LEARNING DYNAMIC NETWORK MODELS FOR COMPLEX SOCIAL SYSTEMS

by

ALIREZA HAJIBAGHERI
B.S. Shiraz University, 2009
M.S. Shiraz University, 2012

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Department of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2017

Major Professor: Gita Sukthankar

ABSTRACT

Human societies are inherently complex and highly dynamic, resulting in rapidly changing social networks, containing multiple types of dyadic interactions. Analyzing these time-varying multiplex networks with approaches developed for static, single layer networks often produces poor results. To address this problem, our approach is to explicitly learn the dynamics of these complex networks. This dissertation focuses on five problems: 1) learning link formation rates; 2) predicting changes in community membership; 3) using time series to predict changes in network structure; 4) modeling coevolution patterns across network layers and 5) extracting information from negative layers of a multiplex network.

To study these problems, we created a rich dataset extracted from observing social interactions in the massively multiplayer online game Travian. Most online social media platforms are optimized to support a limited range of social interactions, primarily focusing on communication and information sharing. In contrast, relations in massively-multiplayer online games (MMOGs) are often formed during the course of gameplay and evolve as the game progresses. To analyze the players' behavior, we constructed multiplex networks with link types for raid, communication, and trading.

The contributions of this dissertation include 1) extensive experiments on the dynamics of networks formed from diverse social processes; 2) new game theoretic models for community detection in dynamic networks; 3) supervised and unsupervised methods for link prediction in multiplex coevolving networks for both positive and negative links. We demonstrate that our holistic approach for modeling network dynamics in coevolving, multiplex networks outperforms factored methods that separately consider temporal and cross-layer patterns.

To my mother for always being there for me

ACKNOWLEDGMENTS

First, I would like to thank my parents for always being there for me and their unconditional support from the very first day. I would like to thank my advisor, Gita Sukthankar, for all of her kindness, guidance and persistence throughout the process of completing my PhD research. She is all a PhD student could ever ask for. I had the honor of close collaboration with Dr Kiran Lakkaraju, senior member of technical staff at Sandia National Labs. I would like to thank him for his valuable feedback and helping me to have a better understanding of research in general. I would also like to thank my committee members Damla Turgut, Mainak Chatterjee for their guidance and support. Last but not least, I want to thank my teammates in Intelligent Agents Lab (IAL), specifically Hamidreza Alvari, Rahmatollah Beheshti, Xi Wang, Erfan Davami, Awrad Mohammed Ali, Astrid Jackson and Saif Mohammed who helped me on various occasions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

The natural flux of people's changing social ties and interests generates a dynamic social network. This invisible network can be observed by capturing daily or weekly snapshots of user activities on social media platforms and massively multiplayer online games (MMOGs), allowing these environments to to serve as "laboratories" for studying large-scale human behaviors. It is informative to visualize this data as a set of graphs for each time period, where vertices correspond to users and edges represent interactions. Multiple types of dyadic associations can be represented by encoding the data as a multiplex network, where the links at each layer represent a different type of interaction between the same set of nodes. Often these network layers coevolve, due to interdependencies between the social processes represented by different layers.

1.1 Datasets for Analyzing Large Scale Human Behaviors

The main goal in this dissertation is to study large-scale human behaviors in coevolving multiplex networks (Figure 1.1). Due to the lack of good standardized datasets, there has been relatively little research on dynamic multiplex networks, as compared to static single layer ones [16–18, 42, 44]. One of the contributions of this dissertation is the creation of a dynamic multiplex network dataset from a massively multiplayer online game called Travian, which is potentially useful for studying a variety of social phenomena. Most online social media platforms are optimized to support a limited range of social interactions, primarily focusing on communication and information sharing. In contrast, relations in massively-multiplayer online games (MMOGs) are often formed during the course of gameplay and evolve as the game progresses [114]. Even though these relationships are conducted in a virtual world, they are cognitively comparable to real-world friendships or co-worker relationships [150]. The amount and richness of social intercourse makes it possible to observe a broader gamut of human experiences within MMOGs such

as World of Warcraft [134], Sony EverQuest II [70, 114], and Travian [73, 144] than can be done with other data sources.

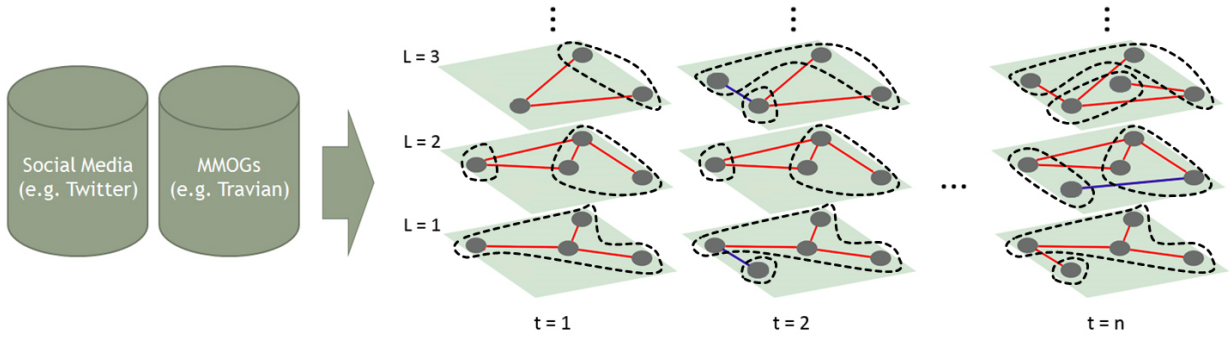


Figure 1.1: A dynamic multiplex network with changing community structure. Each person is associated with a node in a network. The network structure is *dynamic* and changes over time as new users join and social connections are formed. Adding *multiple layers* to the network allows a variety of interactions to be represented within the same network. Users self-organize into *communities* based on shared interests that also change over time.

1.2 Predicting Community Changes in Dynamic Networks

It is informative to study changes in the network at the community level, as well as the individual level. Communities are emergent groups that are created as people form highly connected subnetworks with their families, co-workers, and friends. Often communities are formed by participants with the same goals, interests, or a geographic location. For instance, in MMOGs, network communities may emerge from guilds of players with common economic interests or alliances who share strategic goals. As the network changes, user groups can grow, shrink, or disappear, causing drastic changes in the total number of network communities.

Community detection can help us understand the hidden social structure of the user populations, but the dynamic aspect of networks can pose problems for standard algorithms. The method proposed in this dissertation, D-GT (Dynamic Game Theoretic community detection),

uses stochastic optimization to find the best community structure, assuming that the nodes are modeled as rational players who seek to maximize their personal utility while playing a community membership game with neighboring nodes. In this game, the active agent decides to join or leave different communities; agents receive benefits from being part of the same community as their network neighbors but are penalized for joining too many communities. The Nash equilibrium of the current game corresponds to the community structure of the current snapshot. As the network evolves, agents usually find it advantageous to modify their community membership strategy. In this dissertation, we examine the performance of varying the amount of information propagated from prior snapshots. Even in dynamic networks, there are many nodes that retain the same community membership or rejoin their former communities. Thus propagating information from previous snapshots can provide more favorable initialization conditions for the stochastic optimization procedure.

1.3 Modeling Network Dynamics

Our next contribution is a link prediction model, RPM (Rate Prediction Model). The aim of link prediction is to forecast connections that are most likely to occur in the future, based on examples of previously observed links. A key insight is that it is useful to explicitly model *network dynamics*, how frequently links are created or destroyed when doing link prediction. Not only do different networks change at different rates, but individuals within a network can have disparate tempos of social interaction. Modeling this aspect of network dynamics can ameliorate performance on link prediction tasks.

Link prediction approaches commonly rely on measuring topological similarity between unconnected nodes [5,54,138]. It is a task well suited for supervised binary classification since it is easy to create a labeled dataset of node pairs; however, the datasets tend to be extremely unbalanced with a preponderance of negative examples where links were not formed. Topological metrics are

used to score node pairs at time t in order to predict whether a link will occur at a later time $t'(t' > t)$. However, even though these metrics are good indicators of future network connections, they are less accurate at predicting *when* the changes will occur (the exact value of t'). To overcome this limitation, we explicitly learn link formation rates for all nodes in the network; first, a time series is constructed for each node pair from historic data and then a forecasting model is applied to predict future values. The output of the forecasting model is used to augment topological similarity metrics within a supervised link prediction framework. Prior work has demonstrated the general utility of modeling time for link prediction (e.g., [21, 67, 108]); our results show that our specific method of rate modeling outperforms the use of other types of time series.

1.4 Cross-Layer Link Prediction

Networks extracted from social media platforms frequently include multiple types of links that dynamically change over time; these links can be used to represent dyadic interactions such as economic transactions, communications, and shared activities. Organizing this data into a dynamic multiplex network, where each layer is composed of a single edge type linking the same underlying vertices, can reveal interesting cross layer interaction patterns. Rather than organizing this data into social networks separately chronicling the history of different forms of user interaction, dynamic multiplex networks [71] offer a richer formalism for modeling the social fabric of online societies. A multiplex network is a multilayer network that shares the same set of vertices across all layers. This network can be modeled as a graph $G =< V, E >$ where V is the set of vertices and E is the set of edges present in the graph. The dynamic graph $G = \{G_0, G_1, ..., G_t\}$ represents the state of the network at different times. The network is then defined as: $G_t =< V, E_t^1, ..., E_t^M >$ with $E_t^\alpha \subseteq V \times V, \forall \alpha \in \{1, ..., M\}$, where each set E_t^α corresponds to the edge set of a distinct layer at time t. Thus a dynamic multiplex network is well suited for representing diverse user activities over a period of time. In coevolving networks, links in one layer result in an increased

probability of other types of links forming between the same node pair. Hence we believe that a holistic approach in which all the layers are simultaneously considered can outperform a factored approach in which link prediction is performed separately in each layer.

1.5 Signed Link Prediction in Multiplex Networks

While both positive and negative relationships clearly exist in many social network settings, the vast majority of research has only considered positive relationships. On social media platforms, people form links to indicate friendship, trust, or approval, but they also link to signify disapproval of opinions or products. In online games it is even possible to directly attack other players' avatars or armies. The recent availability of signed networks in social media sites such as Epinions and Slashdot or online games such as Travian and EverQuest has motivated more research on signed network analysis [32,74,82]. This recent work has shown that negative links have significant added value over positive links for various analytical tasks. For example, a small number of negative links improves the performance of recommender systems in social media [90, 137]. Similarly, trust and distrust relations in Epinions can help users find high-quality and reliable reviews [58].

In this dissertation, we address the problem of predicting future user interactions from other layers of the network where a layer could represent either negative or positive user interactions. To do this, it is crucial to determine the correlation between layers. Two network layers of opposite valence are likely to have negatively correlated link formation processes. To capture the interdependencies between different layers, we use mutual information to determine the sign of the correlation. Mutual information expresses the reduction in uncertainty due to another variable; we demonstrate that the average value of a layer's mutual information can be used to calculate the correlation of link formation processes across network layers with unknown valences.

Link prediction algorithms [63, 85, 93, 117] have been implemented for many types of online social networks, including massively multiplayer online games and location-based social

networks. Despite the fact that link prediction is a well studied problem, few link prediction techniques specifically address the problem of simultaneously predicting positive links across multiple networks [39, 66, 112, 133], and there is little literature available on predicting negative links in such networks.

CHAPTER 2: LITERATURE REVIEW

Network structure analysis has become an increasingly important aspect of understanding user behavior on social media platforms. In the last decade, much work has been done in investigating and characterizing the dynamics of complex systems using massive network data from diverse domains. This process poses relations and links among entities, or people, at the center of investigation. Network analysis techniques have been applied to social networks, technology networks, the WWW, and biological networks, and extensive research has been devoted to the extraction of non trivial knowledge from such networks. Predicting future links among the actors of a network [25, 85], detecting and studying the structure of communities [7] and mining frequent patterns of users behaviors [19, 36], are only a few examples of problems studied in *complex network analysis*.

To capture the global properties of these networks and solve a variety of related problems, the first approach is to model them as graphs whose nodes represent the dynamical actors, and whose links stand for the interactions/relationships between them. These graphs might be static or dynamic on one hand and single layer or multiplex on the other. Many studies on online social networks, WWW, and biological networks focused on the macroscopic properties of static networks (e.g., degree distributions, diameter, clustering coefficient); work in this area includes [6, 26, 50, 125]. However, social networks are not static. They are dynamic structures that evolve over time either by the addition of new vertices or nodes or by new links that form between nodes. Thus, the study and modeling of the dynamics in the network structure are important and the focus of a number of papers [12, 14, 81].

Most of the networks studied are single layer with only one link between nodes. Network analytics on these graphs has focused on the measurement of local and global properties such as diameter, degree distribution, centrality, and connectivity. However, in the real world, networks are often multiplex, i.e there might be a variety of relationships between any pair of nodes and

aggregating the different types might discard important information about the structure and function of the original system [27, 75]. For instance, the same set of individuals in a social system can be connected through friendship, collaboration, communication and co-location relationships, or in massively multiplayer online games (MMOGs), players might have variety of connections with each other such as trade, message and attack. In these systems, each type of interaction may have a different semantic meaning, relevance, importance, and cost, so that treating all the links as being equivalent loses important information. A better description of such systems is in terms of multiplex networks, i.e. networks where each node appears in a set of different layers, and each layer describes all the edges of a given type.

Recently, a considerable amount of effort has been devoted to the characterization and modeling of multiplex networks, with the aim of creating a consistent mathematical framework to study, understand and reproduce the structure of these systems. A number of measures have been proposed in the context of real-world multiplex networks such as air transportation systems [29] and massive multiplayer online games [128]. Some other works use a statistical mechanics formulation of multiplex networks [22]. Others simply extend classical network metrics to handle multiple layers [47, 124] and to model the growth of systems of this kind [101]. Finally, another active research direction is that of characterizing the dynamics and the emergent properties of multilayer systems, especially with respect to epidemic [116] and information spreading [28, 94], cooperation [57], diffusion processes [56] and random walks on multiplex networks [46]. A review of recent papers in this field can be found in [72].

2.1 Community Detection

The problem of community detection in static networks has appeared in multiple disciplines including sociology and computer science. This has yielded a diverse set of approaches ranging from traditional network structure based algorithms [55, 96, 99], optimization techniques [8, 31],

label propagation [111, 146, 147], propinquity [153] and information diffusion [59, 60]. Detecting community structure in dynamic networks, on the other hand, has attracted less research attention due to the complexity of the problem and dearth of good datasets. There are some community detection algorithms originally designed for static networks that continue to perform well in dynamic datasets. For instance, Lancichinetti et al.'s OSLOM (Order Statistics Local Optimization Method) works on single snapshots but also benefits from information from previous network partitions. Like D-GT, OSLOM's optimization procedure can be initialized with the partition from the previous snapshot; it aims to optimize cluster significance with respect to a global null model [78]. We use this method as one of our benchmarks, along with two other static community detection algorithms, Louvain [23] and InfoMap [113]. These algorithms perform well on many static community detection problems and have the benefit of being fast to compute on a single network snapshot.

Other network properties have also been used to perform dynamic community detection; for instance Hui et al. [68] proposed a distributed method for community detection in which modularity was used as a measure instead of the objective function. QCA (Quick Community Adaptation) is a modularity-based approach that focuses explicitly on the changes in the network structure, rather than recomputing community structure from scratch at each time step [100]. Here, we evaluate the use of personal modularity as an alternative gain function to neighborhood similarity.

Some studies have focused on studying the evolution of communities over time. For instance, [65] identified subsets of nodes, "natural communities", that were stable to small perturbations of the input data. Communities detected in later snapshots were matched to earlier snapshots using the natural community tree structure. Palla et al. [104] proposed an innovative method for detecting communities in dynamic networks based on the k-clique percolation technique; in their approach, communities are defined as adjacent k-cliques, that share $k - 1$ nodes.

Machine learning has also been employed to model changes in community structure; for instance, [130] predict transitions in community structure by learning supervised machine learning

classifiers. This requires data on past transitions to train the classifiers, which limits its applicability to certain datasets. [126] adopt a data mining approach to detect clusters on time-evolving graphs; community discovery and change detection are performed using the minimum description length (MDL) paradigm.

Rather than independently detecting communities at each snapshot and matching them, another option is to make a local decision to add nodes to existing communities when new edges appear in the network. One of our benchmarks, iLCD (intrinsic Longitudinal Community Detection) [30], updates the community structure of the network based on time-stamped sets of edges. Nodes are added to communities if its mean number of second neighbors and robust second neighbors exceeds the current average for the community. However, this model is limited to certain types of network changes, and cannot handle interconnected pairs of nodes being simultaneously added or the removal of edges.

Optimization can be used to identify minimum cost community assignments in dynamic graphs. FacetNet [87] is a framework for analyzing communities in dynamic networks based on an optimization of snapshot costs. It is guaranteed to converge to a local optimal solution; however, its convergence speed is slow, and it needs to be initialized with the number of communities which is usually unknown in practice. [51] modeled dynamic community detection as a multi-objective optimization problem. Their approach is parameter free and uses evolutionary clustering to optimize a dual objective function. The first objective selects for highly modular structures at the current time step, and the second minimizes the differences between community structures in the current and previous time steps. D-GT also uses a stochastic optimization procedure, but all of the agents individually optimize their utilities based on local network information.

The Markov Cluster Algorithm (MCL) [136] identifies graph clusters by computing the probabilities of random walks through the graph; flow simulations are performed by alternating expansion (matrix squaring) with inflation operations (Hadamard powers). Due to its mathematical simplicity, it is popular for community detection in many domains but is slow and often overesti-

mates the number of communities in the dataset. Regularized-MCL [115] is a variant of MCL that prevents overfitting by taking into account neighbor flows. It can also be used within a multi-level framework (Multi-level Regularized MCL) to speed up computation by executing a sequence of coarsening operations on the graph before executing R-MCL.

LabelRankT [145], shares some similarities with the MCL techniques described above while improving upon them in several ways; it is a label propagation approach in which the inflation operation is applied to the label distribution matrix rather than to the adjacency matrix. Each node requires only local information during propagation making it more scalable than MCL and amenable to parallelization. Due to LabelRankT's strong performance and good implementation, it was selected as the best label propagation benchmark for our work.

Our proposed method (D-GT) attempts to simulate the decision-making process of the individuals creating the communities, rather than focusing on statistical correlations between labels of neighboring nodes. We believe that exploiting game theory for dynamic community detection yields more realistic, fine-grained communities since intrinsically game theory is a good representation for expressing the behavior of individuals and strategic interactions among them [3, 92]. We have demonstrated the success of game-theoretic approaches in static community detection across several domains, including detecting guilds in massively multiplayer online games [10] and predicting trust between users on e-commerce sites. Many of these domains featured overlapping communities in static networks [8,9]; however in this dissertation the datasets are dynamic, but not overlapping.

2.2 Link Prediction

Online social networks, such as Facebook, Twitter and Foursquare, have become more popular in recent years. Information entities in online social networks can be represented as nodes and the relationships between the nodes can be denoted as links. Two types of common links are: social

connections [151, 152] and location check-ins between users and locations [152]. Link prediction has extensive applications in real-world social networks and many concrete social services can be cast as link prediction tasks, e.g., friend recommendation services can be solved by predicting the social links among users [40,41,43,117,139] and location recommendation services can be treated as the location link prediction task [33, 149]. Link prediction problems in online social networks can divided into two categories: (1) link prediction problems in single layer networks [85], (2) link prediction problems in multiplex networks [33, 127, 148, 149].

2.2.1 Link Prediction Problem in Single Layer Networks

Approaches to the link prediction problem in single layer networks are commonly categorized as being *unsupervised* [67, 84, 141] or *supervised* [64, 86, 123, 140]. In unsupervised approaches, pairs of non connected nodes are initially ranked according to a chosen similarity metric (for instance, the number of common neighbors) [89, 95]. The top k ranked pairs are then assigned as the predicted links. The strength of this paradigm is that it is simple and generalizes easily to many types of data, but there are some limitations: for instance, how to *a priori* select the cutoff threshold for link assignment? Implicitly, these approaches assume that the links with the highest scores are most likely to occur and form the earliest; however this is often not the case in many dynamic networks [95]. If the rank correlation between the selected metric and the sequence of formed links is poor, the accuracy of this approach suffers.

In addition to node similarity metrics, random walk is often used as the basis of unsupervised learning algorithms, most famously PageRank [103]. Another variant - Random Walk with Restart (RWR) is also commonly used [105, 135]. Though RWR performs well in many real world problems, its unsupervised nature means that it doesn't always adapt well to new domains.

Supervised approaches have the advantage of being able to 1) simultaneously leverage multiple structural patterns and 2) accurately fit model parameters using training data. In this case, the link prediction task is treated as a classification problem, in which pairs of nodes that

are actually linked are assigned to class 1 (positive class), whereas the non-connected ones are assigned to class 0 (negative class). The standard model assumes that feature vectors encapsulating the current network structure at time t are used to predict links formed at $t + 1$; in some sense, this model is "amnesiac", ignoring the past connection history of individual nodes.

Supervised learning based link prediction algorithms were first introduced by Liben-Nowell and Kleinberg in 2003 [84], who studied the usefulness of graph topological features by testing them on bibliographic data sets. Hasan et al. extended their work in 2006 [4] by identifying a list of easily-computable features (specifically for the coauthorship domain) and compared different models (namely decision tree, kNN, multilayer perception, SVM, RBF network) on the features and data set. Cukierski et al. incorporated 94 distinct graph features in their random forest classifier and achieved reasonable results in predicting links on Flickr dataset [37]. [13] proposed a supervised random walk algorithm for link prediction and recommendation in social networks. They focus on learning weights for the edges in social network, on the basis of rich attribute information of the nodes (e.g. users of Facebook), and then the problem of classification is transferred to optimization. Although this method performs well on user recommendation in the Facebook network, it requires rich information of user node features like age, gender and hometown, and edge features like interaction activity. Additionally, their method is computationally intensive since it tries to learn all of the weights in the matrix.

2.2.2 Link Prediction Problem in Multiplex Networks

Many interesting real-world systems form complex networks with multiple distinct types of inter-related objects and relationships. Link prediction in these networks has typically been performed by treating all relationships equally or by studying each relationship separately and ignoring dependency patterns across types; both of these approaches potentially lose valuable information. Different edge types may have a different topology or link formation mechanisms but nonetheless influence each other, such that various combinations have specific relevance to the

link prediction task. In order to avoid this information loss, in this dissertation we try to use cross-layer information to enhance the performance of our link prediction models. The process of using cross-layer information for link prediction in multiplex networks can be treated as a transfer learning problem where information is learned from a source layer and applied to improve prediction performance the target layer. Tang et al. [133] introduced a transfer-based factor graph (TranFG) model which incorporates social theories into a semi supervised learning framework. This model is then used to transfer supervised information from a source network to infer social ties in the target network.

Another strategy is to create more general versions of the topological measures that capture activity patterns in multilayer networks. Davis et al. [39] introduced a probabilistically weighted extension of the Adamic/Adar measure for these networks. Weights are calculated by doing a triad census to estimate the probability of different link type combinations. The extended Adamic/Adar metric is then used, along with other unsupervised link predictors, as input for a supervised classifier. Similarly, Hristova et al. [66] extend the definition of network neighborhood by considering the union of neighbors across all layers. These multilayer features are then combined in a supervised model to do link prediction. One weakness with the above mentioned models is their inability to use temporal information accrued over many snapshots, rather than relying on a single previous snapshot. Rossetti et al. [112] combined multidimensional versions of Common Neighbors and Adamic/Adar with predictors that are able to utilize temporal information. However, like the standard version of these metrics, these extended versions do not necessarily generalize to networks generated from different processes.

Finally, in another research proposed by Pujari et al., the authors extend their previous method for single layer networks [109] to multiplex networks [110]. They compute topological attributes for each network layer and combine them using 1) a simple aggregation of these scores across all layers or 2) an entropy-aggregation of values. These combinations are then used as a series of features in a decision tree model.

2.2.3 Signed Link Prediction Problem

Recently several papers have begun to investigate negative as well as positive relationships in online contexts. For example, users on Epinions express trust or distrust of others [58], and Slashdot participants declare users to be either friends or foes [74, 77]. More generally, arbitrary hyperlinks on the Web can be used to indicate agreement or disagreement with the target of the link, though the lack of explicit labeling in this case makes it more difficult to reliably determine this sentiment [106]. The sign of a given link in a social network is defined as positive or negative based on the attitude from the source of the link to the recipient. A fundamental question is then the following: How do negative links affect the formation of those with positive sign? Can we extract information from negative relationships to predict positive relationships and vice-versa? Answers to these questions can help us reason about how negative relationships are used in online systems. When used correctly, negative link information benefits user searches. For example, the addition of a small number of negative links has been shown to improve the performance of recommender systems in social media [90, 137]. Similarly, trust and distrust relations in Epinions can help users find high-quality and reliable reviews [58].

Our aim in this dissertation is to leverage correlations between layers to improve link prediction in multiplex networks. More specifically, we present a theoretical analysis of the link prediction problem from the perspective of information theory. This process of information extraction from negative layers can be treated as measuring the average mutual information of a target layer based on a negative or positive predictor layer. Tan et al. [131] developed a framework to uncover missing edges in networks via the mutual information of network topology. They estimate mutual information parameters for a node pair based on structural features of a network. Node pairs are then sorted according to their mutual information and those with highest values are deemed the best candidates to link in the future. Here, we use mutual information simply to determine the sign of correlations between layers; the scoring process is done separately.

CHAPTER 3: MULTIPLEX NETWORK DATA ANALYSIS: AN INTRODUCTION TO TRAVIAN MMOG

Massively-multiplayer online games (MMOGs) can serve as a unique laboratory for studying large-scale human behaviors. However, one question that often arises is whether the observed behavior is specific to the game world and its winning conditions. In particular, inter-player aggression is more openly expressed within MMOGs since combat often comprises a large portion of gameplay. In this dissertation, we aim to study how conflict in MMOGs shapes the underlying social networks. In MMOGs, conflict and cooperation are inextricably linked since many attacks are launched by coalitions of players to gain resources, control territory, or subjugate enemies. For our analysis of conflict within MMOGs, we selected two browser-based games, Game X and Travian, in which there have been extensive previous studies of cooperation between players [11, 49, 73, 76, 79, 144]. The two games differ in game objectives: there is no official winning condition for Game X. In contrast, Travian players develop their civilizations over a fixed period of time in order to be the first to build a magnificent Wonder of the World, a construct that can only be erected through extensive collaboration among a team of players. Constant raiding is required to amass the resources required to grow one's civilization.

To analyze the players' behavior, we constructed multiplex networks for both games, with link types for attack, communication, and trading. Our research compares and contrasts the network structure of players in both games; while there have been other studies of multiplex networks in a single game (e.g., [79, 114]), there have been few studies that have looked for gameplay patterns across multiple games. For instance, griefing behavior was compared between World of Warcraft and Toontown, but not within the context of social network structures [143]. The overarching aim of our study is to understand the evolution of conflict in different game worlds. We discuss how differences in MMOG game objectives shape the structure of conflict. Then we study

how the attack networks differ from communication and trade networks. Finally, we analyze how communication, trade, and geographic connections affect the likelihood of two players engaging in hostilities.

In the next section, we present a short introduction to Massively Multiplayer Online Games and recent research in the area. The, two games that we used in our data analysis and create multiplex networks from are introduced. Finally, we study the nature of conflict and communication across two game worlds (Travian and Game X). We compare and contrast the structure of attack networks with trade and communication networks. Similar to real-life, social structures play a significant role in the likelihood of inter-player conflict.

3.1 Massively Multiplayer Online Games (MMOG)

Massively multiplayer online games (MMOGs) are highly graphical 2- or 3-D videogames played online, allowing individuals to interact not only with the gaming software (the designed environment of the game and the computer-controlled characters within it) but with other players as well. Most online social media platforms are optimized to support a limited range of social interactions, primarily focusing on communication and information sharing. In contrast, relations in massively-multiplayer online games are often formed during the course of gameplay and evolve as the game progresses [114]. Even though these relationships are conducted in a virtual world, they are cognitively comparable to real-world friendships or co-worker relationships [150]. The amount and richness of social intercourse makes it possible to observe a broader gamut of human experiences within MMOGs such as World of Warcraft [134], Sony EverQuest II [70, 114], and Travian [73, 144] than can be done with other data sources.

Massively-multiplayer online games have been a fertile testing ground for many types of human studies, enabling scientists to overcome key difficulties in studying social dynamics by providing an experimental platform for collecting high resolution data over longer time pe-

riod [73, 114, 134, 144]. They have been particularly valuable for studying groups [134], teams, and organizations [73], since banding together yields economic and combat advantages in most games. Geographically-separated players must work together to achieve shared goals using a similar combination of email, chat, and videoconferencing as remote employees, hence game guilds can be viewed as analogous to virtual workplace organizations [73]. Trust between guild members is positively correlated with group performance [49], and willingness to grant shared access to property, items, or user accounts can be used to measure trust between players. Roy et al. [114] demonstrated that network structures in multiplex networks are very useful for predicting trust between MMOG players. Similarly we believe that the network structure of multiplex networks is correlated with the likelihood of conflict among players.

In real-life there are myriad potential motivations for choosing to fight. Humphreys and Weinstein [69] categorized key determinants of participation in conflicts as being long-term grievances (i.e. economic or political disenfranchisement), selective incentives (money or safety), and community cohesion. Community cohesion predicts that a person is more likely to join the conflict if they are members of a tightly-knit community and their friends have already joined. This factor is the most relevant to fighting within MMOGs. Not only are there conflicts between guilds and alliances, but pick-up groups may spontaneously form to tackle larger challenges such as boss fights [20].

3.2 Introduction to Travian

Travian is a popular browser-based real-time strategy game with more than 5 million players. Games can be played in over 40 different languages on more than 300 game servers worldwide. Players start the game as chieftains of their own villages and can choose to be a member of one of three tribes (Gaul, Roman, or Teuton). Each of these three tribes has its own advantages and disadvantages. For instance, Teutons produce the cheapest military units and are the best raiders,

whereas Gauls are the best at living in peace and have fast units and merchants. Players seek to improve their production capacity and construct military units in order to expand their territory through a combination of colonization and conquest. Each game cycle lasts a fixed period (a few months) during which time the players vie to create the first civilization to complete construction on one of the Wonders of the World. In the race to dominate, actors form alliances of up to 60 members under a leader or a leadership team. Alliances are equipped with a shared forum, a chat room and an in-game messaging system. Similar to the real world, teamwork and negotiation skills play a crucial role in game success (see Figure3.1).

Each server is a closed game environment and holds approximately 25,000 users on maximum. Playing with up to 20,000 users on one server with scarce resources, actors soon find themselves in a social dilemma [45], which is typical for organizations, project teams and economies where parties need to both coordinate and compete with one another. In the race to dominate, actors form teams or alliances of up to 60 members under a leader or a leadership team. Teams are equipped with a shared forum, a chat room and an ingame messaging system. Like in virtual teams at work, teamwork and negotiation skills play a crucial role in this context. Given these characteristics of Travian, the virtual teams in Travian afford an excellent opportunity to study various facets of virtual organizations.

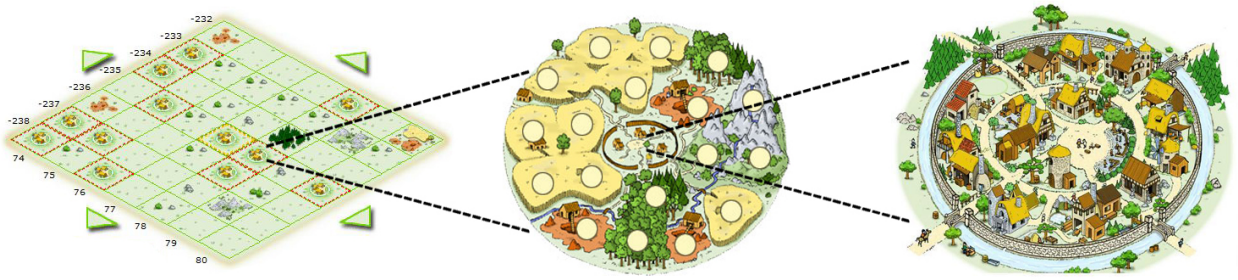


Figure 3.1: The virtual world of Travian (http://travian.us/)

Conflicts in Travian can be divided into two categories: attacks and raids. The goal of

an attack is to destroy its target, whereas raids are meant to gather bounty and are much less vicious. The armies will do battle until at least one side is reduced in strength by 50%, and therefore the loss on both sides is usually smaller. For this dissertation, we construct an attack network from the raid data. A trade is an exchange of different resources (gold, wood, clay, wheat) necessary to upgrade a village's buildings. In Travian, villages may trade their resources with other villages if both villages have a marketplace. Travian has an in-game messaging system (IGM) for player communication. IGMs can also include broadcast messages, i.e. messages sent to all players by the game moderators. To study these processes we created trade and communication networks. However, in this analysis, broadcast messages were not considered as their volume could introduce bias in the results. For our analysis, we have used data collected from a server in Germany specifically designed for research purposes. The data set contains variety of tables including logs and reports from different actions of users.

3.3 Game X

Game X is a browser-based exploration game in which players act as adventurers traveling a fictional game world in a vehicle. Similar to real-life, there is no absolute victory within the Game X world; instead players are engaged in an ongoing process of exploring the game world, mining resources, and engaging in commerce and battle with other players. Players can use resources to build factory outlets and create products that can be sold to other players. Unlike other MMOG's like World of Warcraft (WoW) and Everquest (EQ), Game X has a turn-based play system. Every day each player gets an allotment of turns. Every action (except communication) requires some number of turns to execute; example actions include: 1) move vehicle 2) mine resources 3) buy/sell resources 4) build vehicles, products, or factory outlets, 5) fight non-player characters 6) fight player characters. Players can communicate with each other through in-game personal messages, public forum posts and in chat rooms; they can also denote other players as friends or as hostiles.

In Game X fighting is one mechanism for advancement, but since fighting expends turns, the players must carefully weigh the tradeoffs between combat and pursuing an economic agenda. Players can engage in combat with non-player characters, other players, and even market centers and factory outlets. A player's skills affect the ability to attack/defend, and they can modify their vehicles to include new weaponry and defensive elements.

Game X contains four types of groups: 1) nations 2) agencies 3) races and 4) guilds. Wars in Game X can only occur between nations. There are three nations in the game, which were pre-defined by the game creators. Joining a nation provides access to restricted nation-controlled areas, quests, and vehicles. Each nation can have one of the following diplomatic relations to all others: benign, neutral, strained, or hostile. If enough members of a governing body select hostile diplomatic relations against another nation, a war is declared between the respective nations. After war has broken out, additional combat actions are available for the warring nations. In particular, war quests are available, which provide medals of valor to the players that wish to undertake and complete the quests. Any attack against the opposing nation results in accumulating a set number of war points. When the war ends, these war points determine the "winner" of the large-scale conflict. A war situation will (via the game's design) gravitate towards a state of peace. Each of the respective governing bodies must maintain a majority vote to continue the war effort. Over time, the amount of votes required to continue is increased by the game itself. Eventually, no amount of votes will suffice and the nations return to a state of peace.

Players also have a bidirectional "reputation" measure with the nations in the game. Combat with members of other nations incurs a negative penalty to this measure. During the war period this negative penalty is dropped for players of the two warring nations – allowing unrestricted combat.

3.4 Travian Dataset Analysis

The Game X dataset consists of data from 730 days, and the Travian data is composed of one game cycle on a high speed server in an expedited game (a period of 144 days). The experiments in this dissertation were conducted on a 30 day period in the middle of the Travian game cycle. This period has fewer transient bursts of activity and a more stable network than the early period (which has many less committed players who drop out) and the late period where the focus is on the Wonder of the World construction.

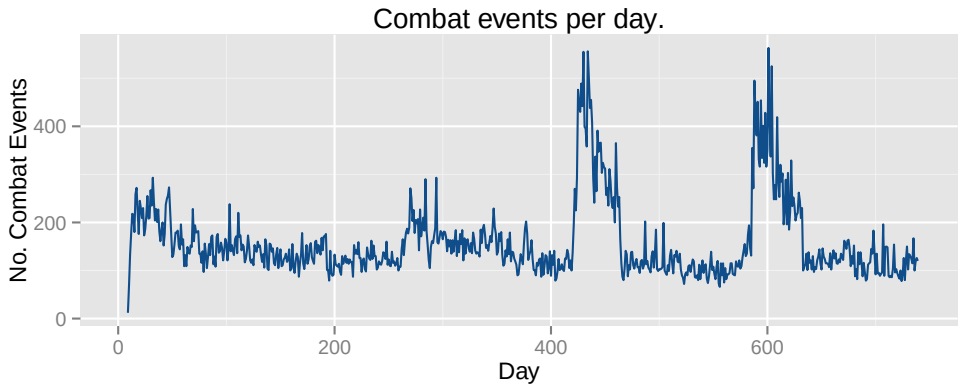


Figure 3.2: Frequency of attacks in the Game X dataset (730 days). The peaks in activity represent the first and second declaration of war between nations.

Table 3.1: Travian and Game X attack, message, and trade network statistics.

	Travian			Game X		
Parameter/Network	**Attack**	**Message**	**Trade**	**Attack**	**Message**	**Trade**
# of Vertices	4,418	3,092	2,649	2,898	5,112	5,860
Frequency	633,105	451,669	271,039	14,270	907,868	389,629
Diameter	17	9	10	14	8	10
Avg. Path Length	5.312	3.471	2.849	4.476	3.402	4.218
Avg. Degree	7.998	14.591	32.828	8.375	27.645	25.01
Avg. Clustering Coefficient	0.065	0.319	0.154	0.012	0.137	0.117

To analyze the relationship between conflict and communication in Game X and Travian, we identified a period of the game with comparable attack network statistics. First, for every day of the two games, we created a vector of features (the network properties from Table 3.1). Then, using the standard Euclidean distance measure, we find the most similar day from the Game X data for every Travian day. The features included: 1) nodes, 2) edges, 3) average path length, 4) diameter, 5) local transitivity, and 6) global transitivity. Results of this comparison reveal that the days found using this matching algorithm tend to fall in a 30 day period ([590,620]) within the second major Game X war (Figure 3.2). Table 3.1 shows the statistics for attack, trade, and message networks during the time period selected for this analysis.[1] In these networks, each node represents an individual player, and directed edges represent attacks, trades, and messages between players. Attack graphs in both Travian and Game X have a higher diameter, lower average degree, and lower clustering coefficient than either the message or trade graphs.

[1]This dataset has been made available at: http://ial.eecs.ucf.edu/travian.php

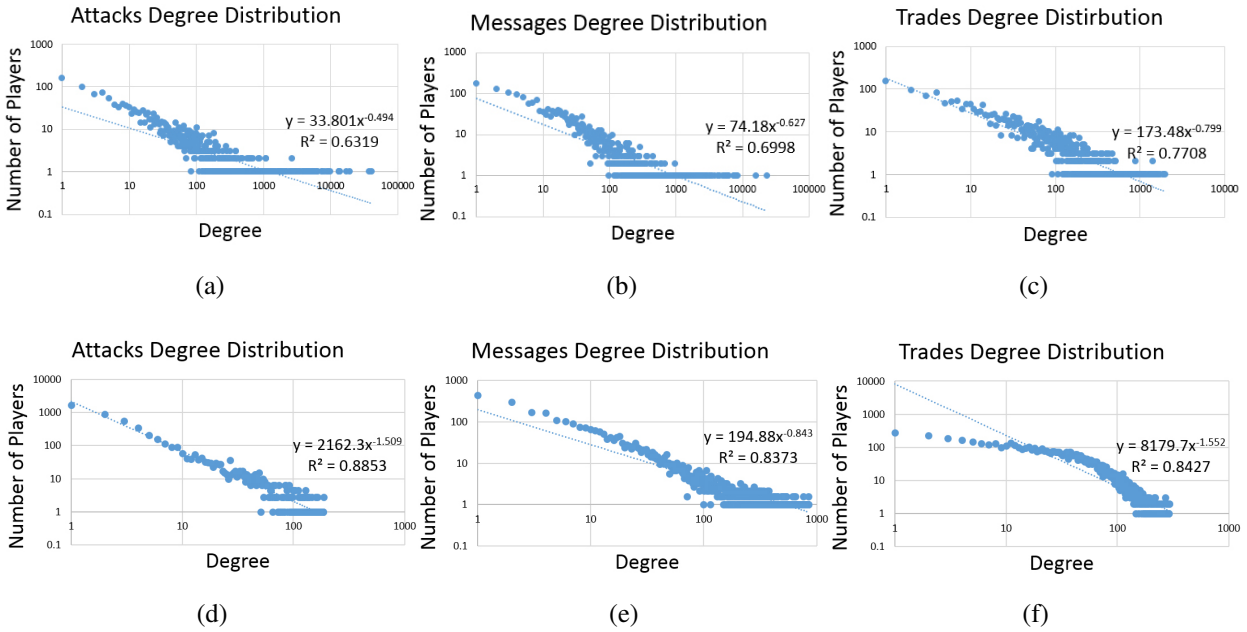


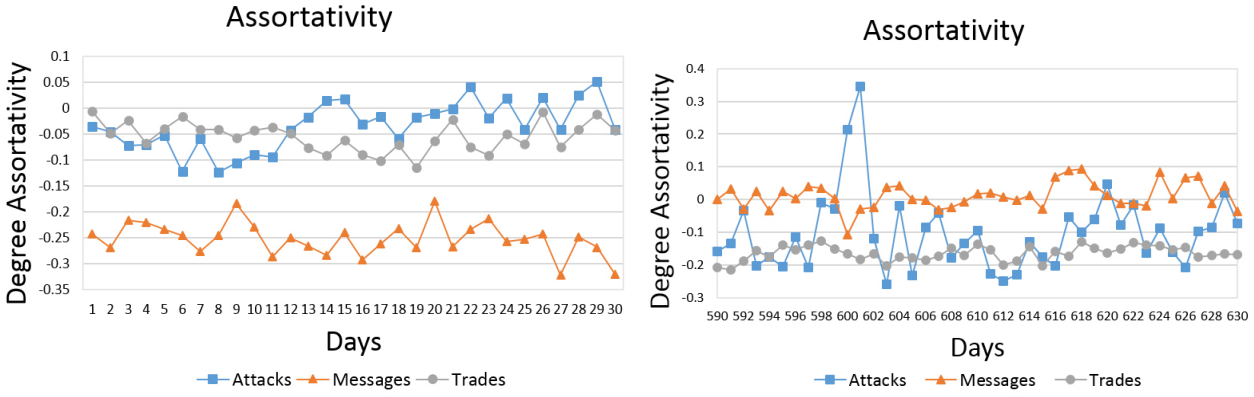
Figure 3.3: Degree distribution (log-log scale) for (a) attack, (b) message, and (c) trade networks from Travian (top) and Game X (bottom)

Although Travian and Game X possess many commonalities, an important difference between the two games is in the role of combat in gameplay. In Travian, attacking (raiding) is one of the easiest pathways for gaining the necessary resources for growing one's civilization. In Game X, attacks between players cause a loss of reputation if they are performed outside of war periods. The role of time is different in the two games. In Travian, players need to rush to grow their civilizations within a short period of time. In Game X, the players have infinite time to explore the richness of the world, but they can only take a limited number of actions per day. Each action spent fighting limits the number of actions available for economic development, hence Game X has a lower absolute frequency of attacks.

Interestingly, the degree distributions of attack, messages, and trades in both games conform to a power law distribution (Figure 3.3). Clauset et al. [34] proposed a robust estimating technique to estimate the parameters of a power law; to verify the distributions, we used this

method which employs a maximum likelihood estimator. This model calculates the goodness-of-fit between the data and the power law. If the resulting value is greater than 0.1 the power law is a plausible hypothesis for the data, otherwise it is rejected. Visually the trades in Game X appear to follow a double pareto lognormal distribution, with an exponential decay for higher trade values [120].

Assortativity is a preference for a network's nodes to attach to others that are similar in some way. Though the specific measure of similarity may vary, network theorists often examine assortativity in terms of a node's degree. Correlations between nodes of similar degree are found in the mixing patterns of many observable networks. For instance, in social networks, highly connected nodes tend to be connected with other high degree nodes. This tendency is referred to as assortative mixing, or assortativity. On the other hand, technological and biological networks typically show disassortative mixing, or dissortativity, as high degree nodes tend to attach to low degree nodes [98].



(a) (b)

Figure 3.4: Travian (a) and Game X (b) node degree assortativity

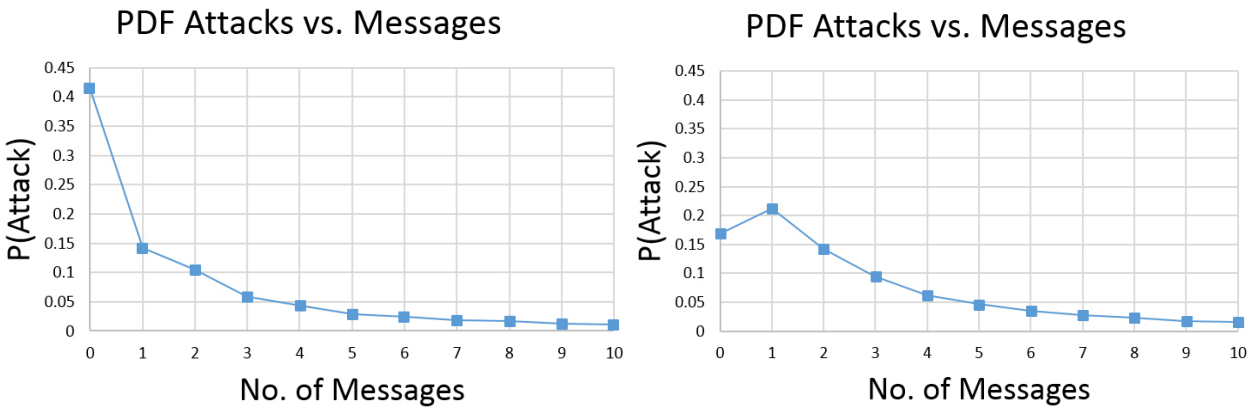
For Travian, as shown in Figure 3.4, while the message network displays disassortative mixing, attack and trade networks tend to show a non-assortative mixing. This suggests that play-

ers who send more messages are in contact with others who rarely send messages; communication in Travian often flows from alliance leaders outward to the other alliance members, reflective of a spoke-hub communication structure. In contrast, the degree of the members appears to be an unimportant consideration in dictating connectivity in the attack and trade networks. Non-assortative networks may arise either because the networks possess a balanced number of assortative and disassortative links or because a greater number of links in one direction is counterbalanced by a greater weight in the other [107].

For Game X, the attack and trade networks show a disassortative mixing while the message network displays non-assortative mixing. This indicates that attack and trade activity is centered around group leaders, whereas communication is more likely to be agnostic to node degree. Note that Game X has a more complicated group structure since players can belong to nations, guilds, and agencies.

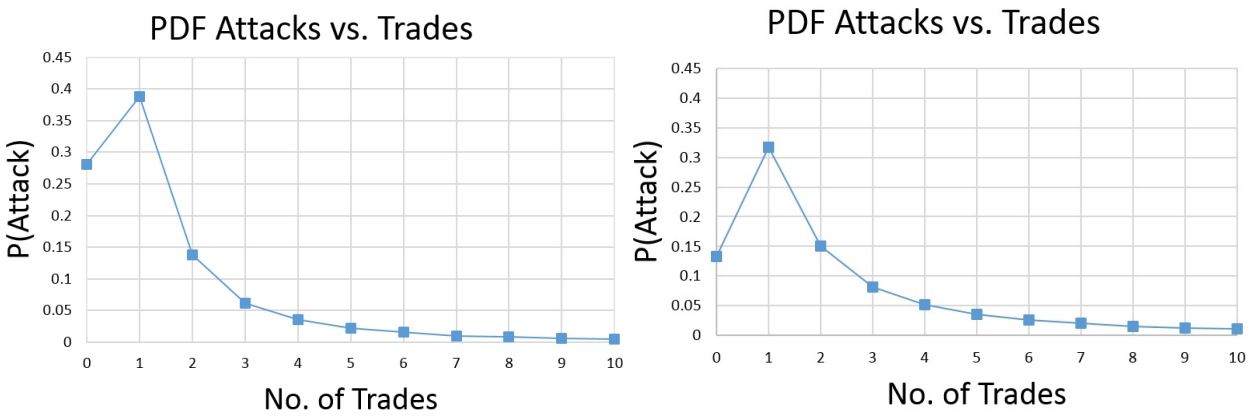
Attacks in both Travian and Game X are generally inversely proportional to other types of activity. In Travian, in 41% of cases, players do not attack other players with whom they have been in contact at least once (Figure 3.5). On the other hand in Game X, the above statement stands for only 17% of attacks. 21% of attacks occur between players who have exchanged one message.

In Travian a large number of players do not attack players with whom they have traded resources. As shown in Figure 3.6, 28% of the attacks in Travian occurred between two players without any trade history. However, this rate is surprisingly low in Game X; only 13% of attacks occur between players who lack a trade history. Once players have traded together, there is a sharp increase, followed by a slow decrease in attack frequency. Trading with other players indicates that they have desirable resources, making them worth attacking, and after only one trade, the players are unlikely to have established the sense of trust that may deter an attack. We believe that in some cases players who have never traded together or exchanged messages are geographically separated; hence they are less likely to attack each other because they are unaware of each other's existence.



(a) (b)

Figure 3.5: Probability of attacks occurring between a pair of users vs. the number of messages they have exchanged (P(Attack and Message=x)) in Travian (a) and Game X (b)



(a) (b)

Figure 3.6: Probability of attacks occurring between a pair of users vs. the number of trades they have made (P(Attack and Message=x)) in Travian (a) and Game X (b)

To test this hypothesis, we analyzed the probability of attack based on the distance between player territories in Travian (Figure 3.7). It was not possible to conduct a similar analysis in Game X, which has a spatial layout based on discrete tiles. To estimate distance, we calculated

the territory centroids by averaging the latitudes and longitudes of the villages. Then, standard Euclidean distance was used to measure the distance between each pair of players in the attack network. Our analysis shows that attacks between immediate neighbors are frequent. Attacks with close (but not immediate) neighbors are common, followed by a decay in attack activity with distance.

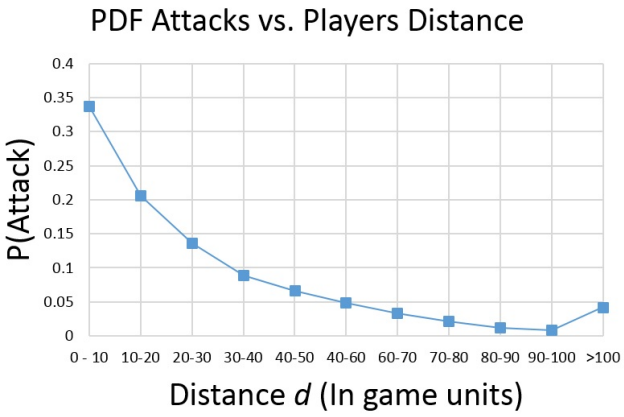


Figure 3.7: Probability of attacks based on players' distance from each other

Attacks are generally rare between alliance and guild members, indicating a strong level of trust in those relationships. In Travian, 4% of the attack edges are between two players within the same alliance. Surprisingly, the same rate stands for Game X, and only 4% of players attack their guild-mates.

3.5 Summary

This chapter summarizes our findings across two massively multiplayer games with different game objectives, Game X and Travian. Despite the fact that Travian's game structure encourages a higher level of combat activity than Game X, attack networks in both games possess a similar network structure, and are distinctly different from message and trade networks. In summary, our analysis reveals the following:

1. The attack networks in both Travian and Game X share a higher diameter, lower average degree, and lower clustering coefficient than either the message or the trade networks.
2. All networks in both games have similar power law degree distributions, but different degree assortativity. In Travian attack networks show non-assortative mixing, whereas in Game X they are disassortative.
3. The general trend is that attacks are inversely proportional to message frequency, trade frequency, and distance, with some specific exceptions. Players rarely attack fellow alliance or guild members in either game.

CHAPTER 4: IDENTIFYING COMMUNITY STRUCTURES IN DYNAMIC NETWORKS

Most real-world social networks are inherently dynamic and are composed of communities that are constantly changing in membership. As a result, recent years have witnessed increased attention toward the challenging problem of detecting evolving communities. As the network changes, user communities evolve and can grow, shrink, or disappear. Here, the word "community" refers to the emergent groups that are created as people form highly connected subnetworks with their families, co-workers, and friends. In this context, we regard the communities as simply friendship networks in which the participants share common interests, activities, or a geographic location. Intuitively we expect more edges inside the community compared to its outside, i.e. intra-connections tend to be more common than inter-connections. Community detection can help us understand the hidden social structure of the user populations, but the dynamic aspect of networks can pose problems for standard algorithms. In this dissertation, we leverage the concept of game theory to attack the problem of dynamic community detection.

4.1 Problem Formulation

Here, we provide the formal definition of the problem and the notations used throughout the dissertation. Given snapshots $\mathbf{T} = \{T^t \mid \forall t, t = 1, ..., M\}$ of a dynamic network and their corresponding underlying graphs $\mathbf{G}^t = (V^t, E^t)$, with $n^t = |V^t|$ vertices and $m^t = |E^t|$ edges, where t=1,...,M, we aim to detect community structure $\mathbf{C} = \{C^t \mid \forall t, t = 1, ..., M\}$ of the network. Table 4.1 shows the symbols and definitions used throughout this chapter.

Table 4.1: Definition of Symbols.

Symbol	Definition
T	set of snapshots
C	set of communities
$\mathbf{G}^t$	graph of t-th snapshot with no self-edges
C^t	community structure of t-th snapshot
C_k^t	k-th community in C^t
m^t, n^t	number of edges and vertices of G^t
$\mathbf{A}^t$	adjacency matrix of G^t
$\mathbf{S}^t$	profile of strategies of t-th snapshot
s_i^t	i-th agent's strategy in t-th snapshot
g_i^t	i-th agent's gain function in t-th snapshot
l_i^t	i-th agent's loss function in t-th snapshot
u_i^t	i-th agent's utility function in t-th snapshot
c_{ij}^t	similarity between i-th and j-th agents in the t-th snapshot

Table 4.2: Definition of Possible Actions.

Action	Definition
Join	add a new label to s_i^t
Leave	remove a label from s_i^t
Switch	remove a label from s_i^t and add a new one
No operation	no action is performed

4.2 Proposed Method

We leverage a dynamic agent-based model to detect communities by optimizing each user's utility through a stochastic search process [8]. In this dissertation, we evaluate four different variants of the procedure: D-GT (**D**ynamic **G**ame **T**heoretic community detection), D-GTP (**D-GT** with passing one **P**revious Snapshot) D-GTS (**D-GT** with **S**eparate Runs) and D-GTG (**D-GT** with passing **G**round Truth).

4.2.1 Dynamic Game Theory (D-GT)

In this section, we present the framework for D-GT. We treat the process of community detection as an iterative game performed in a dynamic multi-agent environment in which each node of the underlying graph is a selfish agent who decides to maximize its total utility u_i. Note that hereafter we use the terms *node*, *user*, and *agent* interchangeably. The terms *pool*, *list*, or *set* of agents are used to denote the set of nodes maintained during each game.

During the community formation game, each agent assesses whether taking an action (*join*, *switch*, or *leave*) (Table 4.2) will increase its utility. An agent **joins** a new community $c^t \subseteq C^t$ by adding its label to s_i^t. It then gains utility u_{Join}^t and its community membership changes:

$$s_i^t \leftarrow s_i^t \cup \{c^t\}. \tag{4.1}$$

It may **leave** one of its own communities, say c'^t by removing its label from s_i^t which results in utility u_{Leave}^t and changes the community membership as follows:

$$s_i^t \leftarrow s_i^t / \{c'^t\}. \tag{4.2}$$

The agent can also simultaneously switch communities:

$$s_i^t \leftarrow s_i^t / \{c'^t\}, s_i^t \leftarrow s_i^t \cup \{c^t\}. \tag{4.3}$$

Even though the **switch** action is not strictly necessary, having this additional action speeds up the convergence of the stochastic search process.

The new utility $u_i'^t$ for this agent is updated as follows:

$$u_i'^t \leftarrow \max\{u_{Join}^t, u_{Leave}^t, u_{Switch}^t, u_{noOp}^t\}. \tag{4.4}$$

To reduce computation time, only communities containing the agent's nearest network neighbors are considered as candidates for the join/switch operation, and we independently identify the best communities for the join and leave operations. If no action improves the agent's utility, it does not change its strategy (*no operation*).

The set of all communities at the t-th snapshot is denoted by C^t. We define a strategy profile $S^t = (s_1^t, s_2^t, ..., s_n^t)$ which represents the set of all strategies of all agents, where $s_i^t \subseteq C^t$ denotes the strategy of agent i, i.e. the set of its labels at snapshot t. In our framework, for each snapshot, the best response strategy of an agent i with respect to strategies S_{-i}^t of other agents is calculated as:

$$\arg\max_{s_i^t \subseteq C^t} u_i^t(S_{-i}^t, s_i^t) \tag{4.5}$$

Agents are selected randomly, without replacement, until all the agents have had the opportunity to play the community formation game in order to guarantee adequate exploration of the strategy search space. The utility function for each agent is calculated by combining the benefit of its community memberships, based on a gain function and subtracting losses incurred:

$$u_i^t(S_{-i}^t, s_i^t) = g_i^t(S_{-i}^t, s_i^t) - l_i^t(S_{-i}^t, s_i^t), \tag{4.6}$$

We have experimented with two variants on the gain function for agent i.[1] The first gain function is based on **similarity** between agents:

$$g_i^t(S^t) = \frac{1}{m^t} \sum_{k \in s_i^t} \sum_{j \in C_k^t, j \neq i} c_{ij}^t. \tag{4.7}$$

Here, we use neighborhood similarity to quantify the structural equivalence between users

[1]We employ the notation for directed graphs, although it is straightforward to generalize to undirected graphs by ignoring the *in/out* superscripts.

at time t:

$$c_{ij}^t = \begin{cases} w_{ij}^t(1 - d_i^{in,t} d_j^{out,t}/2m^t) & A_{ij}^t = 1, w_{ij}^t >= 1 \\ w_{ij}^t/n^t & A_{ij}^t = 0, w_{ij}^t >= 1 \\ d_i^{in,t} d_j^{out,t}/4m^t & A_{ij}^t = 1, w_{ij}^t = 0 \\ -d_i^{in,t} d_j^{out,t}/4m^t & A_{ij}^t = 0, w_{ij}^t = 0 \end{cases} \tag{4.8}$$

w_{ij}^t is defined as the number of common neighbors possessed by nodes i and j, where common neighbors are nodes with direct in-edges from both i and j. $d_i^{in,t}$ and $d_i^{out,t}$ are the in- and out-degrees of node i at snapshot t. m^t and n^t are the number of edges and nodes respectively and primarily serve as normalization constants. Note that c_{ij}^t assumes its highest value when two nodes have at least one common neighbor and are also directly connected, i.e. $A_{ij}^t = 1$. Hence agents playing the community formation game benefit from joining communities containing connected nodes with many common neighbors.

The second gain function measures the personalized **modularity** of the i-th agent:

$$g_i^t(S^t) = \frac{1}{2m^t} \sum_{k \in s_i^t} \sum_{j \in C_k^t, j \neq i} \sum_{k' \in s_j^t} (A_{ij}^t \delta(i,j) - \frac{d_i^{in,t} d_j^{out,t}}{2m^t} |k \cap k'|), \tag{4.9}$$

where $k \in s_i^t$ and $k' \in s_j^t$ refer to the community labels at snapshot t that agent i and j belong to respectively; $\delta(i,j)$ is an indicator function that is 1 when i and j are members of the same community. More intuitively, this gain function explains how well the i-th agent fits the communities it belongs to, compared to a randomly assigned community.

Similar to what happens in real life, we also consider the loss function l_i^t for each agent, which is linear in the number of labels each agent has, This can be used to model the intrinsic communication overhead of belonging to multiple communities and prevents the agents from indiscriminately joining every available community. Therefore we define the following loss function

for agent i:

$$l_i^t(S_{-i}^t, s_i^t) = \frac{|s_i^t|}{m^t}. \tag{4.10}$$

Here $s_i^t = \{1, 2, ..., k\}$ is the set of labels at snapshot t which agent i belongs to.

In our framework, the best response strategy of the agent i with respect to strategies S_{-i}^t of other agents is calculated by:

$$\arg \max_{s_i'^t \subseteq C^t} g_i^t(S_{-i}^t, s_i'^t) - l_i^t(S_{-i}^t, s_i'^t). \tag{4.11}$$

The strategy profile S^t forms a pure Nash equilibrium of the community formation game if no agent can unilaterally improve its own utility by changing its strategy:

$$\forall i, s_i'^t \neq s_i^t, u_i^t(S_{-i}^t, s_i'^t) \leq u_i^t(S_{-i}^t, s_i^t). \tag{4.12}$$

A local equilibrium is reached if all agents play their local optimal strategies:

$$\forall i, s_i'^t \in ls(s_i^t), u_i^t(S_{-i}^t, s_i'^t) \leq u_i^t(S_{-i}^t, s_i^t). \tag{4.13}$$

Here $ls(s_i^t)$ refers to local strategy space of agent i, which is the set of *possible* label sets it can obtain by performing the actions defined earlier.

4.2.2 Algorithm

An overview of the D-GT framework is shown in Algorithm 1. For every snapshot of the network, a set of agents, one representing each node in the graph, is created to play the community formation game. The community structure is initialized either with a set of singleton communities or with communities passed from previous snapshots. During game play, an agent is randomly

selected (without replacement) from the pool; it selects an action (join, leave, switch, or no op) by calculating the strategy that yields the highest utility. After the agent plays, the community structure is updated.

The game is played until the number of agents changing their play strategy between permutations falls below the threshold, or the maximum iteration is reached. Empirically, we have discovered that $8n$ is a good iteration limit with a threshold of 5%. Thus if there are 1000 nodes in a network snapshot, the community formation game is played until fewer than 50 nodes change strategies or to the maximum of 8000 games. Figure 4.1 shows an example of the convergence in utility vs. iteration.

The outer loop of the algorithm requires iterating over M graph snapshots, and the inner loop requires performing a maximum of $8n$ iterations of the community formation game that, in the worst case, requires considering n community choices. Thus, the overall time complexity of D-GT is $O(Mn^2)$.

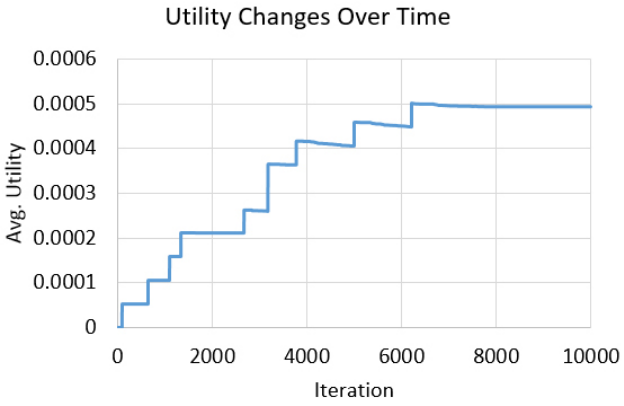


Figure 4.1: Change in average utility summed over all nodes vs. iteration for the Travian-Trades dataset (one snapshot with 964 nodes). The algorithm converges after 6680 iterations which requires 2.8 seconds to complete.

D-GT maintains a candidate set of multiple community assignments per agent until the last iteration and then selects the assignment with the highest utility function as the final disjoint

partition. We evaluate several different variants of the procedure:

- **D-GTS** (D-GT with Separate runs): this version does not employ any information from previous runs and hence is equivalent to a static community detection procedure.
- **D-GTP** (D-GT passing one Previous Snapshot): rather than passing strategy profiles from all previous snapshots, we only initialize the community formation game with the structure from a single previous time slice. For each snapshot T^k, we initialized communities and agents to the existing information from T^{k-1}.
- **D-GTG** (D-GT with passing Ground Truth): D-GTG leverages some ground truth information. A select seed group of ground truth communities with predefined size is used to initialize communities for snapshot T^k; however we do not pass any discovered community structure to the following snapshots (similar to D-GTS). This variation cannot be used unless some of the agents' community membership is known in advance.

Algorithm 1 D-GT Community Formation Game

Input: Snapshots $\mathbf{T} = \{T^1, T^2, \ldots, T^M\}$
Output: Communities $\mathbf{C} = \{C^1, C^2, \ldots, C^M\}$
for all $T^t \in \mathbf{T}$ **do**
  Initialize $p \leftarrow 0$
  repeat
    Initialize $q \leftarrow 0$
    for all agents in randomized order **do**
      Select best action a using Eqn. 4.5
      if $a =$ "No operation" **then**
        $q \leftarrow q + 1$
      else
        Update C^t according to a
      end if
    end for
    $p \leftarrow p + 1$
  until $p > 8$ or $q >$ threshold θ
end for

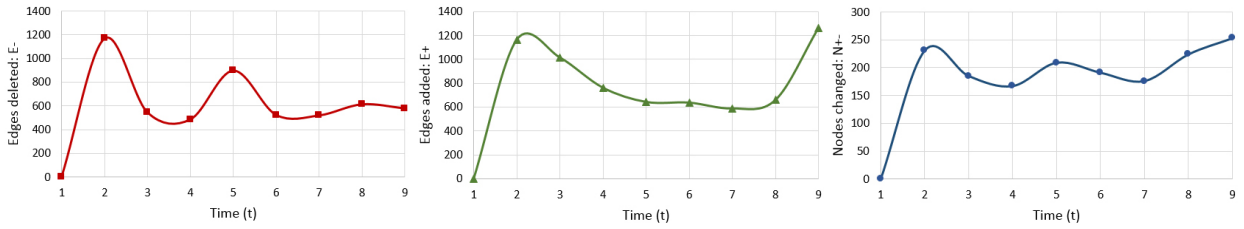


Figure 4.2: The structural changes in the AS-Oregon dataset over 9 snapshots including the number of edges deleted (E_{-}) and added (E_{+}), as well as the number of nodes involved in changes (N_{+-}). The community detection problem becomes more challenging when there are significant structural changes between snapshots.

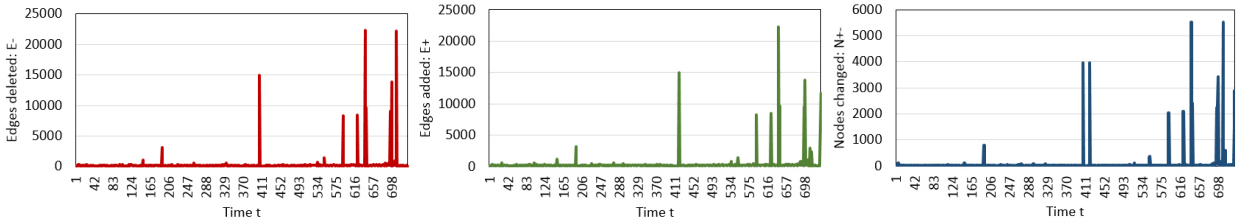


Figure 4.3: The structural changes in the AS-Internet dataset over 733 snapshots.

4.2.3 Experimental Results

Algorithms were evaluated together on a system with 12G of RAM and Intel CPU 2.53 GHz, and all reported results were averaged over ten repetitions.

4.2.3.1 Baseline Methods

We compare D-GT with the following community detection baselines:

- **LabelRankT**[2] **[145].** LabelRankT functions according to the generalized LabelRank, in which each node requires only local information during label propagation processing. Several parameters must be set before running the algorithm on the data; we used the best performing values reported in the original paper.

[2]https://sites.google.com/site/communitydetectionslpa

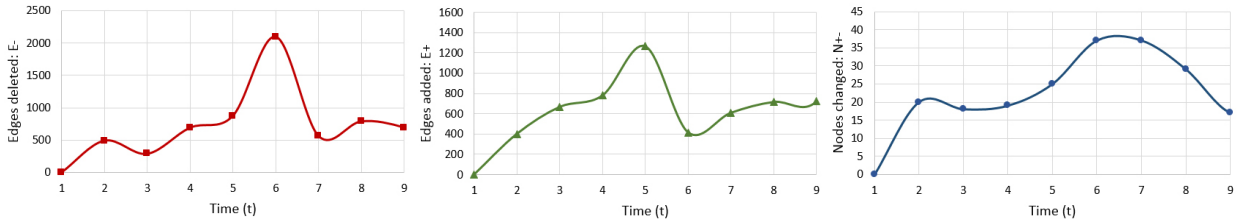


Figure 4.4: The structural changes in the Enron dataset over 12 snapshots.

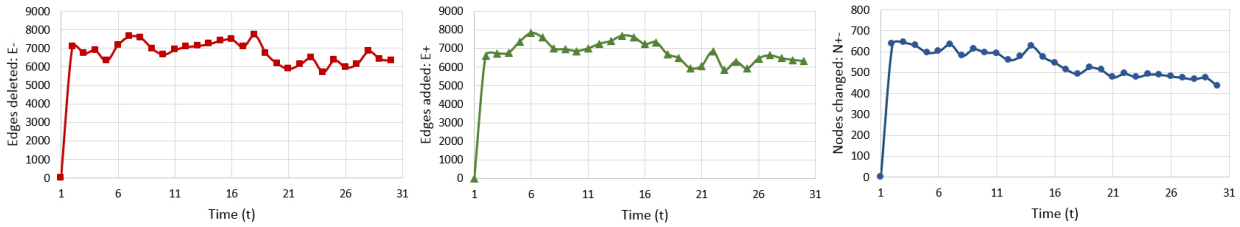


Figure 4.5: The structural changes in the Travian Trades dataset over 30 snapshots.

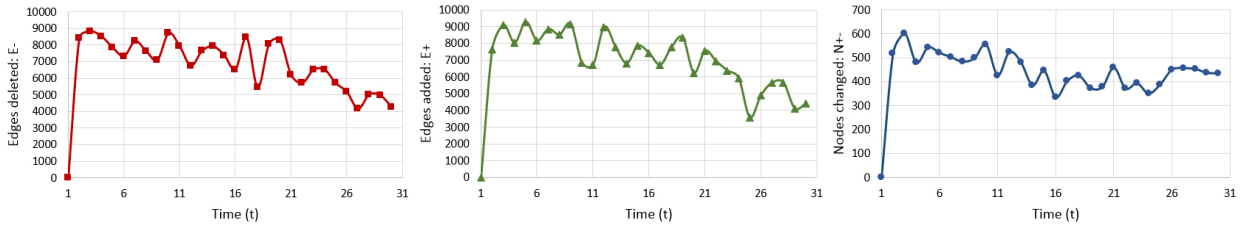


Figure 4.6: The structural changes in the Travian Messages dataset over 30 snapshots.

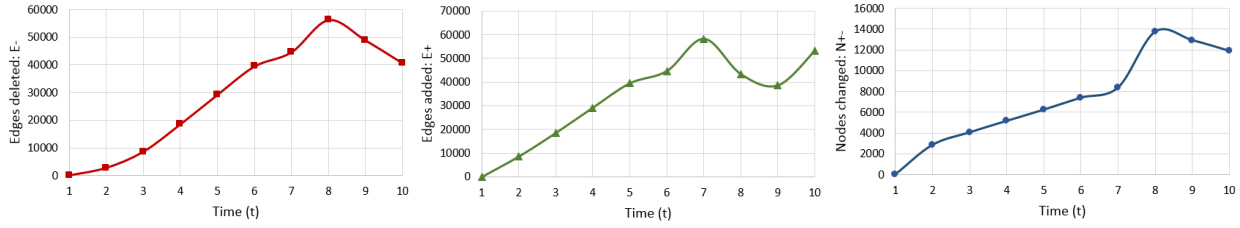


Figure 4.7: The structural changes in hep-ph dataset over 10 snapshots.

- **iLCD**[3] **[30].** iLCD is another well known community detection approach for dynamic social networks which works by first adding edges and then merging the similar ones. It takes the dynamics of the network into account.

- **OSLOM**[4] **[78].** The Order Statistics Local Optimization Method (OSLOM) is a versatile community detection algorithm that can handle most types of graph properties including edge directions and weights, overlapping communities, hierarchies and community dynamics. It is based on the local optimization of a fitness function expressing the statistical significance of clusters with respect to random fluctuations.

- **InfoMap**[5] **[113].** InfoMap is a static community detection method that calculates the probability flow of random walks and decomposes the network into modules by compressing a description of the flows. Since this is a static algorithm, we run it separately on each snapshot.

- **Louvain**[6] **[23].** The Louvain method is a static community detection approach designed to optimize modularity using heuristics. Small communities are found by optimizing modularity locally for all nodes. Then each community is grouped into a single node, and the first step is repeated. We run this algorithm separately on every network snapshot.

4.2.3.2 Datasets

To illustrate the strength and effectiveness of our approach, we selected some communication networks from the SNAP[7] graph library as well as two networks (messages and trades) from

[3] http://www.cazabetremy.fr/iLCD.html

[4] http://www.oslom.org/software.htm

[5] http://www.mapequation.org/code.html

[6] https://sites.google.com/site/findcommunities

[7] http://snap.stanford.edu/

a well-known multiplayer online game. Statistics for the datasets are provided in Table 4.3, and description of the datasets is as follows:

AS-Oregon Graph [83]. The dataset contains 9 graphs of Autonomous Systems (AS) peering information inferred from Oregon route-views between March 31, 2001 and May 26, 2001. These 9 graphs are different snapshots from the data with a minimum of 10,670 and maximum of 11,174 nodes. The number of edges ranges from 21,999 in the snapshot of April 07, 2001 to 23,409 in May 26, 2001. Figure 4.2 shows the number of edges added and deleted, as well as the number of nodes involved in the changes for the AS-Oregon dataset.

Table 4.3: Dataset Summary

Data	Oregon	Internet	Enron	Travian (Messages)	Travian (Trades)	hep-ph
Min # of nodes	10,670	2,948	101	1,373	964	12,711
Max # of nodes	11,174	6,477	137	2,100	1,336	34,401
Min # of edges	21,999	3,386	1,432	8,511	8,080	39,981
Max # of edges	23,409	13,233	5,015	19,242	10,221	51,485
# of snapshots	9	733	12	30	30	10

AS-Internet Routers Graph [83]. Similar to AS-Oregon, this is a communication network of who-talks-to-whom from the Border Gateway Protocol logs of routers in the Internet. The dataset contains 733 daily snapshots for 785 days from November 8, 1997 to January 2, 2000. The number of nodes in the largest snapshot is 6,477 (with 13,233 edges). Figure 4.3 illustrates that the structure of the graph can change dramatically at each snapshot.

Enron Email [126]. The Enron email network contains email message data from 150 users, mostly senior management of Enron Inc., from January 1999 to July 2002. Each email address is represented by an unique ID in the dataset, and each link corresponds to a message between the

sender and the receiver. After a data refinement process, we simulate the network evolution via a series of 12 growing snapshots from January 2000 to December 2000. Enron network changes are shown in Figure 4.4.

Travian[8] [61] Travian is a popular browser-based real-time strategy game with more than 5 million users. Players seek to improve their production capacity and construct military units in order to expand their territory through a combination of colonization and conquest. Each game cycle lasts a fixed period (a few months) during which time the players vie to complete construction on one of the Wonders of the World. To do this, they form alliances of up to 60 members under a leader or a leadership team; in this article these alliances are used as the ground truth for evaluating the community detection procedure.

Travian has an in-game messaging system (IGM) for player communication which was used to create our Messages network. Each player can submit a request to trade a specific resource. If another player finds this request interesting, he/she can accept it and the trade will occur; this data was used to build the Trade network. About 70% of messages are exchanged between users in the same alliance (community) making it more predictive of community structure than the Trades network since only 30% of edges in this network represent trades occurred between players within the same alliance. The structural changes in both Travian datasets are shown in Figures 4.5 and 4.6.

HEP-PH Citation Graph [83]. The HEPPH (high energy physics theory) citation graph from the e-print arXiv covers all the citations within a dataset of $n = 29,555$ papers with $e = 352,807$ edges. If a paper i cites paper j, the graph contains a directed edge from i to j. If a paper cites, or is cited by, a paper outside the dataset, the graph does not contain any information about this. This data covers papers in the period from January 1993 to March 2002. There are ten snapshots where each snapshot includes data from twelve months except the last snapshot which only contains edges from the first three months of 2002 (Figure 4.7).

[8] ial.eecs.ucf.edu/travian.php

4.2.3.3 Evaluation Metrics

We evaluated the performance of all methods using the following metrics.

4.2.3.3.1 Normalized Mutual Information

One way to measure the performance of a community detection algorithm is to determine how similar the partition delivered by the algorithm is to the desired partition, assuming ground truth information about the community membership exists. Out of several existing measures [52], we selected the standard version of normalized mutual information (NMI) [38], which is computed as follows:

$$\mathbf{I}_{norm}\mathbf{(X,Y)} = \frac{2I(X,Y)}{H(X) + H(Y)}, \tag{4.14}$$

where $I(X,Y)$ is mutual information between two random variables X and Y (i.e. two community partitions) [91]:

$$\mathbf{I(X,Y)} = \sum_{x}\sum_{y} P(x,y)\log\frac{P(x,y)}{P(x)P(y)}, \tag{4.15}$$

Here $P(x)$ indicates the probability that $X = x$ and joint probability $P(x,y)$ equals to $P(X = x, Y = y)$. $H(X)$ and $H(Y)$ are the entropies of X and Y, respectively. NMI lies in the range [0,1], equaling one when two partitions X and Y are exactly identical and zero when they are totally independent. Code to calculate NMI can be downloaded at: `https://sites.google.com/site/andrealancichinetti/software`.

4.2.3.3.2 *Modularity*

Modularity measures the difference between the number of intra-community edges for a given community partition vs. a random distribution of edges; it is the most popular qualitative measure in detecting communities in social networks. However, it has been shown that modularity has drawbacks and becomes unreliable when networks are too sparse [53]. It is also useful to examine the number of detected communities in conjunction with modularity to ensure that the algorithm is not being overly aggressive about combining small communities in order to maximize overall network modularity.

Standard modularity **Q** is usually defined as follows:

$$\mathbf{Q} = \frac{1}{2m} \sum_{ij} [\mathbf{A}_{ij} - \frac{d_i d_j}{2m}] \delta_{c_i, c_j} \tag{4.16}$$

where $\mathbf{A}_{ij}$ is an element of the adjacency matrix, δ_{ij} is the Kronecker delta symbol, and c_i is the label of the community to which vertex i is assigned. However, modularity can be slightly different for directed networks [80] and can then be reformulated as:

$$\mathbf{Q} = \frac{1}{m} \sum_{ij} [\mathbf{A}_{ij} - \frac{d_i^{in} d_j^{out}}{m}] \delta_{c_i, c_j} \tag{4.17}$$

where $\mathbf{A}_{ij}$ is defined in the conventional manner to be 1 if there is an edge from i to j and zero otherwise. Here, the probability of the existence of an edge from vertex i to vertex j has the probability $d_i^{in} d_j^{out}/m$, where d_i^{in} and d_j^{out} are the in- and out-degrees of the vertices respectively.

4.2.4 *Evaluation*

In this chapter, we examine four research questions:

1. How does the gain function influence community detection performance?
2. Does the initialization procedure affect the performance of dynamic community detection?
3. How does D-GT perform vs. the competitor methods?
4. In cases where the community membership of a small number of the agents is known, can it be used to improve D-GT's performance?

Table 4.4: Modularity evaluation metric on the six datasets; results are averaged over all snapshots.

Algorithm/Dataset	**Oregon**	**Internet**	**Enron**	**Travian (Messages)**	**Travian (Trades)**	**hep-ph**
D-GT + Similarity	0.61±0.02	*0.59±0.02*	**0.50±0.02**	0.45±0.03	*0.43±0.02*	**0.56±0.03**
D-GTS + Similarity	**0.63±0.02**	0.57±0.04	0.48±0.02	0.44±0.02	0.42±0.01	0.55±0.02
D-GTP + Similarity	0.59±0.02	0.58±0.05	0.47±0.03	0.44±0.03	0.41±0.02	0.51±0.04
D-GT + Modularity	0.57±0.03	0.55±0.04	0.47±0.03	0.48±0.04	0.41±0.02	0.55±0.04
D-GTS + Modularity	**0.63±0.02**	0.51±0.02	0.43±0.02	*0.49±0.03*	0.40±0.03	0.54±0.02
D-GTP + Modularity	0.59±0.03	0.48±0.03	0.39±0.02	0.47±0.03	0.42±0.02	0.52±0.04
OSLOM	0.49±0.05	0.52±0.04	0.39±0.03	0.44±0.04	0.32±0.01	0.49±0.01
LabelRankT	0.44±0.01	0.41±0.09	0.34±0.05	0.42±0.04	0.36±0.03	0.43±0.06
iLCD	0.15±0.05	0.11±0.01	0.13±0.05	0.19±0.02	0.09±0.01	0.16±0.07
InfoMap	**0.63±0.04**	**0.61±0.01**	0.49±0.08	0.51±0.02	0.43±0.04	**0.56±0.04**
Louvain	0.59±0.02	0.57±0.04	0.46±0.06	**0.53±0.02**	**0.49±0.02**	0.54±0.07

We analyze the performance of the D-GT variants vs. LabelRankT, iLCD, OSLOM, InfoMap, and Louvain on the two metrics, normalized mutual information (NMI) and modularity. We hypothesize that D-GT with the modularity gain function will perform well on the modularity evaluation metric, since its stochastic search process essentially optimizes local modularity. Also D-GT's initialization procedure will not necessarily help the modularity optimization process since it can be performed effectively without considering community evolution through time. This

suggests that D-GTS is a good candidate for the modularity evaluation metric.

However, in most real-world communities, prior community membership is a good predictor of future membership. Thus we believe that D-GT is more effective at discovering the real community structure as measured by the normalized mutual information (NMI) evaluation metric. Our previous work [8] has shown that the neighborhood similarity function is a good gain function for recovering the true community structure so we hypothesize that D-GT (similarity) is the best choice for NMI.

Figure 4.8 shows the average performance of all the D-GT variants with the similarity gain function vs. OSLOM, LabelRankT, iLCD, InfoMap and Louvain. Note that it is not possible to calculate the NMI performance on the AS-Internet, AS-Oregon, Enron, and hep-ph datasets since we don't have ground truth community structure; for the Travian datasets, we use the alliance membership to calculate the NMI. D-GT (similarity) outperforms all other methods ($p < 0.01$) on this metric.

In D-GT, each node's community membership vector is initialized as the superset of all communities that it has been a member of at any time step. If most of the nodes have remained within the same communities, the correct structure is quickly discovered. In cases where there are cyclic temporal patterns, there is clearly some value in considering earlier community assignments.

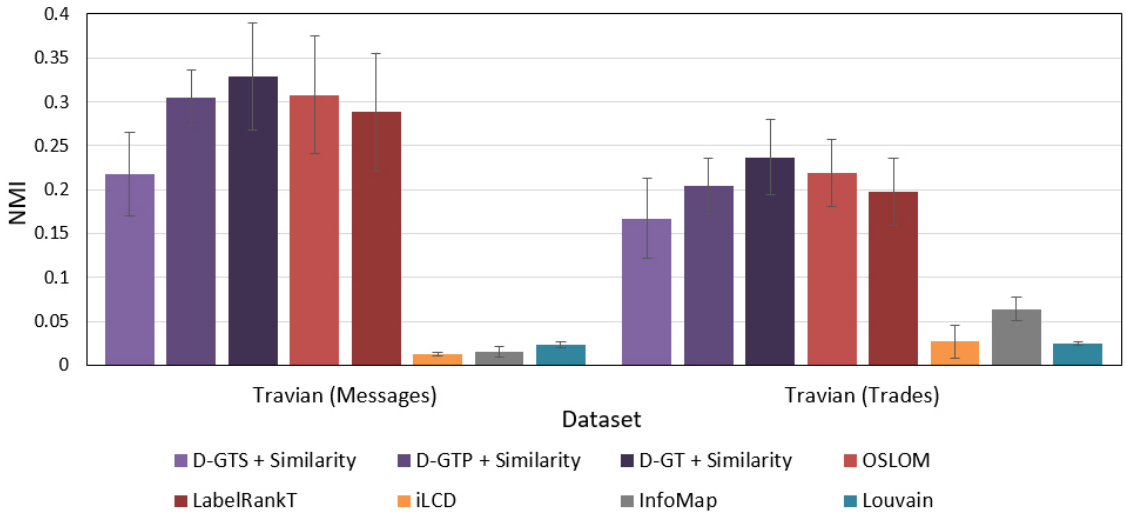


Figure 4.8: Normalized mutual information (NMI) evaluation metric on the two Travian datasets with ground truth community membership information; results are averaged over all snapshots. The variants of D-GT are colored in purple, LabelRankT (red), OSLOM (indian red), iLCD (yellow), InfoMap (gray) and Louvain (blue).

As predicted it outperforms the more myopic D-GTS (that uses no prior information) and D-GTP (one previous snapshot); paired t-test comparisons on the two Travian datasets are significant at the $p < 0.01$ level. Figure 4.9 shows that D-GT (similarity) is more successful than the other D-GT variants at correctly predicting the number of communities, as measured by summed absolute difference between predicted and actual community numbers (lower is better). Note that it is possible to do acceptably well on the NMI metric while still incorrectly estimating the actual number of communities in the dataset. OSLOM also performs well on both metrics (NMI and number of communities).

It is also useful to look at how the number of predicted communities varies between consecutive snapshots. In most cases, the number of communities should remain relatively stable, since the structure of real-world communities rarely changes completely in short period of time. This is definitely true in Travian, where the number of alliances changes relatively slowly. Figure 4.10 shows the number of predicted communities vs. time on the Travian (Trades) dataset; all of the

methods make more consistent predictions over time than LabelRankT.

Table 4.4 shows the average performance of all the D-GT variants vs. OSLOM, LabelRankT, iLCD, InfoMap and Louvain at optimizing modularity. Bold font shows the absolute best performing algorithm, with italics marking the best performing D-GT variant. All D-GT variants are competitive at optimizing modularity but none are exceptional. They outperform the other dynamic algorithms (iLCD, LabelRankT) but rarely the static community detection algorithms (Louvain and InfoMap). This is unsurprising since the modularity metric does not intrinsically reward preserving continuity between snapshots. D-GT (similarity) continues to perform well, as does D-GTS (modularity).

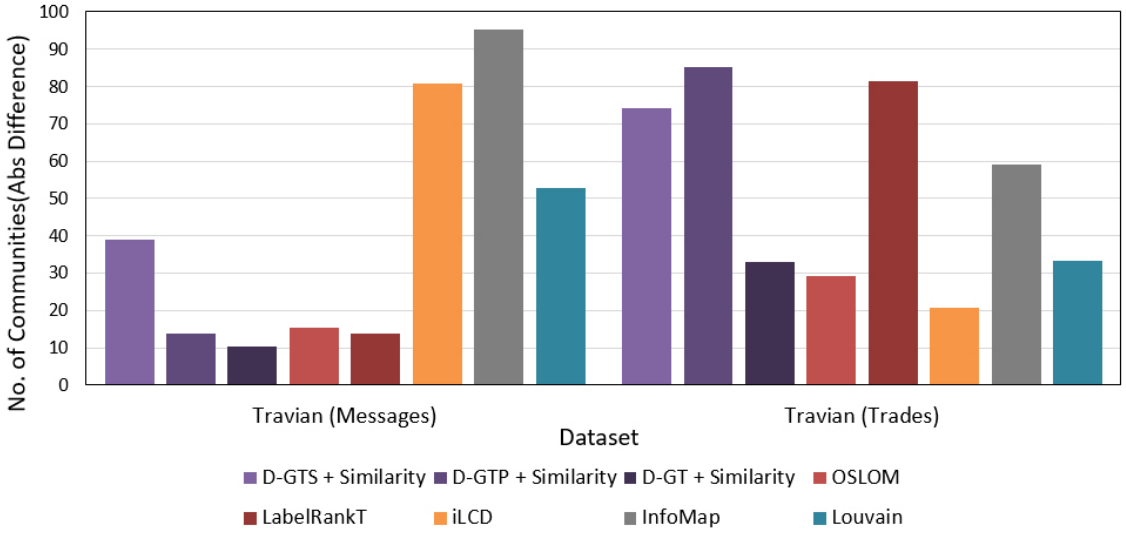


Figure 4.9: Absolute difference between the predicted number of communities and the actual number for the two Travian datasets. D-GT (with the similarity gain function) and OSLOM achieve the best performance overall at correctly predicting the number of alliances.

(Since this serves a prediction error measurement, lower is better.)

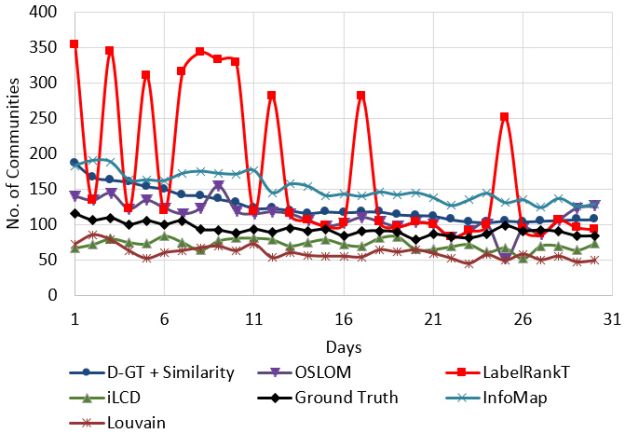


Figure 4.10: Number of predicted communities vs. time for the Travian (Trades) dataset. LabelRankT's (red) predicted number of communities varies drastically between time steps, whereas all other algorithms make more consistent predictions.

In some scenarios, it is plausible that the community membership of a small number of agents is known in advance, and the community detection procedure should leverage this information. For instance, MMOG game alliances often have a leadership council that is publicly known. To handle this problem, we developed a variant (D-GTG: D-GT with passing Ground Truth). Fig. 4.11 shows the performance improvements from increasing the size of the seed groups from 0–20% of the total number of agents for the Travian (Messages) dataset, and Fig. 4.12 shows the performance increase for Travian (Trades). Note that extracting community membership information from the network structure of Travian (Trades) is a difficult problem because only 30% of the edges in Travian (Trades) occur between players within the same alliance (community). Also the dataset has a high number of isolated nodes; about 50% of the nodes do not belong to any alliance.

Table 4.5 shows the running time of all algorithms on our largest datasets, the AS-Internet dataset (with the highest number of snapshots) and hep-ph (with the highest number of nodes). Overall, the running times provided by all algorithms were quite reasonable, as there are 733 snapshots in the Internet dataset and over 10,000 and 30,000 nodes in Oregon and hep-ph respectively.

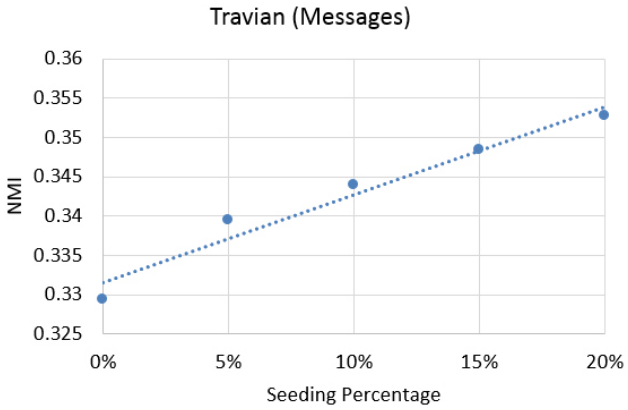


Figure 4.11: D-GTG NMI vs. seed group size on Travian (Messages)

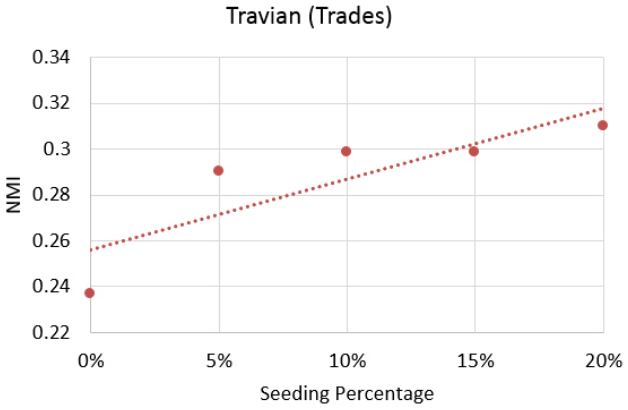


Figure 4.12: D-GTG NMI vs. seed group size on Travian (Trades)

InfoMap is the fastest algorithm on two datasets (Internet and hep-ph) and performs almost as well as OSLOM on third one (Oregon). All of the algorithms are sufficiently fast to run on large datasets.

Table 4.5: Running time (sec) of all algorithms on the AS-Internet, AS-Oregon, and hep-ph datasets

Algorithm	**Internet**	**Oregon**	**hep-ph**
D-GT	610	45	1,140
LabelRankT	840	90	1,250
iLCD	750	80	1,000
OSLOM	590	**35**	900
InfoMap	**240**	42	**620**
Louvain	380	60	730

4.3 Summary

In this dissertation, we analyze the performance of our game theoretic community detection algorithm, D-GT, on dynamic social networks. These social networks are very common in massively multiplayer online games, such as Travian, where players self-organize into rapidly changing guilds and alliances. We show that D-GT's initialization procedure in combination with the similarity gain function is very effective at recovering the true community structure of the network, both in terms of predicting the number of communities and the players' community membership vectors. It outperforms other dynamic community detection methods including LabelRankT, iLCD, and OSLOM. In cases where the communities of a small number of players (e.g. the guild leadership) is known D-GT can leverage the information to improve the NMI performance. When simply optimizing modularity, considering earlier community membership is less important and static community detection algorithms (InfoMap and Louvain) perform well at this task, but all variants of D-GT offer competitive performance.

CHAPTER 5: LEVERAGING NETWORK DYNAMICS FOR IMPROVED LINK PREDICTION

Many social networks are constantly in flux, with new edges and vertices being added or deleted daily. Fully modeling the dynamics that drive the evolution of a social network is a complex problem, due to the large number of individual and dyadic factors associated with link formation. Here we focus on predicting one crucial variable–the *rate* of network change. Not only do different networks change at different rates, but individuals within a network can have disparate tempos of social interaction. This chapter describes how modeling this aspect of network dynamics can ameliorate performance on link prediction tasks.

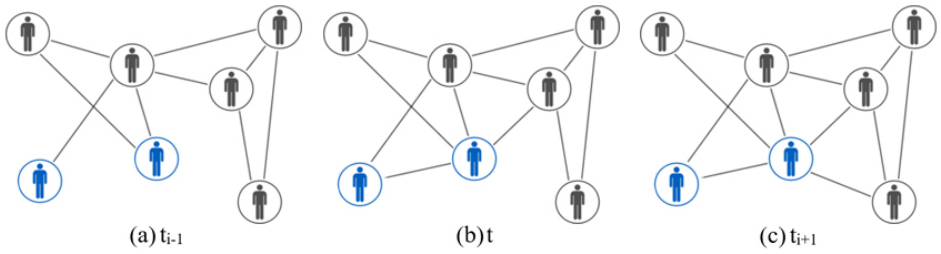


Figure 5.1: Evolution of a network over time. Blue nodes have higher *rates* of link formation. This behavior can only be captured by taking temporal information into account; RPM identifies these nodes through the use of time series.

5.1 Problem Formulation

The problem of link prediction in dynamic networks is defined as: Let graph G be the social network of interest denoted as (V, E), where V is the set of nodes and $E \in V \times V$ is the set of (directed or undirected) interactions. Let G_t be the subgraph of G containing the nodes and edges recorded at time t. In turn, let G_{t+1} be the subgraph of G observed at time $t + 1$. Using network structure up to time t, our goal is then to predict future structure of the network at time

$t + 1$.

5.2 Background

Link prediction approaches commonly rely on measuring topological similarity between unconnected nodes [5,54,138]. It is a task well suited for supervised binary classification since it is easy to create a labeled dataset of node pairs; however, the datasets tend to be extremely unbalanced with a preponderance of negative examples where links were not formed. Topological metrics are used to score node pairs at time t in order to predict whether a link will occur at a later time $t'(t' > t)$. However, even though these metrics are good indicators of future network connections, they are less accurate at predicting *when* the changes will occur (the exact value of t'). To overcome this limitation, we explicitly learn link formation rates for all nodes in the network; first, a time series is constructed for each node pair from historic data and then a forecasting model is applied to predict future values. The output of the forecasting model is used to augment topological similarity metrics within a supervised link prediction framework. Prior work has demonstrated the general utility of modeling time for link prediction (e.g., [21, 67, 108]); our results show that our specific method of rate modeling outperforms the use of other types of time series.

RPM is implemented using Spark MLlib machine learning library. Using Spark, a general purpose cluster computing system, enables us to train our supervised classifiers with the full data distribution, rather than utilizing a small balanced subset, while still scaling to larger datasets. Moreover, we evaluate the classifiers with a full test dataset, so the results are representative of the performance of the method "in the wild". Our experiments were conducted with a variety of datasets, in contrast to prior work on link prediction that has focused on citation or collaboration networks [84]. In addition to a standard co-authorship network (hep-th arXiv [123]), we analyze the dynamics of an email network (Enron [35]) and two player networks from a massively multiplayer online game (Travian [61]). Networks formed from different types of social processes may

vary in their dynamics, but our experiments show that RPM outperforms other standard approaches on all types of datasets.

5.2.1 Time Series

To construct the time series, the network G observed at time t must be split into several time-sliced snapshots, that is, states of the network at different times in the past. Afterwards, a window of prediction is defined, representing how further in the future we want to make the prediction. Then, consecutive snapshots are grouped in small sets called frames. Frames contain as many snapshots as the length of the window of prediction. These frames compose what is called Framed Time-Sliced Network Structure (S) [123]. Let G_t be the graph representation of a network at time t. Let $[G_1, G_2, ..., G_T]$ be the frame formed by the union of the graphs from time 1 to T. Let n be the number of periods (frames) in the series. And let w be the window of prediction. Formally, S can be defined as:

$$S = \{[G_1, ..., G_w], [G_{w+1}, ..., G_{2w}], ...[G_{(n-1)w+1}, ..., G_{nw}]\}$$

For instance, suppose that we observed a network from day 1 to day 9, and our aim is to predict links that will appear at day 10. In this example, the forecast horizon (window of prediction) is one day. Our aim here is to model how the networks evolve every day in order to predict what will happen in the forecast horizon. Figure 5.1 shows an example of the evolution of network over time.

5.2.2 Network Similarity Metrics

Here, we use a standard set of topological metrics to assign scores to potential links:

1. Common Neighbors (CN) [97] is defined as the number of nodes with direct relationships

with both members of the node pair:

$$CN(x, y) = |\Gamma(x) \cap \Gamma(y)| \tag{5.1}$$

where $\Gamma(x)$ is the set of neighbors of node x.

2. Preferential Attachment (PA) [15, 84] assumes that the probability that a new link is created is proportional to the node degree $|\Gamma(y)|$. Hence, nodes that currently have a high number of relationships tend to create more links in the future:

$$PA(x, y) = |\Gamma(x)| \times |\Gamma(y)| \tag{5.2}$$

3. Jaccard's Coefficient (JC) [132] assumes higher values for pairs of nodes that share a higher proportion of common neighbors relative to total number of neighbors they have:

$$JC(x, y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|} \tag{5.3}$$

4. Adamic-Adar (AA) [2], similar to JC, assigns a higher importance to the common neighbors that have fewer total neighbors. Hence, it measures exclusivity between a common neighbor and the evaluated pair of nodes:

$$AA(x, y) = \sum_{z \in |\Gamma(x) \cap \Gamma(y)|} \frac{1}{log(|\Gamma(z)|)} \tag{5.4}$$

These metrics serve as 1) unsupervised baseline methods for evaluating the performance of RPM and 2) are also included as features used by the supervised classifiers.

Table 5.1: Dataset Summary

Data	Enron	Travian (Messages)	Travian (Trades)	hep-th
No. of nodes	150	2,809	2,466	17,917
Link (Class 1)	5,015	44,956	87,418	59,013
No Link (Class 0)	17,485	7,845,525	5,993,738	320,959,876
No. of snapshots	24	30	30	20

5.2.3 Datasets

For our analysis, we selected three datasets: player communication and economic networks from the Travian massively multiplayer online game [61], the popular Enron email dataset [35], and the co-authorship network from arXiv hep-th [123]. Table 5.1 gives the network statistics for each of the datasets:

1. **Enron email dataset** [35]: This email network shows the evolution of the Enron company organizational structure over 24 months (January 2000 to December 2001).
2. **Travian MMOG** [61]: We used the communication and trading networks of users playing the Travian massively multiplayer online game. Travian is a browser-based, real-time strategy game in which the players compete to create the first civilization capable of constructing a Wonder of the World. The experiments in this dissertation were conducted on a 30 day period in the middle of the Travian game cycle (a three month period). Figure 5.2 indicates that Travian is a highly dynamic dataset, with over 90% of the edges changing between snapshots.
3. **co-authorship network hep-th arXiv** [123]: This co-authorship network shows the evolution in co-authorship relationships extracted from the arXiv High Energy Physics (Theory) publication repository between 1991 and 2010.

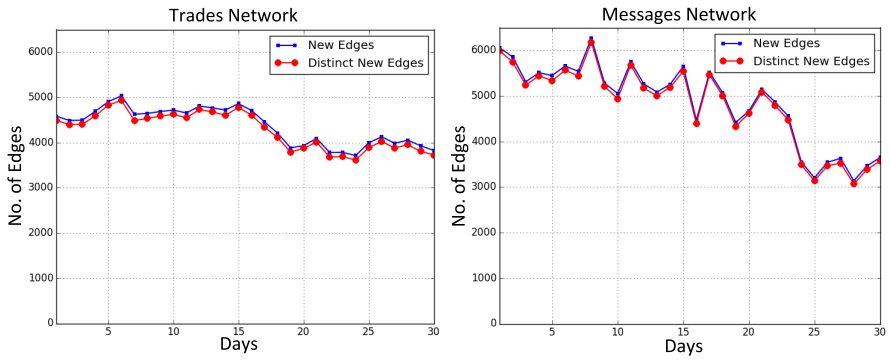


Figure 5.2: Dynamics of the Travian network (trades: left and messages: right). The blue line shows the new edges added, and the red line shows edges that did not exist in the previous snapshot.

5.3 Proposed Method

RPM treats the link prediction problem as a supervised classification task, where each data point corresponds to a pair of vertices in the social network graph. This is a typical binary classification task that could be addressed with a variety of classifiers; we use the Spark support vector machine (SVM) implementation. All experiments were conducted using the default parameters of the Spark MLlib package: the SVM is defined with a polynomial kernel and a cost parameter of 1. Algorithms were implemented in Python and executed on a machine with Intel(R) Core i7 CPU and 24GB of RAM. We have made our code and some example datasets available at: `http://ial.eecs.ucf.edu/travian.php`.

In order to produce a labeled dataset for supervised learning, we require timestamps for each node and edge to track the evolution of the social network over time. We then consider the state of the network for two different time periods t and t' (with $t < t'$). The network information from time t is used to predict new links which will be formed at time t'. One of the most important challenges with the supervised link prediction approach is handling extreme class skewness. The number of possible links is quadratic in the number of vertices in a social network, however the number of actual edges is only a tiny fraction of this number, resulting in large class skewness.

The most commonly used technique for coping with this problem is to balance the training dataset by using a small subset of the negative examples. Rather than sampling the network, we both train and test with the original data distribution and reweight the misclassification penalties. Let $G(V, A)$ be the social network of interest. Let G_t be the subgraph of G containing the nodes and edges recorded at time t. In turn, let $G_{t'}$ be the subgraph of G observed at time t'. In order to generate training examples, we considered all pairs of nodes in G_t. Even though this training paradigm is more computationally demanding it avoids the concern that the choice of sampling strategy is distorting the classifier performance [86].

Selecting the best feature set is often the most critical part of any machine learning implementation. In this dissertation, we supplement the standard set of features extracted from the graph topology (described in the previous section), with features predicted by a set of time series. Let $F_t(t = 1, ..., T)$ be a time series with T observations with A_t defined as the observation at time t and F_{t+1} the time series forecast at time $t + 1$. First, we analyze the performance of the following time series forecasting models for generating features:

1. **Simple Mean**: The simple mean is the average of all available data:

$$F_{t+1} = \frac{A_t + A_{t-1} + ... + A_{t-T}}{T}$$

2. **Moving Average**: This method makes a prediction by taking the mean of the n most recent observed values. The moving average forecast at time t can be defined as:

$$F_{t+1} = \frac{A_t + A_{t-1} + ... + A_{t-n}}{n}$$

3. **Weighted Moving Average**: This method is similar to moving average but allows one period

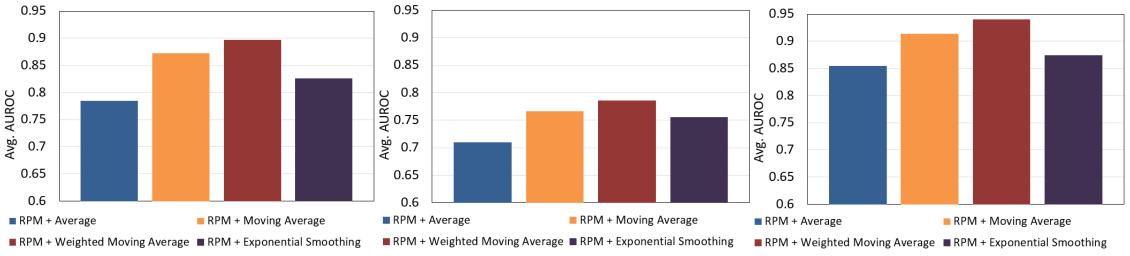
to be emphasized over others. The sum of weights must add to 100% or 1.00:

$$F_{t+1} = \sum C_t A_t$$

4. **Exponential Smoothing**: This model is one of the most frequently used time series methods because of its ease of use and minimal data requirements. It only needs three pieces of data to start: last period's forecast (F_t), last period's actual value (A_t) and a value of smoothing coefficient,α, between 0 and 1.0. If no last period forecast is available, we can simply average the last few periods:

$$F_{t+1} = \alpha A_t + (1 - \alpha) F_t$$

We identify which time series prediction model produces the best rate estimate, according to the AUROC performance of its RPM variant. Parameters of weighted moving average and exponential smoothing were tuned to maximize performance on the training dataset. Figure 5.3 shows that the best performing model was Weighted Moving Average with $n = 3$ and parameters C_1, C_2 and C_3 set to 0.2,0.3, and 0.5 respectively.



(a) (b) (c)

Figure 5.3: Performance of RPM using different forecasting models on (a) Travian Messages (b) hep-th (c) Enron. Weighted Moving Average is consistently the best performer across all datasets and is used in RPM.

Table 5.2: AUROC Performance

Algorithms / Networks	Travian(Messages)	Travian(Trades)	Enron	hep-th
RPM	**0.8970**	**0.7859**	**0.9399**	**0.7834**
Supervised-MA	0.8002	0.6143	0.8920	0.7542
Supervised	0.7568	0.7603	0.8703	0.7051
Common Neighbors	0.4968	0.5002	0.7419	0.5943
Jaccard Coefficient	0.6482	0.4703	0.8369	0.5829
Preferential Attachment	0.5896	0.5441	0.8442	0.5165
Adamic/Adar	0.5233	0.4962	0.7430	0.6696

5.4 Results

Our evaluation measures receiver operating characteristic (ROC) curves for the different approaches. These curves show achievable true positive rates (TP) with respect to all false positive rates (FP) by varying the decision threshold on probability estimations or scores. For all of our experiments, we report area under the ROC curve (AUROC), the scalar measure of the performance over all thresholds. Since link prediction is highly imbalanced, straightforward accuracy measures are well known to be misleading; for example, in a sparse network, the trivial classifier that labels all samples as missing links can have a 99.99% accuracy.

In all experiments, the algorithms were evaluated with stratified 10-fold cross-validation. For more reliable results, the cross-validation procedure was executed 10 times for each algorithm and dataset. We benchmark our algorithm against **Supervised-MA** [123]. Supervised-MA is a state of the art link prediction method that is similar to our method, in that it is supervised and uses moving average time series forecasting. In contrast to RPM, Supervised-MA creates time series for the unsupervised metrics rather than the link formation rate itself. **Supervised** is a baseline supervised classifier that uses the same unsupervised metrics as features without the time

series prediction model. As a point of reference, we also show the unsupervised performance of the individual topological metrics: 1) **Common Neighbors**, 2) **Preferential Attachment**, 3) **Jaccard Coefficient**, and 4) **Adamic-Adar**. Table 5.2 presents results for all methods on Travian (communication and trade), Enron, and hep-th networks. Results for our proposed method are shown using bold numbers in the table; in all cases, RPM outperforms the other approaches. Two-tailed, paired t-tests across multiple network snapshots reveal that the RPM is significantly better ($p < 0.01$) on all four datasets when compared to Supervised-MA.

We discover that explicitly including the rate feature (estimated by a time series) is decisively better than the usage of time series to forecast topological metrics. The rate forecast is useful for predicting the source node of future links, hence RPM can focus its search on a smaller set of node pairs. We believe a combination of topological metrics is useful for predicting the destination node, but that relying exclusively on the topological metrics, or their forecasts, is less discriminative.

5.5 Summary

In this dissertation, we introduce a new supervised link prediction method, RPM (Rate Prediction Model), that uses time series to predict the rate of link formation. The performance of RPM relies on three innovations: 1) explicit modeling of link formation rates at a node level, 2) the usage of multiple time series to leverage information from earlier snapshots, 3) training and testing with the full data distribution courtesy of the Spark fast cluster computing system. Rate is an important concept in many generative network models, but its usage has been largely ignored within discriminative classification frameworks. For instance, the stochastic actor-oriented model of network dynamics contains a network rate component that is governed by both the time period and the actors [122]. RPM does not attempt to create a general model of how the rate is affected by the properties of the actor (node), but instead predicts the link formation rate of

each node with a time series. By accurately identifying the most active individuals in the social network, RPM achieves statistically significant improvements over related link prediction methods. Our experiments were performed on networks created by a variety of social processes, such as communication, collaboration, and trading; they show that the rate of link generation varies with the type of network.

CHAPTER 6: A HOLISTIC APPROACH FOR PREDICTING LINKS IN COEVOLVING MULTIPLEX NETWORKS

As social media platforms offer customers more interaction options, such as *friending*, *following*, and *recommending*, analyzing the rich tapestry of interdependent user interactions becomes increasingly complicated. In this dissertation, we study two types of online societies: 1) players in a massively multiplayer online game (Travian) [61] 2) dialogs between Twitter users before, during, and after an exceptional event [102]. Although standard social network analysis techniques [118] offer useful insights about these communities, there is relatively little theory from the social sciences on how to integrate information from multiple types of online interactions. Rather than organizing this data into social networks separately chronicling the history of different forms of user interaction, dynamic multiplex networks [71] offer a richer formalism for modeling the social fabric of online societies.

6.1 Problem Formulation

A multiplex network is a multilayer network that shares the same set of vertices across all layers. This network can be modeled as a graph $G =< V, E >$ where V is the set of vertices and E is the set of edges present in the graph. The dynamic graph $G = \{G_0, G_1, ..., G_t\}$ represents the state of the network at different times. The network is then defined as: $G_t =< V, E_t^1, ..., E_t^M >$ with $E_t^\alpha \subseteq V \times V$, $\forall \alpha \in \{1, ..., M\}$, where each set E_t^α corresponds to the edge set of a distinct layer at time t. Thus a dynamic multiplex network is well suited for representing diverse user activities over a period of time. Here, we address the problem of predicting future user interactions from the history of past connections. Assuming the data is represented as a graph, our goal is then to predict the structure of graph G_t with α as the target layer, using information from previous snapshots as well as other layers of the network.

6.2 Proposed Method

MLP is a hybrid architecture that utilizes multiple components to address different aspects of the link prediction task. We seek to extract information from all layers of the network for the purpose of link prediction within a specific layer known as the target layer. To do so, we create a weighted version of the original target layer where interactions and connections that exist in other layers receive higher weights. After reweighting the layer, we employ the collection of node similarity metrics on the weighted network. To express the temporal dynamics of the network, we use a decay model on the time series of similarity metrics to predict future values. Finally, the Borda rank aggregation method is employed to combine the ranked lists of node pairs into a single list that predicts links for the next snapshot of the target network layer. Each component of the model is explained in more detail in the following sections.

6.2.1 Multiplex Likelihood Assignment and Edge Weighting

This component leverages information about cross-layer link co-occurrences. During the coevolution process, links may be engendered due to activity in other network layers. Some layers may evolve largely independently of the rest of the network, whereas links in other layers may be highly predictive of links in the target layer. In our proposed method, a weight is assigned to each layer based on its influence on the target layer. Weights are calculated using a likelihood function:

$$w_i = Likelihood(\text{Link in } L^{Target} | \text{Link in } L^i) \tag{6.1}$$

where L^i and w_i represent the ith layer and the weight calculated for it respectively. L^{Target} indicates the target layer for which we want to predict future links. The *Likelihood* function computes the similarity between the target layer and the ith layer; to do this, we use the current ratio of overlapping edges. Next, we calculate weights for every node pair by checking the link correspondence between two layers using the likelihood of a link being present in the target layer given the

existence of the link in the other layer at any other previous snapshot. This orders other layers in terms of their relative importance for a specific target layer. The process assigns higher weights to node pairs which occur in more than one layer (multiplex edges). The rate of link formation is incorporated into the model as the first term of the edge weight. Algorithm 2 shows the process of assigning likelihoods to layers and reweighting the adjacency matrix.

Algorithm 2 Likelihood Assignment and Edge Weighting

Input: Edge sets $(E^1, ..., E^M)$ for M layers where E^α is the edge set of target layer
Output: E^α_w weighted adjacency matrix for layer α (target layer)
//Calculate weights for the layers
for $i \in \{1, 2, ..., M\} - \{\alpha\}$ **do**
    $w_i = Likelihood(\text{Link in } L^\alpha | \text{Link in } L^i)$
end for
//Weighting target layer
for edge $e \in E^\alpha$ **do**
    $w_e = rate + \sum_{i=1 \& i \neq \alpha}^{M} w_i \times linkExist(e)$
end for

The term *rate* is defined as the average value of the source node's out-degree over previous timesteps. Function *linkExist* is used to obtain information about a link's existence in other layers during previous snapshots. It checks each layer for the presence of an edge and returns 1 if an edge is present in that layer.

6.2.2 Node Similarity Metrics

This section provides a brief description of the topological and path-based metrics for encoding node similarity that are used within our MLP framework to create ranked score lists for each node pair. These techniques are often used in isolation as unsupervised methods for link prediction. Note that $\Gamma(x)$ stands for the set of neighbors of vertex x while $w(x, y)$ represents the weight assigned to the interaction between node x and y. More details about these metrics could

be found in [142].

- **Number of Common Neighbors (CN)**

  The CN measure is defined as the number of nodes with direct relationships with both evaluated nodes x and y [97]. For weighted networks, the CN measure is:

$$CN(x,y) = \sum_{z \in |\Gamma(x) \cap \Gamma(y)|} w(x,z) + w(y,z) \tag{6.2}$$

- **Jaccard's Coefficient (JC)**

  The JC measure assumes higher values for pairs of nodes who share a higher proportion of common neighbors relative to their total neighbors. The Jaccard's Coefficient is defines as following:

$$JC(x,y) = \frac{\sum_{z \in \Gamma(x) \cap \Gamma(y)} w(x,z) + w(y,z)}{\sum_{a \in \Gamma(x)} w(x,a) + \sum_{b \in \Gamma(y)} w(y,b)} \tag{6.3}$$

- **Preferential Attachment (PA)**

  The PA measure assumes that the probability that a new link originates from node x is proportional to its node degree. Consequently, nodes that already possess a high number of relationships tend to create more links [15]:

$$PA(x,y) = \sum_{z_1 \in \Gamma(x)} w(x,z_1) \times \sum_{z_2 \in \Gamma(y)} w(y,z_2) \tag{6.4}$$

- **Adamic-Adar Coefficient (AA)**

  This metric [2] is closely related to Jaccard's coefficient in that it assigns a greater importance to common neighbors who have fewer neighbors. Hence, it measures the exclusivity of the

relationship between a common neighbor and the evaluated pair of nodes:

$$AA(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{w(x,z) + w(y,z)}{log(1 + \sum_{c \in \Gamma(z)} w(z,c))} \tag{6.5}$$

- **Resource Allocation (RA)**

  RA was first proposed in [154] and is based on physical processes of resource allocation:

$$RA(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{w(x,z) + w(y,z)}{\sum_{c \in \Gamma(z)} w(z,c)} \tag{6.6}$$

- **Page Rank (PR)**

  The PageRank algorithm [24] measures the significance of a node based on the significance of its neighbors. We use the weighted PageRank algorithm proposed in [48]:

$$PR_w(x) = \alpha \sum_{k \in \Gamma(x)} \frac{PR_w(x)}{L(k)} + (1-\alpha) \frac{w(x)}{\sum_{y=1}^{N} w(y)} \tag{6.7}$$

  where $L(x)$ is the sum of outgoing link weights from node x, and $\sum_{y=1}^{N} w(y)$ is the total weight across the whole network.

- **Inverse Path Distance (IPD)**

  The Path Distance measure for unweighted networks simply counts the number of nodes along the shortest path between x and y in the graph. Note that $PD(x,y) = 1$ if two nodes x and y share at least one common neighbor. In this article, the Inverse Path Distance is used to measure the proximity between two nodes, where:

$$IPD(x,y) = \frac{1}{PD(x,y)} \tag{6.8}$$

IPD is based on the intuition that nearby nodes are likely to be connected. In a weighted network, IPD is defined by the inverse of the shortest weighted distance between two nodes.

- **Product of Clustering Coefficient (PCF)**

  The clustering coefficient of a vertex v is defined as:

$$PCF(v) = \frac{3 \times \text{\# of triangles adjacent to v}}{\text{\# of possible triples adjacent to v}} \tag{6.9}$$

  To compute a score for link prediction between the vertex x and y, one can multiply the clustering coefficient score of x and y.

6.2.3 Temporal Link Structure

Given the network history for T time periods, we need to capture the temporal dependencies of the coevolution process. To do so, our framework uses a weighted exponentially decaying model [1]. Let $\{Sim_t(i,j), t = t_0 + 1, ..., t_0 + T\}$ be a time series of similarity score matrices generated by a node similarity metric on a sliding window of T successive temporal slices. An aggregated weighted similarity matrix is constructed as follows:

$$Sim_{(t_0+1)\sim(t_0+T)}(i,j) = \sum_{t=t_0+1}^{t_0+T} \theta^{t_0+T-t} Sim_t(i,j) \tag{6.10}$$

where the parameter $\theta \in [0,1]$ is the smoothing weight for previous time periods. Different values of θ modify the importance assigned to the most or least recent snapshots before current time $t+1$. This procedure generates a composite temporal score matrix for every node similarity metric. $Sim_{(t_0+1)\sim(t_0+T)}$ (shortened to Sim) is used by the algorithm as a summary of network activity, encapsulating the temporal evolution of the similarity matrix.

6.2.4 Rank Aggregation

Before describing the final step of our approach, let us briefly discuss existing methods for *ranked list aggregation/rank aggregation*. List merging or list aggregation refers to the process of combining a number of lists with the same or different numbers of elements in order to get one final list including all the elements. In rank aggregation, the order or rank of elements in input lists is also taken into consideration. The input lists can be categorized as *full, partial, or disjoint lists*. Full lists contain exactly the same elements but with a different ordering, partial lists may have some of the elements in common but not all, and disjoint lists have completely different elements. In this case, we are only dealing with full lists since each similarity metric produces a complete list for the same set of pairs, differing only in ordering.

In rank aggregation, distance metrics are used to find the disagreement between two lists/rankings. In general, any method of rank aggregation is desired to produce an aggregate ranking with minimal total disagreement among the input lists. Two well-known distance measures are:

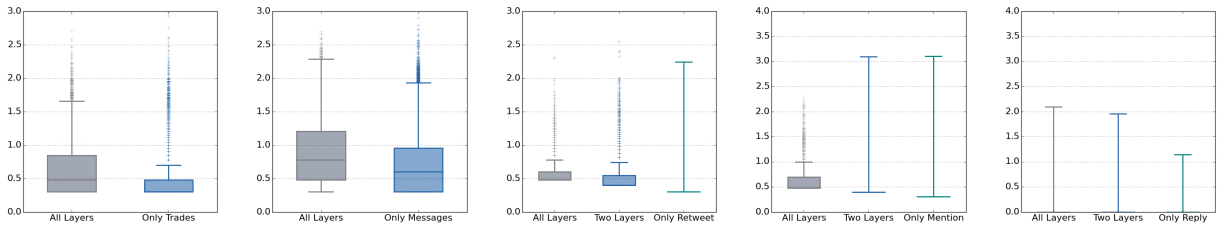
- **Spearman Footrule Distance:** This computes the distance between two ranked lists by computing the sum of differences in rankings of each element. Formally, it is given by:

$$F(L_1, L_2) = \sum_{i \in n} |L_1(i) - L_2(i)| \tag{6.11}$$

- **Kendall Tau Distance:** This counts the number of pairs of elements that have opposite rankings in the two input lists i.e. it calculates the pairwise disagreements.

$$K(L_1, L_2) = |(i, j) s.t. L_1(i) \leq L_2(j) \& L_1(i) \geq L_2(j)| \tag{6.12}$$

  where L_1 and L_2 are the input lists and $L_1(i)$ and $L_2(i)$ represent the ranks of element i in the two lists correspondingly.



(a) (b) (c) (d) (e)

Figure 6.1: Log scale box-whisker plots for user interactions in different layers of the network: (a) Travian (Trades) (b) Travian (Messages) (c) Cannes2013 (Retweets) (d) Cannes2013 (Mentions) (e) Cannes2013 (Replies)

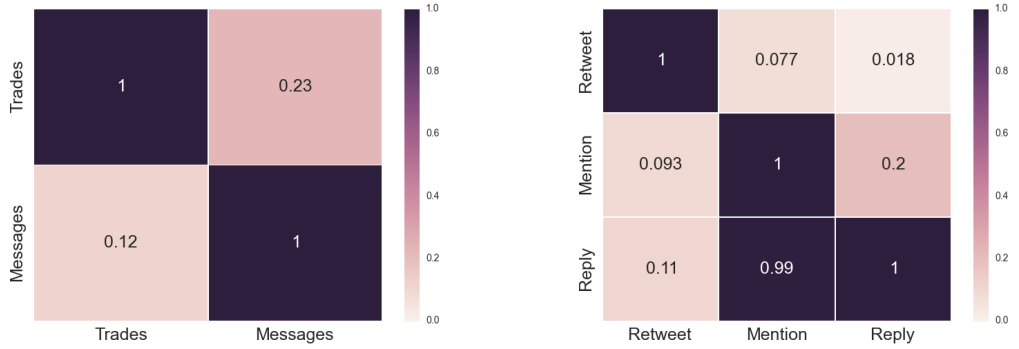


Figure 6.2: Heatmap representing the edge overlap between pairs of layers for datasets (a) Travian (b) Cannes2013

Rank aggregation methods can be categorized into two types: order-based and score-based. Order-based methods use the rank information [88] while score-based aggregation methods use score information from individual rankers. Several rank aggregation methods are described in [119], including Borda's, Markov chain, and median rank methods. Borda's method is a *rank-then-combine* method originally proposed to obtain a consensus from a voting system. Since it is based on the absolute positioning of the rank elements and not their relative rankings, it can be considered a truly positional method. For every element in the lists, a Borda score is calculated

and elements are ranked according to this score in the aggregated list. For a set of complete ranked lists $L = [L_1, L_2, L_3, ..., L_k]$, the Borda score for an element i and a list L_k is given by:

$$B_{L_k}(i) = \{count(j)|L_k(j) < L_k(i)\&j \in L_k\} \tag{6.13}$$

The total Borda score for an element is given as:

$$B(i) = \sum_{t=1}^{k} B_{L_t(i)} \tag{6.14}$$

Borda's method is computationally cheap, which is a highly desirable property for link prediction in large networks.

Algorithm 3 shows our proposed framework which incorporates edge weighting, the temporal decay model, and rank aggregation to produce an accurate prediction of future links in a dynamic multiplex network. The Borda function produces the final output of the MLP framework. Results of the proposed algorithm are compared with other state-of-the-art techniques in the next section.

Algorithm 3 Multiplex Link Prediction Framework (RPM)

1: Input: Weighted edge sets of the target layer for T previous snapshots
2: Output: Temporal aggregated score matrix S for the target layer
3: **for** each node similarity metric u **do**
4: **for** $t \in \{1, ..., T\}$ **do**
5: Calculate score matrix Sim^u_{t0+t}
6: **end for**
7: Calculate temporal similarity matrix Sim^u
8: **end for**
9: Final score matrix $S = Borda(Sim^1, ..., Sim^u)$

6.3 Experimental Study

This paper evaluates the MLP framework on networks extracted from two real-world datasets, Travian and Cannes2013. To investigate the impact of each component of our proposed method, not only do we compare our results with two other approaches for fusing cross-layer information, but we also analyze the performance of ablated versions of our method. The complete method, MLP (Hybrid), is compared with MLP (Decay Model + Rank Aggregation) and MLP (Weighted + Rank Aggregation). All of the algorithms were implemented in Python and executed on a machine with the Intel(R) Core i7 CPU and 24GB of RAM for the purpose of fair comparison. Our implementation uses Apache Spark to speed the link prediction process.

6.3.1 Datasets

We use two real-world dynamic multiplex networks to demonstrate the performance of our proposed algorithm. These networks are considerably disparate in structure and were selected from different domains (a massively multiplayer online game (MMOG) and an event-based Twitter dataset). Table 6.1 provides the network statistics for each of the datasets:

- **Travian MMOG** [61] Travian is a browser-based, real-time strategy game in which the players compete to create the first civilization capable of constructing a Wonder of the World. The experiments in this paper were conducted on a 30 day period in the middle of the Travian game cycle. In Travian, players can execute different game actions including: sending messages, trading resources, joining alliances, and attacking enemy villages. In this research, we focus on networks created from trades and messages.

- **Twitter Interactions** [102] This dataset consists of Twitter activity before, during, and after an "exceptional" event as characterized by the volume of communications. Unlike most Twitter datasets which are built from follower-followee relationships, links in this multiplex

network correspond to retweeting, mentioning, and replying to other users. The Cannes2013 dataset was created from tweets about the Cannes film festival that occurred between May 6,2013 to June 3, 2013. Each day is treated as a separate network snapshot.

Table 6.1: Dataset Summary: Number of edges, nodes, and snapshots for each network layer

Dataset		**Travian**		**Cannes2013**
No. of Nodes		2,809		438,537
No. of Snapshots		30		29
Layers/No. of Edges	Trades	87,418	Retweet	496,982
	Messages	44,956	Mention	411,338
			Reply	83,534

6.3.2 Evaluation Metrics

For the evaluation, we measure receiver operating characteristic (ROC) curves for the different approaches. The ROC curve is a plot of the *true positive rate (tpr)* against the *false positive rate (fpr)*. These curves show achievable true positive rates (TP) with respect to all false positive rates (FP) by varying the decision threshold on probability estimations or scores. For all of our experiments, we report area under the ROC curve (AUROC), the scalar measure of the performance over all thresholds. Since link prediction is highly imbalanced, straightforward accuracy measures are well known to be misleading; for example, in a sparse network, the trivial classifier that labels all samples as missing links can have a 99.99% accuracy.

6.3.3 Analysis of Cross-layer Interaction

Figure 6.1 shows log scale box-whisker plots that depict the frequency of interactions between users who are connected across multiple layers. We compare the frequency of interactions in cases where the node pair is connected on all layers vs. the frequency of being connected in a single layer (Travian) or less than all layers (for Cannes which has three layers). As expected, in cases where users are connected on all layers, the number of interactions (trades, messages, retweets, mentions and replies) is higher. The heatmap of the number of overlapping edges between different network layers (Figure 6.2) suggests that a noticeable number of edges are shared between all layers. This clearly indicates the potential value of cross-layer information for the link prediction task on these datasets. Our proposed likelihood weighting method effectively captures the information revealed by our analysis.

6.3.4 Performance of Multilayer Link Prediction

For our experiments, we adopted a moving-window approach to evaluate the performance of our temporal multiplex link prediction algorithm. Given a specified window size T, for each time period $t(t > T)$, graphs of T previous periods $(G_{tT}, ..., G_{t1})$ (where each graph consists of M layers) are used to predict links that occur at the target layer α in the current period (G_t^α). To assess our proposed framework and study the impact of its components, we compare against the following baselines:

- **RPM (Hybrid)**: incorporates all elements discussed in the framework section. It utilizes the likelihood assignment and edge weighting procedure to extract cross-layer information. Node similarity scores are modified using the temporal decay model and combined with Borda rank aggregation.

- **RPM (Likelihood + Rank Aggregation)**: This method only uses the aggregated scores calculated from the graphs weighted with cross-layer information. It does not consider the

temporal aspects of network coevolution.

- **RPM (Decay Model + Rank Aggregation)**: This method does not use the cross-layer weighting scheme and relies on temporal information alone to predict future links. The final aggregated score matrix is calculated based on forecast values at time t for each node similarity metric using the decay model.

- **Likelihood**: Weights generated by the cross-layer likelihood assignment procedure are treated as scores for every node pair. We then sort the pairs based on their score and calculate the AUROC.

- **Rank Aggregation**: This method is a simple aggregated version of all unsupervised scoring methods using the Borda's rank aggregation method applied to node similarity metrics from the target layer.

- **Unsupervised Methods**: The performance of our proposed framework is compared with eight well-known unsupervised link prediction methods described in proposed method under node similarity metrics. All unsupervised methods are applied to the binary static graph from time 0 to $t-1$ in order to predict links at time t. Only the structure of the target layer is used.

- **Average Aggregation**: In order to extend the rank aggregation model to include information from other layers of the network, we use the idea proposed in [110]. Node similarity metrics are aggregated across all layers. So for attribute X (Common Neighbors, Adamic/Adar, etc.) over M layers the following is defined:

$$X(u,v) = \frac{\sum_{\alpha=1}^{M} X(u,v)^{\alpha}}{M} \tag{6.15}$$

where $X(u,v)$ is the average score for nodes u and v across all layers and $X(u,v)^{\alpha}$ is the score at layer α. Borda's rank aggregation is then applied to the extended attributes to

calculate the final scoring matrix.

- **Entropy Aggregation**: Entropy aggregation is another extended rank aggregation model proposed in [110] where $X(u, v)$ is defined as follows:

$$X(u,v) = -\sum_{\alpha=1}^{M} \frac{X(u,v)^{\alpha}}{X_{total}} \log(\frac{X(u,v)^{\alpha}}{X_{total}}) \tag{6.16}$$

  where $X_{total} = \sum_{\alpha=1}^{M} X(u,v)^{\alpha}$. The entropy based attributes are more suitable for capturing the distribution of the attribute value over all dimensions. A higher value indicates a uniform distribution of attribute values across the multiplex layers.

- **Multiplex Unsupervised Methods**: Finally, using the definition of core neighborhood proposed in [66], we extend four unsupervised methods (Common Neighbors, Preferential Attachment, Jaccard Coefficient and Adamic/Adar) to their multiplex versions.

Table 6.2 shows the results of different algorithms on the Travian and Cannes2013 datasets. With 30 days of data from Travian and 27 days for Cannes2013, we were able to extensively compare the performance of the proposed methods and the impact of using different elements. Bold numbers indicate the best results on each target layer considered; MLP (Hybrid) is the best performing algorithm in all cases.

6.4 Discussion

In this section, we discuss the most interesting findings:

Does rank aggregation improve the performance of the unsupervised metrics? As shown in Table 6.2, although the aggregated scores matrix produced by Borda's method achieves better results than unsupervised methods in some cases (Travian message, Cannes2013 retweet and mention networks) and comparable results on others (Travian trade and Cannes2013 reply networks), it is not able to significantly outperform all unsupervised methods in any of the networks.

As discussed before, we are using the simple Borda method for the rank aggregation which does not consider the effect of each ranker on the final performance. While adding weights to the rankers or using more complex rank aggregation models such as Kemeny might achieve better results, it has been shown that those approaches have high computational complexity which makes them less suitable for large real-world networks [109, 129]. Despite the fact that the rank aggregation alone does not significantly improve the overall performance of the link prediction task, it enables us to effectively fuse different kinds of information (edge and node features, nodes similarity, etc.).

On the other hand, the Average and Entropy Aggregation methods, which are designed to consider attribute values from other layers, are able to outperform regular Rank Aggregation and MLP (Decay Model + Rank Aggregation). However, both methods use the static structure of all snapshots from time 0 to $t-1$, while MLP (Decay Model + Rank Aggregation) only incorporates the past T snapshots which makes it more suitable for large networks.

Does the likelihood assignment procedure outperform the unsupervised scores? To study the ability of our likelihood weighting method to model the link formation process, we generate results for two methods: using likelihood explicitly as a scoring method as well as using the values to generate a weighted version of the networks. First, the *Likelihood* method is used in isolation to demonstrate the prediction power of its weights as a new scoring approach. Table 6.2 shows significant improvements on unsupervised scores as well as the aggregated version of them. As expected, the more overlap between the target layer and predictor layers, the more performance improvement *Likelihood* achieves. As an example, Likelihood achieves $\sim 7\%$ of improvement on Travian (Trade) compared with $\sim 5\%$ of improvement on Travian (Message). Not only is there a lower rate of overlapping edges between those layers, but also the number of interactions is higher than the two other layers. The same holds true for Cannes2013 (Retweet) compared with the mention and reply layers.

Table 6.2: AUROC performances for a target layer averaged over all snapshots with a sliding time window of $T = 3$ for Travian layers and $T = 5$ for Cannes2013 layers used in the decay model. Variants of our proposed framework are shown at the top of the table, followed by standard unsupervised methods. The algorithms shown in the bottom half of the table are techniques for multiplex networks proposed by other research groups. The best performer is marked in bold.

Algorithms / Networks	Trade	Message	Retweet	Mention	Reply
MLP (Hybrid)	**0.821±0.001**	**0.803±0.002**	**0.812±0.002**	**0.834±0.003**	**0.839±0.002**
MLP (LH/RA)	0.802±0.001	0.790±0.002	0.809±0.003	0.814±0.004	0.816±0.003
MLP (DM/RA)	0.722±0.002	0.731±0.002	0.727±0.002	0.728±0.003	0.733±0.002
Likelihood	0.770±0.033	0.760±0.041	0.752±0.022	0.781±0.052	0.757±0.042
Rank Aggregation	0.694±0.001	0.712±0.001	0.700±0.002	0.706±0.002	0.700±0.003
Common Neighbors	0.656±0.002	0.667±0.002	0.699±0.002	0.705±0.003	0.699±0.001
Jaccard Coefficient	0.628±0.002	0.680±0.003	0.594±0.002	0.733±0.002	0.711±0.003
Preferential Attachment	0.709±0.002	0.637±0.001	0.584±0.002	0.612±0.002	0.587±0.003
Adamic/Adar	0.635±0.003	0.700±0.003	0.700±0.002	0.642±0.002	0.516±0.003
Resource Allocation	0.625±0.005	0.690±0.003	0.597±0.002	0.622±0.002	0.672±0.003
Page Rank	0.595±0.001	0.687±0.002	0.660±0.002	0.630±0.003	0.613±0.002
Inverse Path Distance	0.572±0.003	0.650±0.003	0.631±0.003	0.641±0.002	0.561±0.004
Clustering Coefficient	0.580±0.002	0.633±0.003	0.570±0.020	0.621±0.011	0.522±0.004
Average Aggregation	0.744±0.030	0.752±0.020	0.740±0.003	0.737±0.011	0.761±0.003
Entropy Aggregation	0.731±0.004	0.763±0.020	0.75±0.0030	0.758±0.031	0.744±0.002
Multiplex CN	0.729±0.004	0.643±0.013	0.672±0.003	0.716±0.003	0.733±0.002
Multiplex JC	0.666±0.031	0.619±0.012	0.580±0.003	0.736±0.002	0.722±0.002
Multiplex PA	0.722±0.010	0.646±0.012	0.580±0.003	0.640±0.003	0.621±0.003
Multiplex AA	0.671±0.010	0.690±0.031	0.671±0.003	0.669±0.003	0.552±0.003

On the other hand, the method introduced in Algorithm 2 generates a weighted version of input graphs which is used to generate a weighted version of unsupervised methods to produce the final scoring matrix. This paired with the rank aggregation method generates significantly better average AUROC performance compared with other proposed methods. Also, when temporal information from previous snapshots of the network is included, MLP (Hybrid) outperforms other

variants of MLP as well as well-known unsupervised methods. This indicates the power of overlapping links in improving the performance of link prediction in coevolving multiplex networks.

Does including temporal information improve AUROC performance? The importance of incorporating temporal information into link prediction has been discussed in our previous work [63]. However, here we are interested in analyzing the impact of this information on improving the performance of MLP. For that purpose, first, the decay model is employed in MLP (Decay Model + Rank Aggregation) to determine whether it improves the results generated by the aggregated score matrix. The final aggregated score matrix is calculated based on forecast values at time t for each unsupervised method using the decay model. As expected, this version of MLP is able to achieve up to $\sim 3\%$ of AUROC improvement using only information from the last three and five snapshots of the Travian and Cannes2013 networks respectively. On the other hand, we observed the same pattern when the decay model was added to MLP (Hybrid) along with likelihood and rank aggregation. Using the scores generated by our hybrid approach outperformed all other proposed and existing methods. The results presented here have been obtained using $T = 3$ for the Travian dataset and $T = 5$ for Cannes2013. These values are based on experiments performed on both datasets. While for Travian layers, increasing the value of T tends to improve the prediction performance slightly until $T = 3$; higher values of T may decrease the performance. The same pattern occurs for Cannes2013 layers when $T = 5$. Similarly, the value of θ is set to 0.4 for both datasets.

In summary, MLP (Decay Model + Rank Aggregation) is able to achieve results comparable to other baseline methods except Average and Entropy Aggregation since they benefit from the entire graph structure. Although rank aggregation by itself is not able to significantly improve the performance of unsupervised methods, paired with decay models and taking temporal aspects of the network, it can achieve better performance. On the other hand, the multiplex versions of the neighborhood based unsupervised methods are able to improve average AUROC performance, however the results are inconsistent and they achieve lower performance in many cases. Finally,

both MLP (Hybrid) and MLP (Likelihood + Rank Aggregation) achieve higher performance compared with all other methods, illustrating the importance of the cross-layer information created by the network coevolution process. A paired two-sample t-test is used to indicate the significance of the results produced by each method where the p-value is smaller than 0.0001. It is worth mentioning that, even though MLP (Hybrid) is able to outperform all other methods, its performance is not significantly better than MLP (Likelihood + Rank Aggregation) in the case of Travian (Message) and Cannes (Retweet).

6.5 Summary

We introduce a new link prediction framework, MLP (Multiplex Link Prediction), that employs a holistic approach to accurately predict links in dynamic multiplex networks using a collection of topological metrics, the temporal patterns of link formation, and overlapping edges created by network coevolution. Our analysis on real-world networks created by a variety of social processes suggests that MLP effectively models multiplex network coevolution in many domains.

CHAPTER 7: EXTRACTING INFORMATION FROM NEGATIVE INTERACTIONS IN MULTIPLEX NETWORKS USING MUTUAL INFORMATION

Many interesting real-world systems are represented as complex networks with multiple types of interactions and complicated dependency structures between layers. These interactions can be encoded as having a valence with positive links marking interactions such as trust and friendship and negative links denoting distrust or hostility. Extracting information from these negative interactions is challenging since standard topological metrics are often poor predictors of negative link formation, particularly across network layers. In this chapter, we introduce a method based on mutual information which enables us to predict both negative and positive relationships. Our experiments show that SMLP (Signed Multiplex Link Prediction) can leverage negative relationship layers in multiplex networks to improve link prediction performance. The role of mutual information in determining the sign of the correlation between different layers of a multiplex network is also investigated.

7.1 Proposed Method

In previous chapter, we introduced MLP [62], a hybrid architecture that utilizes multiple components to address different aspects of the link prediction task. MLP tries to extract information from all layers of the network for the purpose of predicting links within a specific layer known as the target layer. To do so, we create a weighted version of the original target layer where interactions and connections that exist in other layers receive higher weights. After reweighting the layer, we employ a collection of node similarity metrics on the weighted network. To express the temporal dynamics of the network, a decay model is used on the time series of similarity metrics to predict future values. Finally, the Borda rank aggregation method is employed to combine the

ranked lists of node pairs into a single list that predicts links for the next snapshot of the target network layer.

While MLP utilizes information from all layers of a network to improve link prediction in a target layer, it is not able to capture negative correlations among different layers of a network. For example, let us consider α and β as layers of a network N where α and β represent positive (trades) and negative (raids) relationships between users in a game respectively. Now, if our goal is to predict future links of layer α using information from β, MLP fails to capture the negative effect of a link between two nodes on β. In order to model such relationships, we propose a model based on *mutual information* that can capture the correlation sign between different layers in order to modify the MLP weighting procedure. In this section, we first provide a short description on the concept of mutual information and describe how it has been previously used for the link prediction task in single layer networks. Finally, we introduce our signed multiplex link prediction model.

7.1.1 Using Mutual Information for Link Prediction

Considering a random variable X associated with outcome x_k with probability $p(x_k)$, its self-information $I(x_k)$ can be denoted as $I(x_k) = \log \frac{1}{p(x_k)} = -\log p(x_k)$ [121]. The higher the self-information is, the less likely the outcome x_k occurs. On the other hand, the mutual information of two random variables can be denoted as:

$$I(x_k; y_j) = \log \frac{p(x|y)}{p(x)} = -\log p(x_k) - (-\log p(x_k|y_j)) = I(x_k) - I(x_k|y_j) \tag{7.1}$$

The mutual information is the reduction in uncertainty due to another variable. Thus, it is a measure of the dependence between two variables. It is equal to zero if and only if two variables are independent. Tan et al. [131] proposed the following link prediction model based on mutual information. Let $\Gamma(x)$ represent node x's neighbors, then for the node pair (x, y), the set of their

common neighbors is denoted as $O_{xy} = \Gamma(x) \cap \Gamma(y)$. Given a disconnected node pair (x, y), if the set of their common neighbors O_{xy} is available, the likelihood score of node pair (x, y) is defined as:

$$s_{xy}^{MI} = -I(L_{xy}^{1}|O_{xy}) \tag{7.2}$$

where $I(L_{xy}^{1}|O_{xy})$ is the conditional self-information of the existence of a link between node pair (x, y) when their common neighbors are known. According to the property of self-information, the smaller $I(L_{xy}^{1}|O_{xy})$ is, the higher the likelihood of link existence, $I(L_{xy}^{1}|O_{xy})$ can thus be derived as:

$$I(L_{xy}^{1}|O_{xy}) = I(L_{xy}^{1}) - I(L_{xy}^{1}; O_{xy}) \tag{7.3}$$

where $I(L_{xy}^{1})$ is the self-information of that node pair (x, y) being connected and can be estimated as:

$$p(L_{xy}^{1}) = 1 - p(L_{xy}^{0}) = 1 - \frac{C_{M}^{k_x} - k_y}{C_{M}^{k_x}} \tag{7.4}$$

where M is the total number of links and k_x is the degree of node x (refer to [131] for more details). Also, $I(L_{xy}^{1}; O_{xy})$ is the mutual information between the event that node pair (x, y) is linked and the event that the node pair's common neighbors are known. Note that $I(L_{xy}^{1})$ is calculated by the prior probability of node x and node y being connected. $I(L_{xy}^{1}; O_{xy})$ indicates the reduction in the uncertainty of nodes x and y linking due to the information provided by their common neighbors. Node pairs are then sorted based on their score s_{xy}^{MI} to determine the best candidates for link formation in future timesteps.

7.1.2 Signed Multiplex Link Prediction

Various experiments on multilayer link prediction have indicated that using neighborhood information from different layers of a network in a multiplex environment can improve the performance of link prediction. Hristova et al. [66] proposed the concept of a multilayer neighborhood where a link that exists on more than one layer in a multiplex network is called a *multiplex link*. Following the definition of a multilayer network, the ego network of a node can be redefined as the multilayer neighborhood. While the simple node neighborhood is the collection of nodes one hop away from it, the multilayer global neighborhood (denoted by GN) of a node i can be derived by the total number of unique neighbors across layers:

$$\Gamma_{GNi} = \{j \in V^{\mathcal{M}} : e_{i,j} \in E^{\alpha \cup \beta}\} \tag{7.5}$$

Similarly, the core neighborhood (denoted by CN) of a node i across layers of the multilayer network is defined as:

$$\Gamma_{CNi} = \{j \in V^{\mathcal{M}} : e_{i,j} \in E^{\alpha \cap \beta}\} \tag{7.6}$$

The goal is to determine the type of correlation (negative or positive) between two layers of a multiplex network. To this end, we calculate the average mutual information value for a target layer based on predictor layers. There are two assumptions: 1) Using the core neighborhood, if there is a negative correlation between two layers, the value of average mutual information would decrease substantially but would not change significantly if there is a positive correlation between the two layers. 2) If the global neighborhood is used to calculate the value of mutual information for a node pair, this would increase the value of average mutual information if there is a positive correlation between two layers and would not change significantly otherwise. After the sign of the relationship between the target layer and other predictor layers has been determined, MLP is used

for predicting future links with the minor addition that layer weights can have both negative and positive values.

Given a disconnected node pair (x, y), the mutual information of link existence (assuming that the set of their common neighbors O_{xy} is available) can be derived as:

$$I(L^1_{xy}; O_{xy}) = I(L^1_{xy}) - I(L^1_{xy}|O_{xy}) \tag{7.7}$$

where:

$$I(L^1_{xy}|O_{xy}) = \sum_{z \in O_{xy}} I(L^1_{xy}|z) \tag{7.8}$$

For a multiplex network, O_{xy} can be redefined to use the global and core neighborhood of the two nodes. As a result, the average mutual information of a target layer would be defined as:

$$MI^{\alpha} = \sum_{x,y \in E^{\alpha} \& x \neq y} I(L^1_{xy}; O_{xy}) \tag{7.9}$$

The value of MI^{α} is calculated using information from a predictor layer β. It can be used to indicate the sign (negative or positive) of the correlation between the link formation in two layers. This sign is then used within the second phase of link prediction task (MLP) to assign weights to all predictor layers and hence improve the performance of link prediction task for the target layer α. More details on the relationship between the value of MI^{α} and the sign of correlation between layers are provided in Section 7.2.3.

7.2 Experimental Study

We evaluate the SMLP framework on networks extracted from two real-world datasets, Travian and Cannes2013. Not only do we compare our results with two other approaches for fusing cross-layer information, but we also consider scores generated by mutual information paired with

core and global neighborhood as distinct methods. This enables us to investigate the impact of each multiplex neighborhood metric on the final results. Hence, SMLP is compared with our previous model MLP, Mutual Information (N) a single layer link prediction method, Mutual Information (CN) and (GN) which incorporate core and global neighborhood definitions respectively. All of the algorithms were implemented in Python and executed on a machine with the Intel(R) Core i7 CPU and 24GB of RAM for the purpose of fair comparison.

7.2.1 Datasets

We use two real-world dynamic multiplex networks to demonstrate the performance of our proposed algorithm. These networks are considerably disparate in structure and were selected from different domains. Negative links (raids) exist between users of our MMOG dataset to evaluate the performance of our method in predicting negative links as well as examining these layers on positive ones. Table 7.1 provides the network statistics for each of the datasets:

- **Travian MMOG** [61] Travian is a browser-based, real-time strategy game in which the players compete to create the first civilization capable of constructing a Wonder of the World. The experiments in this paper were conducted on a 30 day period in the middle of the Travian game cycle. In Travian, players can execute different game actions including: sending messages, trading resources, joining alliances, and raiding enemy villages to loot resources. In this research, we focus on networks created from trades and messages as well as raids which represents a negative relationship between players.

- **Twitter Interactions** [102] This dataset consists of Twitter activity before, during, and after an "exceptional" event as characterized by the volume of communications. Unlike most Twitter datasets which are built from follower-followee relationships, links in this multiplex network correspond to retweeting, mentioning, and replying to other users. The Cannes2013 dataset was created from tweets about the Cannes film festival that occurred between May

6,2013 to June 3, 2013. Each day is treated as a separate network snapshot.

Table 7.1: Dataset Summary (Network Layers)

Dataset		**Travian**		**Cannes2013**
No. of Nodes		2,809		438,537
No. of Snapshots		30		29
Layers/No. of Edges	Trades	87,418	Retweet	496,982
	Messages	44,956	Mention	411,338
	Raids	56,765	Reply	83,534

7.2.2 Evaluation Metrics

For the evaluation, we measure receiver operating characteristic (ROC) curves for the different approaches. The ROC curve is a plot of the *true positive rate (tpr)* against the *false positive rate (fpr)*. These curves show achievable true positive rates (TP) with respect to all false positive rates (FP) by varying the decision threshold on probability estimations or scores. For all of our experiments, we report area under the ROC curve (AUROC), the scalar measure of the performance over all thresholds. Since link prediction is highly imbalanced, straightforward accuracy measures are well known to be misleading; for example, in a sparse network, the trivial classifier that labels all samples as missing links can have a 99.99% accuracy.

7.2.3 Analysis of Multilayer Neighborhood

As mentioned before, our assumption is that both core and global neighborhood definitions would enable us to study correlations between different layers of a network. Figure 7.1 shows av-

erage mutual information values for Travian network for each day of the 30 day period. There are negative and positive correlations between trades (or messages)-raids and trades-messages respectively. As shown in the figure, at every timestep of the network, the average value decreases when the core neighborhood is used in the case of negative correlation (Figure 7.1 (a) and (b)) and does not change significantly when global neighborhood is used to calculate the average value. On the other hand, as shown in Figure 7.1 (c), when dealing with positive correlations (trades-messages), the value of average mutual information does not change significantly using the core neighborhood but increases drastically using the global neighborhood definition (the average difference is less than one). Hence, this justifies our assumption that the average mutual information value of a certain layer can be used to determine the sign of its role in target layer link prediction.

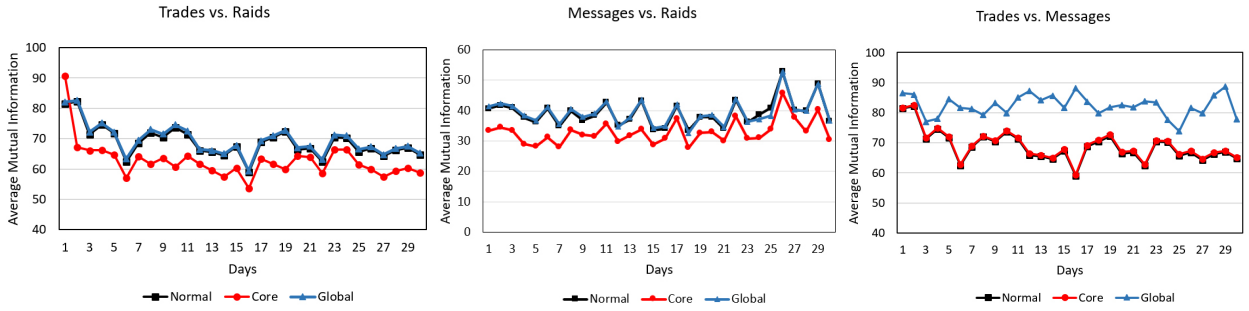


Figure 7.1: Average mutual information values over time calculated using normal and core neighborhood definitions for (a) Travian trades as the target layer and raids as the predictor, (b) Travian messages as the target layer and raids as the predictor, and (c) Travian trades as the target layer and messages as the predictor.

7.2.4 Performance of Signed Multiplex Link Prediction (SMLP)

Table 7.2 shows the results of different algorithms on the Travian and Cannes2013 datasets. With 30 days of data from Travian and 27 days for Cannes2013, we were able to extensively compare the performance of the proposed methods and the impact of using different elements. AUROC performances for a target layer averaged over all snapshots are calculated, and our proposed frame-

work is shown at the top of the table, followed by variants of mutual information based link prediction models using different definitions of neighborhood (N which stands for Normal proposed by Tan et al. [131], CN stands for Core Neighborhood and GN stands for Global Neighborhood). The algorithms shown in the bottom half of the table (Average Aggregation and Entropy Aggregation [110]) are techniques for multiplex networks proposed by other research groups. The settings given in [62] were used for MLP.

Table 7.2: AUROC performances for a target layer averaged over all snapshots and ten runs. Our proposed framework is shown at the top of the table, followed by variants of mutual information based link prediction models. The algorithms shown in the bottom half of the table are techniques for multiplex networks proposed by other research groups. The best performer is marked in bold.

Algorithms / Networks	Trade	Message	Raids	Retweet	Mention	Reply
SMLP	**0.871±0.013**	**0.843±0.031**	**0.793±0.007**	**0.812±0.002**	**0.834±0.003**	**0.839±0.002**
MLP	0.821±0.001	0.803±0.002	0.758±0.001	0.812±0.002	0.834±0.003	0.839±0.002
Mutual Information (CN)	0.740±0.016	0.753±0.011	0.744±0.013	0.774±0.009	0.759±0.012	0.782±0.005
Mutual Information (GN)	0.737±0.010	0.746±0.011	0.747±0.009	0.771±0.011	0.767±0.012	0.773±0.009
Mutual Information (N)	0.716±0.012	0.727±0.006	0.703±0.012	0.731±0.007	0.725±0.013	0.742±0.014
Average Aggregation	0.744±0.030	0.752±0.020	0.658±0.017	0.740±0.003	0.737±0.011	0.761±0.003
Entropy Aggregation	0.731±0.004	0.763±0.020	0.661±0.009	0.749±0.003	0.758±0.031	0.744±0.002

Bold numbers indicate the best results on each target layer considered. As expected, SMLP is the best performing algorithm in all cases since not only it utilizes both historical and cross-layer information, but also mutual information enables SMLP to capture negative correlations between different layers. As a result, node pairs that are connected by raids in Travian, are penalized for this connection and eventually receive a lower score compared to node pairs that are only connected on messages and trades. This holds true when raids is the target layer and two nodes are connected on either messages or trades layers which are negatively correlated with raids. On the other hand, it is evident that methods specifically designed for multiplex link prediction outper-

form Mutual Information (N) which is unable to leverage cross-layer information. Also, Average Aggregation and Entropy Aggregation are able to achieve higher AUROC scores compared with Mutual Information based methods since they collect more information from the network using different similarity metrics such as common neighbors, Adamic/Adar, etc. Finally, for Twitter layers, SMLP and MLP achieve similar results since there are no negative layers to modify the sign of the weights associated with different layers of the network.

CHAPTER 8: CONCLUSION AND FUTURE WORK

The overarching aim of my research is to study large-scale human behaviors in coevolving multiplex networks. Due to the lack of good standardized datasets, there has been relatively little research on dynamic multiplex networks, as compared to static single layer ones. The main contribution of this dissertation is 1) extensive experiments on the dynamics of networks formed from diverse social processes and the creation of a dynamic multiplex network dataset from a massively multiplayer online game, which can be used to study a variety of human behaviors; 2) introducing a new game theoretic model for community detection in dynamic networks (D-GT); 3) proposing supervised (RPM) and unsupervised methods (MLP and SMLP) for link prediction in multiplex coevolving networks for both positive and negative links. We demonstrate that our holistic approach for modeling network dynamics in coevolving, multiplex networks outperforms state of the art approaches.

At the beginning of this research, a detailed analysis was done on the Travian MMOG to gain a better understanding of the nature of such environments as well as to collect data for future studies. Most online social media platforms are optimized to support a limited range of social interactions while relations in MMOGs are often formed during the course of gameplay and evolve as the game progresses. These relations are cognitively comparable to real-world relationships which makes them more interesting for researchers. The domain knowledge extracted from the extensive analysis of this dataset helped us to introduce various models for predicting group structures as well as link formation in these networks. The Travian massively multiplayer online game has served as a valuable testbed, enabling us to evaluate our social modeling algorithms in a complex and rich environment. Due to the dearth of publicly available data, many of the published prediction models have only been tested on coauthorship networks, such as DBLP and arXiv. However our results show that networks formed through different social processes (e.g., aggression vs. communication) exhibit different characteristics, necessitating experimentation on many types of datasets.

To solve the problem of community and group detection in social networks, we proposed D-GT (Dynamic Game Theoretic community detection), a model which uses stochastic optimization to find the best community structure, assuming that the nodes are modeled as rational players who seek to maximize their personal utility while playing a community membership game with neighboring nodes. We examined the performance of varying the amount of information propagated from prior snapshots. Our results on various datasets from different backgrounds indicate that D-GT is able to achieve a higher performance compared with other state-of-the-art methods.

After completing the study of the high level player behaviors and their alliance formation processes, we investigated dyadic interactions between players in order to create predictive models of link formation. We proposed RPM (Rate Prediction Model), a supervised link prediction method that learns the rates of player interactions. The aim of link prediction is to forecast connections that are most likely to occur in the future, based on examples of previously observed links. A key insight is that it is useful to explicitly model *network dynamics*, how frequently links are created or destroyed when doing link prediction. Not only do different networks change at different rates, but individuals within a network can have disparate tempos of social interaction. Modeling this aspect of network dynamics can ameliorate performance on link prediction tasks.

Finally, as networks extracted from social media platforms frequently include multiple types of links that dynamically change over time; we need a more advanced algorithm to model link formation in such environment. We organize this data into a dynamic multiplex network, where each layer is composed of a single edge type linking the same underlying vertices. In coevolving networks, links in one layer result in an increased (or a decreased) probability of other types of links forming between the same node pair. Hence we believe that a holistic approach in which all the layers are simultaneously considered can outperform a factored approach in which link prediction is performed separately in each layer. Our models MLP and SMLP are designed such that they offer a holistic approach for multiplex networks but with a slight difference. While MLP is only able to capture cross layer information in a network where all layers are positively correlated,

SMLP uses the concept of mutual information to extract useful data from negative layers as well. It would be interesting to extend our results to other multilayer networks that contain negative interactions, particularly studying datasets that contain multiple negative interaction layers to see if they result in positive correlations. Another promising line of inquiry is investigating the usage of multiple features (beyond common neighbors) to calculate mutual information, since different structural features highlight different aspects of network formation. Also, the rank aggregation method used in our model assigns the same weight to all rankers. However, for different networks, each scoring method might add differing value to the final scoring matrix. Hence, we think that using a weighted Borda rank aggregation to calculate final scores could improve prediction results.

Using a combination of our techniques, D-GT, RPM, and SMLP, we are able to successfully model the changes in community structure, rate of link formation, and the coevolution of different network layers. In future work it would be valuable to introduce a single unified model, capable of exploiting dependencies between the dynamics of different processes.

APPENDIX A: TRAVIAN DATASET TECHNICAL DESCRIPTION

In this appendix, we briefly describe the Travian MySQL dataset which was provided to us by Drs. Rolf T. Wigand and Nitin Agarwal at University of Arkansas. This chapter includes details about the original tables provided to us as well as tables of the final dataset created for this dissertation. The Travian dataset consists of two sub datasets called: `travian_lmu_research_1` and `travian_lmu_research_2` which from now on, will be called *Travian1* and *Travian2* respectively. We include the list of important tables, along with a short description. Then we conclude by discussing the networks generated from this dataset. This chapter also provides a list of SQL queries used for generating networks.

1. **Travian1 (travian_lmu_research_1)**

   (a) **a_stats:** This table contains the following information for a specific day: date, number of registered users, activated players, number of active players (in the last seven days, three days, 24 hours, and six minutes), number of players (all four types), number of visitors (inactive users) and number of attacks. Maintaining this table helps us to save time—-instead of running queries on different tables we can get a summary of the important statistics quickly.

   (b) **s_dorf (s-villages):** This table keeps data about villages in the Travian game. Each village has an owner which is a registered and active user. Users can own multiple villages since they can attack and conquer other users' villages. Each village has a unique ID known as *did* across the whole dataset. An ID for the village owner is stored here as *uid*, based on the data saved in the users' table. Other important information which can be extracted from this table include: village tribe, location of the village on the world map (x,y coordinates), name of the village selected by the owner, type of village (main, occupied and oasis), production rate per hour (clay, wood, iron and

wheat), consumption rate by troops and buildings, point in time of the last production, stock level of the resources and size of hideout, which is important during a war since the user can only protect that amount of resources from attack. Note: each version of the dataset only keeps the resource production rate for the specific date that the data backup occurred. If we want to keep track of changes in the production rates over time, we need to examine other backups of the dataset from different snapshots (daily or weekly).

(c) **s_ereignisse (s_event):** The table tracks the occurrence of specific events: building construction, building demolition, attack, raid, support, new village, troop return and friendly trade. There is a data field called *zeid* which shows the data of the event. Unfortunately, this table is missing in the current version of the dataset.

(d) **s_gebaeude (s_buildings):** This table contains data about different buildings of a village. Each record of this table has a *did* for the village ID, *gid* which is the building ID and stage of the building at this point of time. For example if we have a data backup on day three of the game, this shows the state of the building at that specific day.

(e) **s_marktplatz (s_marketplace):** Information about marketplaces is stored in this table. In the marketplace players trade resources or buy items they need in exchange for gold.

(f) **s_spieler (s_player):** Player information is saved in this table. Right now, the table contains information about 2,107 users which is a small subset of the users registered in the a_stats table. The primary key of the table is the user ID saved as *uid*. Each player selects a name and has a village that they founded designated with a *did*. If that village has joined an alliance, its ID is saved as *aid*. Other important fields of this table include: number of villages, time stamp of last login (to keep track of the user's activities), login status, time zone and time format.

(g) **s_transporte (s_transports):** Another valuable table which is also missing from this

dataset is s_transports. This table saves information about the transportation of resources between different users. Each record of this table contains information about source village, destination village, duration of transport in seconds, number of utilized traders, quantity of materials (wood, clay, iron, crops and wheat) and number of times the trader repeated this transport order.

(h) **s_truppen (s_troops):** Each user has its own army of troops; information about the troops is saved in this table.

(i) **s_ww:** This contains the information about wonders of the world.

2. **Travian2 (travian_lmu_research_2)**

(a) **f_forum:** The forum table keeps information about the forum each alliance is using for communications. Each forum has a unique ID and contains posts from alliance members.

(b) **f_posts:** This table includes all activities of different users posting on different forums within the game.

(c) **f_topics:** This table includes all topics users posted on different forums within the game.

(d) **s_allianz (s_alliance):** Each alliance has a unique ID. Alliance name, abbreviation, date of foundation and description are recorded in this table.

(e) **s_nachrichten1 (s_news):** This table stores news exchanged between users. This news can either be a piece of information about the game or something like a message between users. It uses *uid1* to record the sender's ID and *uid2* to save the receiver's information. The time and status of the message (read/unread) can also be found in this table.

(f) **s_nachrichten2**: Text of the news mentioned in the previous item is included in this table.

(g) **x_eroberungen (x_conquers/captures):** There are some attacks which occur that result in a village becoming conquered/captured. The list of these attacks is saved in this table. The following information is recorded for each attack: *uid* of the conqueror, *uid* of the user who lost the village, alliance of the attacker, alliance of the defender, village ID of the attacker, village ID of the defender, date and time of the conquest.

(h) **x_handel (x_trade):** This table tracks trades between users who are not in the same alliance. It records information about sender, receiver, amount of each resource, total amount and the date of the trade.

(i) **x_handel2 (x_trade2):** x_trade2 stores the total number of trades between two users and also the balance of these trades.

(j) **x_raid:** This is where the data about all attacks is stored. It records the following information: user ID of the attacker and defender, looted amount of resources, sum of the looted amounts and the time of the robbery.

A dataset with the same structure as described above is stored for every day of the Travian game. To have a better understanding of the data, we combined all the data together to create a unified version of the dataset which includes the following tables: alliances, alliance members, conquers, messages, raids, reports1, reports2, reports3, stats, trades and villages. Extracting information from the table in the final unified database and creating networks requires complex queries. Here is a sample query specifically designed to generate desired daily networks from our MySQL database:

```
SELECT DISTINCT uid1,uid2 FROM table_name WHERE uid1 != 0 AND uid2
!= 0 AND uid1 NOT IN (SELECT uid FROM inactive_users) AND uid2 NOT
```

```
IN (SELECT uid FROM inactive_users) AND from_unixtime( zeit, '%Y-%m-%d'
) >= from_date AND from_unixtime( zeit, '%Y-%m-%d' ) <= to_date
```

LIST OF REFERENCES

[1] E. Acar, D. M. Dunlavy, and T. G. Kolda. Link prediction on evolving data using matrix and tensor factorizations. In *Workshops at IEEE International Conference on Data Mining*, pages 262–269, 2009.

[2] L. A. Adamic and E. Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, 2003.

[3] D. Adjeroh and U. Kandaswamy. Game-theoretic analysis of network community structure. *International Journal of Computational Intelligence Research*, 3(4), 2007.

[4] M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. In *SDM06: Workshop on Link Analysis, Counter-terrorism and Security*, 2006.

[5] M. Al Hasan and M. J. Zaki. A survey of link prediction in social networks. In *Social Network Data Analytics*, pages 243–275. Springer, 2011.

[6] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

[7] H. Alvari, A. Hajibagheri, G. Sukthankar, and K. Lakkaraju. Identifying community structures in dynamic networks. *Social Network Analysis and Mining*, 6(1):77, 9 2016.

[8] H. Alvari, S. Hashemi, and A. Hamzeh. Detecting overlapping communities in social networks by game theory and structural equivalence concept. In *Artificial Intelligence and Computational Intelligence*, pages 620–630. Springer Berlin Heidelberg, 2011.

[9] H. Alvari, S. Hashemi, and A. Hamzeh. Discovering overlapping communities in social networks: a novel game-theoretic approach. *AI Communications*, 36(2):161–177, 2013.

[10] H. Alvari, K. Lakkaraju, G. Sukthankar, and J. Whetzel. Predicting guild membership in massively multiplayer online games. In W. Kennedy, N. Agarwal, and S. Yang, editors,

Social Computing, Behavioral-Cultural Modeling and Prediction, volume 8393 of *Lecture Notes in Computer Science*, pages 215–222. Springer International Publishing, 2014.

[11] H. Alvari*, K. Lakkaraju, G. Sukthankar, and J. Whetzel. Predicting guild membership in massively multiplayer online games. In *Proceedings of the International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*, Washington, D.C., April 2014. (to appear).

[12] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 44–54. ACM, 2006.

[13] L. Backstrom and J. Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 635–644. ACM, 2011.

[14] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[15] A.-L. Barabási et al. Scale-free networks: a decade and beyond. *Science*, 325(5939):412, 2009.

[16] R. Beheshti. Normative agents for real-world scenarios. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 1749–1750. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[17] R. Beheshti and G. Sukthankar. A normative agent-based model for predicting smoking cessation trends. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 557–564. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[18] R. Beheshti and G. Sukthankar. Modeling tipping point theory using normative multi-agent systems. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 1731–1732. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

[19] F. Benevenuto, T. Rodrigues, M. Cha, and V. Almeida. Characterizing user behavior in online social networks. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 49–62. ACM, 2009.

[20] U. Bennerstedt, J. Ivarsson, and J. Linderoth. How gamers manage aggression: Situating skills in collaborative computer games. *Computer-Supported Collaborative learning*, 7:43–61, 2012.

[21] M. Berlingerio, F. Bonchi, B. Bringmann, and A. Gionis. Mining graph evolution rules. In *Machine Learning and Knowledge Discovery in Databases*, pages 115–130. Springer, 2009.

[22] G. Bianconi. Statistical mechanics of multiplex networks: Entropy and overlap. *Physical Review E*, 87(6):062806, 2013.

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

[24] S. Brin and L. Page. Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 56(18):3825–3833, 2012.

[25] B. Bringmann, M. Berlingerio, F. Bonchi, and A. Gionis. Learning and predicting the evolution of social networks. *IEEE Intelligent Systems*, 25(4):26–35, 2010.

[26] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. *Computer networks*, 33(1):309–320, 2000.

[27] S. V. Buldyrev, R. Parshani, G. Paul, H. E. Stanley, and S. Havlin. Catastrophic cascade of failures in interdependent networks. *Nature*, 464(7291):1025–1028, 2010.

[28] C. Buono, L. G. Alvarez-Zuzek, P. A. Macri, and L. A. Braunstein. Epidemics in partially overlapped multiplex networks. *PloS one*, 9(3):e92200, 2014.

[29] A. Cardillo, J. Gómez-Gardenes, M. Zanin, M. Romance, D. Papo, F. del Pozo, and S. Boccaletti. Emergence of network features from multiplexity. *arXiv preprint arXiv:1212.2153*, 2012.

[30] R. Cazabet, F. Amblard, and C. Hanachi. Detection of overlapping communities in dynamical social networks. In *IEEE International Conference on Social Computing*, pages 309–314, 2010.

[31] W. Chen, Z. Liu, X. Sun, and Y. Wang. A game-theoretic framework to identify overlapping communities in social networks. *Data Mining and Knowledge Discovery*, 21(2):224–240, 2010.

[32] K.-Y. Chiang, C.-J. Hsieh, N. Natarajan, I. S. Dhillon, and A. Tewari. Prediction and clustering in signed networks: a local to global perspective. *Journal of Machine Learning Research*, 15(1):1177–1213, 2014.

[33] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1082–1090. ACM, 2011.

[34] A. Clauset, C. R. Shalizi, and M. E. Newman. Power-law distributions in empirical data. *SIAM review*, 51(4):661–703, 2009.

[35] W. W. Cohen. Enron email dataset, 2009. http://www.cs.cmu.edu/enron/.

[36] D. J. Cook, A. Crandall, G. Singla, and B. Thomas. Detection of social interaction in smart spaces. *Cybernetics and Systems: An International Journal*, 41(2):90–104, 2010.

[37] W. Cukierski, B. Hamner, and B. Yang. Graph-based features for supervised link prediction. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 1237–1244. IEEE, 2011.

[38] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(09):P09008, 2005.

[39] D. Davis, R. Lichtenwalter, and N. V. Chawla. Supervised methods for multi-relational link prediction. *Social Network Analysis and Mining*, 3(2):127–141, 2013.

[40] A. Davoudi and M. Chatterjee. Probabilistic spreading of recommendations in social networks. In *Military Communications Conference, 2015. MILCOM 2015. IEEE*, pages 1–6. IEEE, 2015.

[41] A. Davoudi and M. Chatterjee. Modeling trust for rating prediction in recommender systems. In *SIAM Workshop on Machine Learning Methods for Recommender Systems*, pages 1–8. SIAM, 2016.

[42] A. Davoudi and M. Chatterjee. Prediction of information diffusion in social networks using dynamic carrying capacity. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 2466–2469. IEEE, 2016.

[43] A. Davoudi and M. Chatterjee. Product rating prediction using trust relationships in social networks. In *Consumer Communications and Networking Conference, 2015. CCNC 2015, 13th IEEE*, pages 1–4. IEEE, 2016.

[44] A. Davoudi and M. Chatterjee. User behavior anomaly detection for product ratings in recommender system. In *International Joint Conference on Artificial Intelligence (HINA)*. IJCAI, 2016.

[45] R. M. Dawes. Social dilemmas. *Annual review of psychology*, 31(1):169–193, 1980.

[46] M. De Domenico, A. Solé, S. Gómez, and A. Arenas. Random walks on multiplex networks. *arXiv preprint arXiv:1306.0519*, 2013.

[47] M. De Domenico, A. Solé-Ribalta, E. Cozzo, M. Kivelä, Y. Moreno, M. A. Porter, S. Gómez, and A. Arenas. Mathematical formulation of multilayer networks. *Physical Review X*, 3(4):041022, 2013.

[48] Y. Ding. Applying weighted PageRank to author citation networks. *Journal of the American Society for Information Science and Technology*, 62(2):236–245, 2011.

[49] M. Drescher, M. Korsgaard, I. Welpe, A. Picot, and R. Wigand. The dynamics of shared leadership: Building trust and enhancing performance. *Journal of Applied Psychology*, 99(5):771–783, 2014.

[50] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *ACM SIGCOMM computer communication review*, volume 29, pages 251–262. ACM, 1999.

[51] F. Folino and C. Pizzuti. An evolutionary multiobjective approach for community discovery in dynamic networks. *Knowledge and Data Engineering, IEEE Transactions on*, 26(8):1838–1852, Aug 2014.

[52] S. Fortunato. Community detection in graphs. *Physics reports*, 486(3):75–174, 2010.

[53] S. Fortunato and M. Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007.

[54] L. Getoor and C. P. Diehl. Link mining: a survey. *ACM SIGKDD Explorations Newsletter*, 7(2):3–12, 2005.

[55] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[56] S. Gomez, A. Diaz-Guilera, J. Gomez-Gardenes, C. J. Perez-Vicente, Y. Moreno, and A. Arenas. Diffusion dynamics on multiplex networks. *Physical review letters*, 110(2):028701, 2013.

[57] J. Gómez-Gardenes, I. Reinares, A. Arenas, and L. M. Floría. Evolution of cooperation in multiplex networks. *Scientific reports*, 2, 2012.

[58] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proceedings of the International Conference on the World Wide Web*, pages 403–412. ACM, 2004.

[59] A. Hajibagheri, H. Alvari, A. Hamzeh, and S. Hashemi. Community detection in social networks using information diffusion. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 702–703, 2012.

[60] A. Hajibagheri, A. Hamzeh, and G. Sukthankar. Modeling information diffusion and community membership using stochastic optimization. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 175–182, 2013.

[61] A. Hajibagheri, K. Lakkaraju, G. Sukthankar, R. T. Wigand, and N. Agarwal. Conflict and communication in massively-multiplayer online games. In *Social Computing, Behavioral-Cultural Modeling, and Prediction*, pages 65–74. Springer, 2015.

[62] A. Hajibagheri, G. Sukthankar, and K. Lakkaraju. A holistic approach for link prediction in multiplex networks. In *International Conference on Social Informatics*, 2016.

[63] A. Hajibagheri, G. Sukthankar, and K. Lakkaraju. Leveraging network dynamics for improved link prediction. In *Proceedings of the International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*, Washington, D.C., June 2016.

[64] M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. In *Proceedings of the SDM Workshop on Link Analysis, Counterterrorism and Security*, 2006.

[65] J. Hopcroft, O. Khan, B. Kulis, and B. Selman. Tracking evolving communities in large linked networks. *Proceedings of the National Academy of Sciences*, 101:5249–5253, 2004.

[66] D. Hristova, A. Noulas, C. Brown, M. Musolesi, and C. Mascolo. A multilayer approach to multiplexity and link prediction in online geo-social networks. *arXiv preprint arXiv:1508.07876*, 2015.

[67] Z. Huang and D. K. Lin. The time-series link prediction problem with applications in communication surveillance. *INFORMS Journal on Computing*, 21(2):286–303, 2009.

[68] P. Hui, E. Yoneki, S. Y. Chan, and J. Crowcroft. Distributed community detection in delay tolerant networks. In *Proceedings of ACM/IEEE International Workshop on Mobility in the Evolving Internet Architecture*, page 7, 2007.

[69] M. Humphreys and J. Weinstein. Who fights? The determinants of participation in civil war. *American Journal of Political Science*, 52(2):436–455, 2008.

[70] B. Keegan, M. Ahmed, D. Williams, J. Srivastava, and N. Contractor. Dark gold: Statistical properties of clandestine networks in massively multiplayer online games. In *IEEE International Conference on Social Computing*, pages 201–208, 2010.

[71] M. Kivela, A. Arenas, M. Barthelemy, J. Gleeson, Y. Moreno, and M. Porter. Multilayer networks. *Journal of Complex Networks*, 2:203–271, 2014.

[72] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter. Multilayer networks. *Journal of complex networks*, 2(3):203–271, 2014.

[73] M. Korsgaard, A. Picot, R. Wigand, I. Welpe, and J. Assmann. Cooperation, coordination, and trust in virtual teams: Insights from virtual games. In *Online Worlds: Convergence of the Real and the Virtual*. 2010.

[74] J. Kunegis, A. Lommatzsch, and C. Bauckhage. The Slashdot zoo: mining a social network with negative edges. In *Proceedings of the International Conference on the World Wide Web*, pages 741–750. ACM, 2009.

[75] M. Kurant and P. Thiran. Layered complex networks. *Physical review letters*, 96(13):138701, 2006.

[76] K. Lakkaraju and J. Whetzel. Group roles in massively multiplayer online games. In *Proceedings of the Workshop on Collaborative Online Organizations at the 14th International Conference on Autonomous Agents and Multiagent Systems*, 2013.

[77] C. A. Lampe, E. Johnston, and P. Resnick. Follow the reader: filtering comments on Slashdot. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1253–1262. ACM, 2007.

[78] A. Lancichinetti, F. Radicchi, J. J. Ramasco, S. Fortunato, et al. Finding statistically significant communities in networks. *PloS One*, 6(4):e18961, 2011.

[79] J. Lee and K. Lakkaraju. Predicting guild membership in massively multiplayer online games. In *Proceedings of the International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*, Washington, D.C., April 2014.

[80] E. A. Leicht and M. E. Newman. Community structure in directed networks. *Physical Review Letters*, 100(11):118703, 2008.

[81] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins. Microscopic evolution of social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 462–470. ACM, 2008.

[82] J. Leskovec, D. Huttenlocher, and J. Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the International Conference on World Wide Web*, pages 641–650. ACM, 2010.

[83] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pages 177–187. ACM, 2005.

[84] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *Proceedings of the International Conference on Information and Knowledge Management*, pages 556–559, 2003.

[85] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[86] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla. New perspectives and methods in link prediction. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 243–252, 2010.

[87] Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng. Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. In *Proceeding of the International Conference on the World Wide Web*, pages 685–694. ACM, 2008.

[88] Y.-T. Liu, T.-Y. Liu, T. Qin, Z.-M. Ma, and H. Li. Supervised rank aggregation. In *Proceedings of the 16th international conference on World Wide Web*, pages 481–490. ACM, 2007.

[89] L. Lü and T. Zhou. Role of weak ties in link prediction of complex networks. In *Proceedings of the ACM International Workshop on Complex networks Meet Information & Knowledge Management*, pages 55–58. ACM, 2009.

[90] H. Ma, M. R. Lyu, and I. King. Learning to recommend with trust and distrust relationships. In *Proceedings of the ACM Conference on Recommender Systems*, pages 189–196. ACM, 2009.

[91] D. J. MacKay. *Information theory, inference and learning algorithms*. Cambridge University Press, 2003.

[92] K. Madani and M. Hooshyar. A game theory–reinforcement learning (gt–rl) method to develop optimal operation policies for multi-operator reservoir systems. *Journal of Hydrology*, 519:732–742, 2014.

[93] A. K. Menon and C. Elkan. Link prediction via matrix factorization. In *Machine Learning and Knowledge Discovery in Databases*, pages 437–452. Springer, 2011.

[94] B. Min and K.-I. Goh. Layer-crossing overhead and information spreading in multiplex social networks. *arXiv preprint arXiv:1307.2967*, 2013.

[95] T. Murata and S. Moriyasu. Link prediction based on structural properties of online social networks. *New Generation Computing*, 26(3):245–257, 2008.

[96] M. E. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, June 2006.

[97] M. E. J. Newman. Clustering and preferential attachment in growing networks. *Phys. Rev. E*, 2001.

[98] M. E. J. Newman. Assortative mixing in networks. *Physical Review Letters*, 89(20):208701, Oct. 2002.

[99] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6):066133, 2004.

[100] N. P. Nguyen, T. N. Dinh, Y. Shen, and M. T. Thai. Dynamic social community detection and its applications. *PLoS One*, 9(4):e91431, 04 2014.

[101] V. Nicosia, G. Bianconi, V. Latora, and M. Barthelemy. Growing multiplex networks. *Physical review letters*, 111(5):058701, 2013.

[102] E. Omodei, M. De Domenico, and A. Arenas. Characterizing interactions in online social networks during exceptional events. *arXiv preprint arXiv:1506.09115*, 2015.

[103] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[104] G. Palla, P. Pollner, A.-L. Barabási, and T. Vicsek. Social group dynamics in networks. In *Adaptive Networks*, pages 11–38. Springer Berlin Heidelberg, 2009.

[105] J.-Y. Pan, H.-J. Yang, C. Faloutsos, and P. Duygulu. Automatic multimedia cross-modal correlation discovery. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 653–658. ACM, 2004.

[106] B. Pang, L. Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1–2):1–135, 2008.

[107] M. Piraveenan, K. S. K. Chung, and S. Uddin. Assortativity of links in directed networks. In *Fundamentals of Computer Science*, 2012.

[108] A. Potgieter, K. A. April, R. J. Cooke, and I. O. Osunmakinde. Temporality in link prediction: Understanding social complexity. *Emergence: Complexity & Organization (E: CO)*, 11(1):69–83, 2009.

[109] M. Pujari and R. Kanawati. Supervised rank aggregation approach for link prediction in complex networks. In *Proceedings of the International World Wide Web Conference*, pages 1189–1196, 2012.

[110] M. Pujari and R. Kanawati. Link prediction in multiplex networks. *Networks and Heterogeneous Media*, 10(1):17–35, 2015.

[111] U. N. Raghavan, R. Albert, and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76(3), 2007.

[112] G. Rossetti, M. Berlingerio, and F. Giannotti. Scalable link prediction on multidimensional networks. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 979–986. IEEE, 2011.

[113] M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008.

[114] A. Roy, Z. Borbora, and J. Srivastava. Socialization and trust formation: A mutual reinforcement? An exploratory analysis in an online virtual setting. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 653–660, 2013.

[115] V. Satuluri and S. Parthasarathy. Scalable graph clustering using stochastic flows: applications to community discovery. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 737–746, 2009.

[116] A. Saumell-Mendiola, M. Á. Serrano, and M. Boguná. Epidemic spreading on interconnected networks. *Physical Review E*, 86(2):026106, 2012.

[117] S. Scellato, A. Noulas, and C. Mascolo. Exploiting place features in link prediction on location-based social networks. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1046–1054, 2011.

[118] J. Scott. *Social Network Analysis*. Sage, 2012.

[119] D. Sculley. Rank aggregation for similar items. In *SIAM International Conference on Data Mining*, pages 587–592, 2007.

[120] M. Seshadri, S. Machiraju, A. Sridharan, J. Bolot, C. Faloutsos, and J. Leskovec. Mobile call graphs: Beyond power-law and lognormal distributions. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2008.

[121] C. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[122] T. Snijders, G. van de Bunt, and C. E. G. Steglich. Introduction to actor-based models for network dynamics. *Social Networks*, 32:44–60, 2010.

[123] P. R. d. S. Soares and R. B. C. Prudêncio. Time series based link prediction. In *International Joint Conference on Neural Networks*, pages 1–7. IEEE, 2012.

[124] A. Sole-Ribalta, M. De Domenico, N. E. Kouvaris, A. Diaz-Guilera, S. Gomez, and A. Arenas. Spectral properties of the laplacian of multiplex networks. *Physical Review E*, 88(3):032807, 2013.

[125] S. H. Strogatz. Exploring complex networks. *Nature*, 410(6825):268–276, 2001.

[126] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 687–696, 2007.

[127] Y. Sun, R. Barber, M. Gupta, C. C. Aggarwal, and J. Han. Co-author relationship prediction in heterogeneous bibliographic networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 121–128. IEEE, 2011.

[128] M. Szell, R. Lambiotte, and S. Thurner. Multirelational organization of large-scale social networks in an online world. *Proceedings of the National Academy of Sciences*, 107(31):13636–13641, 2010.

[129] L. Tabourier, D. F. Bernardes, A.-S. Libert, and R. Lambiotte. Rankmerging: A supervised learning-to-rank framework to predict links in large social network. *arXiv preprint arXiv:1407.2515*, 2014.

[130] M. Takaffoli, R. Rabbany, and O. R. Zaïane. Community evolution prediction in dynamic social networks. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 9–16, 2014.

[131] F. Tan, Y. Xia, and B. Zhu. Link prediction in complex networks: a mutual information perspective. *PloS One*, 9(9):e107056, 2014.

[132] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[133] J. Tang, T. Lou, and J. Kleinberg. Inferring social ties across heterogenous networks. In *Proceedings of the ACM International Conference on Web Search and Data Mining*, pages 743–752, 2012.

[134] C. Thurau and C. Bauckhage. Analyzing the evolution of social groups in World of Warcraft. In *IEEE International Conference on Computational Intelligence in Games*, pages 170–177, 2010.

[135] H. Tong, C. Faloutsos, and J.-Y. Pan. Fast random walk with restart and its applications. 2006.

[136] S. Van Dongen. A cluster algorithm for graphs. Technical Report 10, Centrum voor Wiskunde en Informatica, 2000.

[137] P. Victor, C. Cornelis, M. De Cock, and A. Teredesai. Trust-and distrust-based recommendations for controversial reviews. In *Web Science Conference*, number 161, 2009.

[138] C. Wang, V. Satuluri, and S. Parthasarathy. Local probabilistic models for link prediction. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 322–331. IEEE, 2007.

[139] D. Wang, D. Pedreschi, C. Song, F. Giannotti, and A.-L. Barabasi. Human mobility, social ties, and link prediction. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1100–1108. ACM, 2011.

[140] X. Wang and G. Sukthankar. Link prediction in multi-relational collaboration networks. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 1445–1447, Niagara Falls, Canada, August 2013.

[141] X. Wang and G. Sukthankar. Link prediction in heterogeneous collaboration networks. In R. Missaoui and I. Sarr, editors, *Social Network Analysis: Community Detection and Evolution*, Lecture Notes in Social Networks, pages 165–192. Springer, 2014.

[142] X. Wang and G. Sukthankar. Link prediction in heterogeneous collaboration networks. In *Social Network Analysis-Community Detection and Evolution*, pages 165–192. Springer, 2014.

[143] N. Warner, M. Letsky, and M. Cowen. Cognitive model of team collaboration: macrocognitive focus. In *Human Factors and Ergonomics Society (HFES) 49th Annual Meeting*, 2005.

[144] R. Wigand, N. Agrawal, O. Osesina, W. Hering, M. Korsgaard, A. Picot, and M. Drescher. Social network indices as performance predictors in a virtual organization. In *Computational Analysis of Social Networks*, pages 144–149, 2012.

[145] J. Xie, M. Chen, and B. K. Szymanski. LabelrankT: Incremental community detection in dynamic networks via label propagation. *arXiv preprint arXiv:1305.2006*, 2013.

[146] J. Xie and B. Szymanski. Towards linear time overlapping community detection in social networks. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, LNAI, 2012.

[147] J. Xie and B. K. Szymanski. Community detection using a neighborhood strength driven label propagation algorithm. In *Network Science Workshop (NSW)*, pages 188–195. IEEE, 2011.

[148] Y. Yang, N. V. Chawla, Y. Sun, and J. Han. Link prediction in heterogeneous networks: Influence and time matters. In *Proceedings of The 12th IEEE International Conference on Data Mining, Brussels, Belgium*, 2012.

[149] M. Ye, P. Yin, and W.-C. Lee. Location recommendation for location-based social networks. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, pages 458–461. ACM, 2010.

[150] N. Yee. The labor of fun: How video games blur the boundaries of work and play. *Games and Culture*, 1(1):68–71, 2006.

[151] J. Zhang, X. Kong, and S. Y. Philip. Predicting social links for new users across aligned heterogeneous social networks. In *2013 IEEE 13th International Conference on Data Mining*, pages 1289–1294. IEEE, 2013.

[152] J. Zhang, X. Kong, and P. S. Yu. Transferring heterogeneous links across location-based social networks. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 303–312. ACM, 2014.

[153] Y. Zhang, J. Wang, Y. Wang, and L. Zhou. Parallel community detection on large networks with propinquity dynamics. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 997–1006, New York, NY, USA, 2009.

[154] T. Zhou, L. Lü, and Y.-C. Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71(4):623–630, 2009.