoing dyadic interactions rather than one-time events such as "following" or "friending."

A methodology for identifying persistent roles across time and datasets was presented. Using this methodology, we find the same six user roles which capture distinct structural positions in 26 network snapshots from two online social networks. To my knowledge, this was the first work to present evidence of persistent roles independently derived from multiple datasets. Beyond the discovery of persistent roles, we provide an analysis of the roles and show there are differences in role membership and interaction across the snapshots.

A graph neural network with group-node attention for community evolution prediction (GNAN) was introduced. GNAN is able to incorporate both spatial and temporal information of individual member and neighbor nodes by way of group-node attention. The model is capable of learning a group representation for community evolution prediction.

8.2 Future Research

Though the work presented here has focused on social network analysis, I plan to extend and apply my work to other domains with variable-sized structured data. I am specifically interested in the domain of program analysis and how graph attention models which incorporate the concepts of community detection, role discovery, and representation learning can be used in this different setting. I plan to apply an extension of GNAN for program analysis tasks such as type recovery from program binaries. This application domain is well-suited for models based on graph attention networks. In the case of the type recovery problem, generating large datasets with labels is straightforward as the type information is available from program source code in common languages such as C and C++. This resolves one of the largest challenges in applied machine learning and data mining for social networks: a lack of ground truth and objective class labels.

While we found GNAN to be an improvement over baseline classifiers for predicting community evolution events, there are several ways in which the model can be extended and there are additional domains where learning group embeddings, or even hierarchical group embeddings, may be useful. I plan to extend GNAN with masked self-attention at the group level, support for stacked group-node attention layers that incorporate both

self-attention of nodes with group-node attention, and optimization for training on GPUs.

I had considered developing a model for predicting network role transitions based on the persistent roles presented in Chapter 6 before I began developing the GNAN model. I ultimately chose to pursue community evolution prediction instead as there were more existing frameworks and methodologies that could be used as a setting for the prediction task. With the surge of research on dynamic networks, higher-order node embeddings, and GNNs; I believe much of the foundation for network role prediction is now in place.

Bibliography

Bibliography

[1] M. Revelle, C. Domeniconi, M. Sweeney, and A. Johri, "Finding community topics and membership in graphs," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2015, pp. 625–640.

[2] J. Yang, J. McAuley, and J. Leskovec, "Community detection in networks with node attributes," in *IEEE 13th International Conference on Data Mining*. IEEE, 2013, pp. 1151–1156.

[3] ——, "Detecting cohesive and 2-mode communities in directed and undirected networks," in *Proceedings of the ACM International Conference on Web Search and Data Mining*. ACM, 2014, pp. 323–332.

[4] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *nature*, vol. 435, no. 7043, pp. 814–818, 2005.

[5] M. E. Newman, "Modularity and community structure in networks," *Proceedings of the national academy of sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[6] J. Yang and J. Leskovec, "Structure and overlaps of ground-truth communities in networks," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 5, no. 2, pp. 1–35, 2014.

[7] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: densification laws, shrinking diameters and possible explanations," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, 2005, pp. 177–187.

[8] K. Henderson, B. Gallagher, T. Eliassi-Rad, H. Tong, S. Basu, L. Akoglu, D. Koutra, C. Faloutsos, and L. Li, "Rolx: structural role extraction & mining in large graphs," in *Proceedings of the eighteenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 1231–1239.

[9] R. A. Rossi, B. Gallagher, J. Neville, and K. Henderson, "Modeling dynamic behavior in large evolving graphs," in *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 2013, pp. 667–676.

[10] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, Jun. 2014.

[11] J. Kunegis, "KONECT – The Koblenz Network Collection," in *Proc. Int. Conf. on World Wide Web Companion*, 2013, pp. 1343–1350. [Online]. Available: http://dl.acm.org/citation.cfm?id=2488173

[12] L. Peel, D. B. Larremore, and A. Clauset, "The ground truth about metadata and community detection in networks," *Science advances*, vol. 3, no. 5, p. e1602548, 2017.

[13] D. Hric, R. K. Darst, and S. Fortunato, "Community detection in networks: Structural communities versus ground truth," *Physical Review E*, vol. 90, no. 6, p. 062805, 2014.

[14] M. Newman, *Networks: An Introduction*. New York: Oxford University Press, 2010.

[15] M. Revelle, C. Domeniconi, and A. Johri, "Temporal artifacts from edge accumulation in social interaction networks," in *Italian Workshop on Neural Nets*. Springer, 2017, pp. 11–21.

[16] ——, "Evidence of temporal artifacts in social networks," in *Proceedings of the 6th International Workshop on Mining Ubiquitous and Social Environments (MUSE)*, 2015, pp. 35–42.

[17] ——, "Persistent roles in online social networks," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2016, pp. 47–62.

[18] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3, pp. 75–174, 2010.

[19] Z. Zhao, S. Feng, Q. Wang, J. Z. Huang, G. J. Williams, and J. Fan, "Topic oriented community detection through social objects and link analysis in social networks," *Knowledge-Based Systems*, vol. 26, pp. 164–173, 2012.

[20] J. Leskovec and J. McAuley, "Learning to discover social circles in ego networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 539–547.

[21] H. Deng, J. Han, B. Zhao, Y. Yu, and C. X. Lin, "Probabilistic topic models with biased propagation on heterogeneous information networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2011, pp. 1271–1279.

[22] H.-P. Kriegel, P. Kröger, and A. Zimek, "Subspace clustering," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 4, pp. 351–364, 2012.

[23] C. Domeniconi, D. Papadopoulos, D. Gunopulos, and S. Ma, "Subspace clustering of high dimensional data." in *Proceedings of the SIAM International Conference on Data Mining*. SIAM, 2004, pp. 517–521.

[24] D. M. Blei, "Probabilistic topic models," *Communications of the ACM*, vol. 55, no. 4, pp. 77–84, 2012.

[25] A. McCallum, X. Wang, and N. Mohanty, "Joint group and topic discovery from relations and text," in *ICML Workshop on Statistical Network Analysis*. Springer, 2006, pp. 28–44.

[26] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Technical Report 1999-66, November 1999, previous number = SIDL-WP-1999-0120. [Online]. Available: http://ilpubs.stanford.edu:8090/422/

[27] B. Gonçalves, N. Perra, and A. Vespignani, "Modeling users' activity on twitter networks: Validation of dunbar's number," *PloS one*, vol. 6, no. 8, p. e22656, 2011.

[28] M. T. Rivera, S. B. Soderstrom, and B. Uzzi, "Dynamics of dyads in social networks: Assortative, relational, and proximity mechanisms," *annual Review of Sociology*, vol. 36, pp. 91–115, 2010.

[29] A.-L. Barabasi, "The origin of bursts and heavy tails in human dynamics," *Nature*, vol. 435, no. 7039, pp. 207–211, 2005.

[30] G. Miritello, E. Moro, R. Lara, R. Martínez-López, J. Belchamber, S. G. Roberts, and R. I. Dunbar, "Time as a limited resource: Communication strategy in mobile phone networks," *Social Networks*, vol. 35, no. 1, pp. 89–95, 2013.

[31] G. Miritello, R. Lara, M. Cebrian, and E. Moro, "Limited communication capacity unveils strategies for human interaction," *Scientific reports*, vol. 3, 2013.

[32] G. Miritello, R. Lara, and E. Moro, "Time allocation in social networks: correlation between social structure and human communication dynamics," in *Temporal Networks*. Springer, 2013, pp. 175–190.

[33] C. A. Hidalgo and C. Rodriguez-Sickert, "The dynamics of a mobile phone network," *Physica A: Statistical Mechanics and its Applications*, vol. 387, no. 12, pp. 3017–3024, 2008.

[34] G. Kossinets and D. J. Watts, "Empirical analysis of an evolving social network," *Science*, vol. 311, no. 5757, pp. 88–90, 2006.

[35] R. Kumar, J. Novak, and A. Tomkins, "Structure and evolution of online social networks," in *Link mining: models, algorithms, and applications*. Springer, 2010, pp. 337–357.

[36] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, p. 2, 2007.

[37] J. Yang and J. Leskovec, "Community-affiliation graph model for overlapping network community detection," in *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 2012, pp. 1170–1175.

[38] R. Yu, X. He, and Y. Liu, "Glad: group anomaly detection in social media analysis," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 372–381.

[39] R. Rossi, N. K. Ahmed *et al.*, "Role discovery in networks," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 27, no. 4, pp. 1112–1131, 2015.

[40] Y. Ruan and S. Parthasarathy, "Simultaneous detection of communities and roles from large networks," in *Proceedings of the second edition of the ACM conference on Online social networks.* ACM, 2014, pp. 203–214.

[41] Y. Han and J. Tang, "Probabilistic community and role model for social networks," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 2015, pp. 407–416.

[42] W. X. Zhao, J. Wang, Y. He, J.-Y. Nie, J.-R. Wen, and X. Li, "Incorporating social role theory into topic models for social media content analysis," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 27, no. 4, pp. 1032–1044, 2015.

[43] K. Henderson, B. Gallagher, L. Li, L. Akoglu, T. Eliassi-Rad, H. Tong, and C. Faloutsos, "It's who you know: graph mining using recursive structural features," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2011, pp. 663–671.

[44] S. Gilpin, T. Eliassi-Rad, and I. Davidson, "Guided learning for role discovery (glrd): framework, algorithms, and applications," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2013, pp. 113–121.

[45] Y. Cheng, A. Agrawal, A. Choudhary, H. Liu, and T. Zhang, "Social role identification via dual uncertainty minimization regularization," in *Data Mining (ICDM), 2014 IEEE International Conference on.* IEEE, 2014, pp. 767–772.

[46] Y. Zhao, G. Wang, P. S. Yu, S. Liu, and S. Zhang, "Inferring social roles and statuses in social networks," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2013, pp. 695–703.

[47] S. Choobdar, P. Ribeiro, S. Parthasarathy, and F. Silva, "Dynamic inference of social roles in information cascades," *Data Mining and Knowledge Discovery*, pp. 1–26, 2014.

[48] J. Chan, C. Hayes, and E. M. Daly, "Decomposing discussion forums and boards using user roles." in *Proceedings of the 4th International Conference on Web and Social Media*, vol. 10, 2010, pp. 215–218.

[49] A. J. White, J. Chan, C. Hayes, and B. Murphy, "Mixed membership models for exploring user roles in online fora." in *Proceedings of the 6th International Conference on Web and Social Media*, 2012.

[50] S. Fortunato, "Community detection in graphs," *Physics reports*, vol. 486, no. 3-5, pp. 75–174, 2010.

[51] M. Atzmueller, S. Günnemann, and A. Zimmermann, "Mining communities and their descriptions on attributed graphs: a survey," *Data Mining and Knowledge Discovery*, pp. 1–27, 2021.

[52] S. Günnemann, I. Färber, B. Boden, and T. Seidl, "Subspace clustering meets dense subgraph mining: A synthesis of two paradigms," in *Proceedings of the IEEE International Conference on Data Mining.* IEEE Computer Society, 2010, pp. 845–850.

[53] Y. Sun, J. Tang, J. Han, M. Gupta, and B. Zhao, "Community evolution detection in dynamic heterogeneous information networks," in *Proceedings of the Eighth Workshop on Mining and Learning with Graphs*. ACM, 2010, pp. 137–146.

[54] Y. Sun and J. Han, "Mining heterogeneous information networks: a structural analysis approach," *Acm Sigkdd Explorations Newsletter*, vol. 14, no. 2, pp. 20–28, 2013.

[55] G. Rossetti and R. Cazabet, "Community discovery in dynamic networks: a survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 2, p. 35, 2018.

[56] L. Coppens, J. De Venter, S. Mitrovi, and J. De Weerdt, "A comparative study of community detection techniques for large evolving graphs," in *LEG@ ECML: The third International Workshop on Advances in Managing and Mining Large Evolving Graphs collocated with ECML-PKDD*. Springer, 2019.

[57] T. Khafaei, A. T. Taraghi, M. Hosseinzadeh, and A. Rezaee, "Tracing temporal communities and event prediction in dynamic social networks," *Social Network Analysis and Mining*, vol. 9, no. 1, pp. 1–11, 2019.

[58] S. Saganowski, P. Bródka, M. Koziarski, and P. Kazienko, "Analysis of group evolution prediction in complex networks," *PloS one*, vol. 14, no. 10, 2019.

[59] N. Ilhan and I. G. Ögüdücü, "Community event prediction in dynamic social networks," in *2013 12th International Conference on Machine Learning and Applications*, vol. 1. IEEE, 2013, pp. 191–196.

[60] B. Gliwa, P. Bródka, A. Zygmunt, S. Saganowski, P. Kazienko, and J. Koźlak, "Different approaches to community evolution prediction in blogosphere," in *2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)*. IEEE, 2013, pp. 1291–1298.

[61] J. B. Lee, R. A. Rossi, S. Kim, N. K. Ahmed, and E. Koh, "Attention models in graphs: A survey," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 13, no. 6, pp. 1–25, 2019.

[62] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, 2020.

[63] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[64] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[65] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[66] D. Q. Nguyen, T. D. Nguyen, and D. Phung, "Universal self-attention network for graph classification," *arXiv preprint arXiv:1909.11855*, 2019.

[67] M. Zhang and Y. Chen, "Link prediction based on graph neural networks," *arXiv preprint arXiv:1802.09691*, 2018.

[68] C. Wu, G. Nikolentzos, and M. Vazirgiannis, "Evonet: A neural network for predicting the evolution of dynamic graphs," in *International Conference on Artificial Neural Networks*. Springer, 2020, pp. 594–606.

[69] R. Liao, Y. Li, Y. Song, S. Wang, C. Nash, W. L. Hamilton, D. Duvenaud, R. Urtasun, and R. S. Zemel, "Efficient graph generation with graph recurrent attention networks," *arXiv preprint arXiv:1910.00760*, 2019.

[70] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 793–803.

[71] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[72] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[73] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," *arXiv preprint arXiv:1606.09375*, 2016.

[74] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1263–1272.

[75] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *The World Wide Web Conference*, 2019, pp. 2022–2032.

[76] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[77] D. Jin, X. Wang, M. Liu, J. Wei, W. Lu, and F. Fogelman-Soulié, "Identification of generalized semantic communities in large social networks," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2966–2979, 2020.

[78] S. Günnemann, B. Boden, I. Färber, and T. Seidl, "Efficient mining of combined subspace and subgraph clusters in graphs with feature vectors," in *Advances in Knowledge Discovery and Data Mining*. Springer, 2013, pp. 261–275.

[79] F. Moser, R. Colak, A. Rafiey, and M. Ester, "Mining cohesive patterns from graphs with feature vectors." in *Proceedings of the SIAM International Conference on Data Mining*, vol. 9. SIAM, 2009, pp. 593–604.

[80] A. De Salve, B. Guidi, and A. Michienzi, "Exploiting homophily to characterize communities in online social networks," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 8, p. e5929, 2021.

[81] Y. Liu, A. Niculescu-Mizil, and W. Gryc, "Topic-link LDA: Joint models of topic and author community," in *Proceedings of the International Conference on Machine Learning*. ACM, 2009, pp. 665–672.

[82] O. Shchur and S. Günnemann, "Overlapping community detection with graph neural networks," *arXiv preprint arXiv:1909.12201*, 2019.

[83] J. You, R. Ying, and J. Leskovec, "Position-aware graph neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7134–7143.

[84] J. Sun, W. Zheng, Q. Zhang, and Z. Xu, "Graph neural network encoding for community detection in attribute networks," *IEEE Transactions on Cybernetics*, 2021.

[85] S. Pool, F. Bonchi, and M. v. Leeuwen, "Description-driven community detection," *ACM Transactions on Intelligent Systems and Technology*, vol. 5, no. 2, pp. 28:1–28:28, 2014. [Online]. Available: http://doi.acm.org/10.1145/2517088

[86] V. Fionda and G. Pirro, "Community deception or: How to stop fearing community detection algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 4, pp. 660–673, 2017.

[87] S. Eswar, R. Kannan, R. Vuduc, and H. Park, "Orca: Outlier detection and robust clustering for attributed graphs," *Journal of Global Optimization*, pp. 1–23, 2021.

[88] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing, "Mixed membership stochastic blockmodels," *Journal of Machine Learning Research*, vol. 9, pp. 1981–2014, 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1390681.1442798

[89] R. Balasubramanyan and W. W. Cohen, "Block-LDA: Jointly modeling entity-annotated text and entity-entity links," in *Proceedings of the SIAM International Conference on Data Mining*, vol. 11. SIAM, 2011, pp. 450–461.

[90] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[91] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-6, no. 6, pp. 721–741, 1984.

[92] S. Wasserman, K. Faust *et al.*, *Social network analysis: Methods and applications*. New York: Cambridge university press, 1994.

[93] E. G. Tajeuna, M. Bouguessa, and S. Wang, "Tracking the evolution of community structures in time-evolving social networks," in *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 2015, pp. 1–10.

[94] P. Bródka, S. Saganowski, and P. Kazienko, "Tracking group evolution in social networks," in *International Conference on Social Informatics*. Springer, 2011, pp. 316–319.

[95] ——, "Ged: the method for group evolution discovery in social networks," *Social Network Analysis and Mining*, vol. 3, no. 1, pp. 1–14, 2013.

[96] M. E. G. Pavlopoulou, G. Tzortzis, D. Vogiatzis, and G. Paliouras, "Predicting the evolution of communities in social networks using structural and temporal features," in *2017 12th International Workshop on Semantic and Social Media Adaptation and Personalization (SMAP)*. IEEE, 2017, pp. 40–45.

[97] N. İlhan and Ş. G. Öğüdücü, "Feature identification for predicting community evolution in dynamic social networks," *Engineering Applications of Artificial Intelligence*, vol. 55, pp. 202–218, 2016.

[98] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: Extraction and mining of academic social networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008, pp. 990–998.

[99] J. Yang, J. McAuley, and J. Leskovec, "Community detection in networks with node attributes," in *IEEE 13th International Conference on Data Mining*. IEEE, 2013, pp. 1151–1156.

[100] S. Günnemann, B. Boden, I. Färber, and T. Seidl, "Efficient mining of combined subspace and subgraph clusters in graphs with feature vectors," in *Advances in Knowledge Discovery and Data Mining*. Springer, 2013, pp. 261–275.

[101] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann, "Link communities reveal multiscale complexity in networks," *Nature*, vol. 466, no. 7307, pp. 761–764, 2010.

[102] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman *et al.*, "Scratch: Programming for all," *Communications of the ACM*, vol. 52, no. 11, pp. 60–67, 2009.

[103] J. Yang and J. Leskovec, "Overlapping community detection at scale: A nonnegative matrix factorization approach," in *Proceedings of the ACM International Conference on Web Search and Data Mining*. New York, New York, USA: ACM, 2013, pp. 587–596.

[104] F. Moser, R. Colak, A. Rafiey, and M. Ester, "Mining cohesive patterns from graphs with feature vectors." in *Proceedings of the SIAM International Conference on Data Mining*, vol. 9. SIAM, 2009, pp. 593–604.

[105] X.-X. Zhan, A. Hanjalic, and H. Wang, "Information diffusion backbones in temporal networks," *Scientific reports*, vol. 9, no. 1, pp. 1–12, 2019.

[106] H. Navarro, G. Miritello, A. Canales, and E. Moro, "Temporal patterns behind the strength of persistent ties," *EPJ Data Science*, vol. 6, pp. 1–19, 2017.

[107] R. Rossi and J. Neville, "Modeling the evolution of discussion topics and communication to improve relational classification," in *Proceedings of the First Workshop on Social Media Analytics*. ACM, 2010, pp. 89–97.

[108] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *Proceedings of the 2nd ACM workshop on Online social networks*. ACM, 2009, pp. 37–42.

[109] N. K. Ahmed, R. Rossi, J. B. Lee, T. L. Willke, R. Zhou, X. Kong, and H. Eldardiry, "Learning role-based graph embeddings," *arXiv preprint arXiv:1802.02896*, 2018.

[110] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks," *nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[111] A. Barrat, M. Barthelemy, R. Pastor-Satorras, and A. Vespignani, "The architecture of complex weighted networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 11, pp. 3747–3752, 2004.

[112] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in neural information processing systems*, 2001, pp. 556–562.

[113] C. Boutsidis and E. Gallopoulos, "Svd based initialization: A head start for non-negative matrix factorization," *Pattern Recognition*, vol. 41, no. 4, pp. 1350–1362, 2008.

[114] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[115] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, no. 5, pp. 465–471, 1978.

[116] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Selected Papers of Hirotugu Akaike*. Springer, 1998, pp. 199–213.

[117] A. B. Owen and P. O. Perry, "Bi-cross-validation of the svd and the nonnegative matrix factorization," *The annals of applied statistics*, pp. 564–594, 2009.

[118] C. L. Lawson and R. J. Hanson, *Solving least squares problems*. SIAM, 1974, vol. 161.

[119] B. Gelman, M. Revelle, C. Domeniconi, A. Johri, and K. Veeramachaneni, "Acting the same differently: A cross-course comparison of user behavior in moocs." *International Educational Data Mining Society*, 2016.

[120] S. Yang, C. Domeniconi, M. Revelle, M. Sweeney, B. U. Gelman, C. Beckley, and A. Johri, "Uncovering trajectories of informal learning in large online communities of creators," in *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, 2015, pp. 131–140.

[121] M. Ouellet, M. Bouchard, and Y. Charette, "One gang dies, another gains? the network dynamics of criminal group persistence," *Criminology*, vol. 57, no. 1, pp. 5–33, 2019.

[122] T. S. Evans, "Clique graphs and overlapping communities," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2010, no. 12, p. P12037, 2010.

[123] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[124] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[125] A. Geller, C. Cioffi-Revilla, M. M. Latek, S. M. Mussavi Rizi, A. Berea, J. F. Harrison, M. Revelle, and H. Osman, "Forecasting irregular warfare via agent-based network models," George Mason University, Fairfax, VA, USA, Tech. Rep., 2011.

[126] A. Badawy, E. Ferrara, and K. Lerman, "Analyzing the digital traces of political manipulation: The 2016 russian interference twitter campaign," in *2018 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM)*. IEEE, 2018, pp. 258–265.

Curriculum Vitae

Matthew Revelle received his Bachelor of Science in Computer Science from Mary Washington College in 2003, and his Master of Science in Computer Science from George Mason University in 2009. He entered the PhD program in the Department of Computer Science at George Mason University in 2010.