ABSTRACT

Title of dissertation: COLLECTIVE MULTI-RELATIONAL NETWORK MINING

Seyed Shobeir Fakhraei,
Doctor of Philosophy, 2017

Dissertation directed by: Professor Lise Getoor
Department of Computer Science

Our world is becoming increasingly interconnected, and the study of networks and graphs are becoming more important than ever. Domains such as biological and pharmaceutical networks, online social networks, the World Wide Web, recommender systems, and scholarly networks are just a few examples that include explicit or implicit network structures. Most networks are formed between different types of nodes and contain different types of links. Leveraging these multi-relational and heterogeneous structures is an important factor in developing better models for these real-world networks. Another important aspect of developing models for network data to make predictions about entities such as nodes or links, is the connections between such entities. These connections invalidate the i.i.d. assumptions about the data in most traditional machine learning methods. Hence, unlike models for non-network data where predictions about entities are made independently of each other, the inter-connectivity of the entities in networks should cause the inferred information about one entity to change the models belief about other related entities.

In this dissertation, I present models that can effectively leverage the multi-relational nature of networks and collectively make predictions on links and nodes. In both tasks, I empirically show the importance of considering the multi-relational characteristics and collective predictions. In the first part, I present models to make predictions on nodes by leveraging the graph structure, links generation sequence, and making collective predictions. I apply the node classification methods to detect social spammers in evolving multi-relational social networks and show their effectiveness in identifying spammers without the need of using the textual content. In the second part, I present a generalized augmented multi-relational bi-typed network. I then propose a template for link inference models on these networks and show their application in pharmaceutical discoveries and recommender systems. In the third part, I show that my proposed collective link prediction model is an instance of a general graph-based prediction model that relies on a neighborhood graph for predictions. I then propose a framework that can dynamically adapt the neighborhood graph based on the state of variables from intermediate inference results, as well as structural properties of the relations connecting them to improve the predictive performance of the model.

COLLECTIVE MULTI-RELATIONAL NETWORK MINING

by

Seyed Shobeir Fakhraei

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2017

"Be stubborn about your goals, and flexible about your methods."

–Author Unknown

To my family whose love, support,
strength, patience, and sacrifices
made this journey possible.

Acknowledgments

First and foremost, I am grateful to my adviser Prof. Lise Getoor, who took a chance with me and welcomed me to her group, provided me with opportunities to learn and thrive, and continued to support me in different circumstances. Her support allowed me to take risks, take on challenges and adventures, make mistakes, and learn from them all.

Due to interesting circumstances, I physically spent half of my Ph.D. years at the University of Maryland, College Park (UMD) and the other half at the University of California, Santa Cruz (UCSC), meeting many new friends and colleagues in both places. I would like to thank them all for their friendship and help with my life and research. I would like to especially thank Mohammad Rastegari, Jonghyun Choi, Bert Huang, James Foulds, Jay Pujara, Pigi Kouki, and Dhanya Sridhar who I had the pleasure to collaborate with as co-authors on our papers.

My remote location dynamics would not have been possible without the help of kind administrative staff at both UMD and UCSC. I would like to thank Jennifer Story, Jodie Gray, and Cynthia McCarley for their help in this regard.

I would also like to thank Hal Daumé III for his generous support, kind advice and all the things he thought me, Larry Davis for supporting me in working with his group, Louiqa Raschid for all the great insight on pharmaceutical discoveries, and Hector Corrada Bravo for being on my committee and his constructive feedback.

During my Ph.D. I also had the privilege of working with great mentors and colleagues at several companies and research lab as an intern. I would like to thank

Paul Bennett, Ryen White, and Eric Horvitz at Microsoft Research for their generous time and insightful advice. I would also like to thank Suju Rajan, Nathan Liu, and Yue Shi at Yahoo Labs, Madhusudana Shashanka at ifwe Inc, and the good people of Dato Inc.

Finally, I am heartfully grateful to my parents for their unconditional support and endurance of all the difficulties of the long distance relations. Most important of all, my highest gratitude is to my beloved wife for always being on my side with her trust and support during many uncertain times and her love and sacrifices for me and our children during this journey.

Table of Contents

Preface ii

Dedication iii

Acknowledgments iv

Table of Contents vi

List of Figures viii

1 Introduction 1
- 1.1 Multi-Relational Networks 3
- 1.2 Collective Prediction 4
- 1.3 Summary of Contributions 6

2 Predictions on Nodes 9
- 2.1 Social Spammer Detection 10
  - 2.1.1 Related Work in Spam Detection 13
  - 2.1.2 Our Model 16
    - 2.1.2.1 Graph Structure Features ($\mathbf{X}_{\mathcal{G}}$) 17
    - 2.1.2.2 Sequence-Based Features ($\mathbf{X}_{\mathcal{S}}$) 20
    - 2.1.2.3 Collective Classification with Reports 23
  - 2.1.3 Experimental Validation 25
    - 2.1.3.1 Dataset 26
    - 2.1.3.2 Graph Structure Features Performance 30
    - 2.1.3.3 Sequence-based Features Performance 30
    - 2.1.3.4 Demographic Information Performance 32
    - 2.1.3.5 Collective Classification with Reports 33
  - 2.1.4 Discussion and Conclusion 35

3 Predictions on Links 38
- 3.1 General Augmented Bipartite Structure 39
- 3.2 Proposed General Link Inference Model 43
  - 3.2.1 Similarity-based Reasoning 43
  - 3.2.2 Priors 45

3.2.3 Implementation via HL-MRFs 46
3.3 Drug-Target Interaction Prediction 51
3.3.1 Related Work in Drug-Target Interaction Prediction 54
3.3.2 Our Model 57
3.3.3 Similarity Selection 61
3.3.4 Insights into Collective Link Inference 64
3.3.5 Experimental Validation 67
3.3.5.1 Dataset 68
3.3.5.2 Evaluation Criteria 71
3.3.5.3 Analysis Results 73
3.3.5.4 Predictions Evaluation 80
3.3.6 Discussion and Conclusion 84
3.4 Hybrid Recommender Systems Model 86
3.4.1 Similarity-based Triad Rules 87
3.4.2 Ratings Priors 89
3.4.3 Experimental Validation 91

4 Dynamic Similarity Selection for Link Inference 94
4.1 Neighborhood Graph-based Methods 95
4.2 Neighborhood Graph Construction 97
4.3 General Problem Statement 99
4.4 Proposed General Framework 101
4.5 Methods 103
4.5.1 Nominating Instances 104
4.5.1.1 Disagreement-based Uncertainty 105
4.5.2 Activating Relations 106
4.5.2.1 Value Feature 107
4.5.2.2 Nominated Instance Count Feature 107
4.5.2.3 Observed Instance Count Feature 108
4.5.2.4 Combining Features and Selecting Relations 109
4.6 Neighborhood Graph-based Link Prediction 109
4.7 Experimental Validation 111
4.7.1 Dataset 112
4.7.2 Results 113
4.8 Related Work 114
4.9 Discussion and Conclusion 119

5 Discussion and Conclusion 120
5.1 Future Directions 121

6 Hinge-loss Markov Random Fields 123
6.1 Probabilistic Soft Logic 124
6.2 Inference 125
6.3 Weight Learning 126

Bibliography 127

List of Figures

1.1 Schematic view of predictive tasks in multi-relational networks 3
1.2 Traditional network models versus real-world multi-relational networks. 4
1.3 Independent predictions on unseen data points versus collective predictions. 6

2.1 A time-stamped multi-relation social network with legitimate users and spammers. Each link $\langle v_1, v_2 \rangle$ in the network represents an action (e.g. profile view, message, or poke) performed by v_1 towards v_2 at specific time t. 13
2.2 Social spammer detection framework: It includes graph structure features, action sequence features and model, and a collective model to jointly identify credible accounts and spammers. 17
2.3 Chain-augmented naive Bayes model, for one user. In the diagram, y indicates the label (spammer or not) and x_i represents the ith action performed by the user. 22
2.4 Collective HL-MRFs model to predict spammers based on the reports from other users. 24
2.5 *Precision-Recall* curve and *ROC* for models with k-gram and graph-based features. Using both features with user demographics significantly improve the results. 31
2.6 Simple HL-MRFs models to compare with the collective model shown in Figure 2.4. 34

3.1 Augmented bipartite multi-relational network structures. 41
3.2 Converting different augmented bipartite multi-relational network structures to a single form. 42
3.3 Template model for multi-relational similarity-based link inference, where similar links should have similar values. 44
3.4 Leveraging multi-relational nature of the network and collective predictions in our template model. 45
3.5 Corresponding losses to each template. S(a, d) and L(a, b) indicate SIMILAR(a, d) and LINK(a, b). 49
3.6 Corresponding losses to each prior template. 50
3.7 A schematic overview of a drug-target interaction network. Edges between drugs and between targets represent different similarities. . . 52

3.8 Similar targets tend to interact with the same drug (a), and similar drugs tend to interact with the same target (b). 58
3.9 If one of the drugs interacts with one of the targets, the other drug may interact with the other target as well. 60
3.10 Distribution variation of different similarity values between drugs and between targets. Similarities with values of zero or one are omitted in this plot. 63
3.11 Distribution of annotation-based similarity values for 315 drugs, where dots indicate mean similarity value between each drug and all others, and lines demonstrate standard deviation of the values. Similarities with values of zero or one are omitted in this plot. e.g., the mean of all annotation-based drug similarities with drug #200 is about 0.2 with standard deviation of 0.15. 63
3.12 Predicted interactions can be used for other inferences using target (a) or drug similarities (b), or both. 65
3.13 Network of drug-target interactions in the dataset. Drugs are shown with blue squares and targets with red circles, where size of the node represent their degree. Similarities are not shown to simplify the graph. 68
3.14 Average precision of the top 130 interaction predictions for all 10 folds with $k = 5$. 77
3.15 Relative triad-based rule weights in models with all similarities included and models with only one similarity. 78
3.16 Collective vs. non-collective average precision of the top predictions. . 80
3.17 Comparing Perlman's method with PSL's top 130 predictions using ten-fold cross validation. 82
3.18 Comparing Perlman's method with PSL's top 150 predictions using new interactions. 85
3.19 Recommender systems as an augmented bipartite structured multi-relational graph. 86

4.1 Schematic view of neighborhood graph-based methods. 95
4.2 Neighborhood graph construction from candidate graph with several relations. For simplicity of notation, items are represented only with their labels y. Observed labels y_o are shown as black circles and instances with unobserved labels y_u are shown with white circles and annotated accordingly. Relations $r_{\mathbf{x}_i,\mathbf{x}_j}$ are noted as $r_{i,j}$. The neighborhood graph shown on the right has a subset of selected relations with different types and also the inferred values ($\hat{y}$) for the unobserved labels. 97
4.3 *candidate graph* construction based on the *data graph*. Using the logic shown in (4.9), the similarity between two links that share a node are based on the similarity between their other end nodes. For example, similarity between $\mathbf{x}_p = e^i_{1,2}$ and $\mathbf{x}_q = e^i_{2,3}$ that share v_2 is based on the similarities of v_1 and v_3 which are $e^j_{1,3}$ and $e^k_{1,2}$. 111

4.4 Drug-Target interaction prediction experiments. Horizontal axis show mean and standard deviation (std) of the number of similarities included in the neighborhood graph at each iteration and vertical axis shows the mean and std of the AUPR at each step over ten folds. . . 115

Chapter 1: Introduction

Our world is becoming increasingly connected, and the data collected from many real world phenomena contain an explicit or implicit network structure. The world wide web, social and communication networks, academic and research publications, biological systems and pharmaceutical networks, and healthcare systems are a few examples of such phenomena.

The study of network and graph structures in abstract or for specific applications has been an interesting subject of research for a long time, and predictive models that leverage the network structures have shown to be effective in several domains [1]. However, the real-world networks have two characteristics that most traditional predictive models focused on network domains do not fully leverage; (1) *Heterogeneous and multi-relational nature of the network*, and (2) *the interdependency between the nodes and links.*

Most of the traditional research performed on the graph and network structures assume the nodes and links are of a single type. In this dissertations, I present methods to incorporated the multi-relational and heterogeneous nature of a network in predictive models and show their effectiveness in improving their performance.

Another important aspect of modeling networks for predictive tasks is interdependence between entities (e.g., nodes or links) in the network. Predictive machine learning or data mining models use *training data* to make predictions about the entities with unknown labels. Even most of the network-based predictive models that leverage the connectivity between the *training data* for such tasks make independent predictions about the entities with unknown labels.

The entities with unknown labels are often not only connected to the training data but also connected to other entities with unknown labels. Models that can make *collective predictions* on multiple entities with unknown labels and leverage their connectivity can improve their performances in contrast to those that ignore such characteristics. In this dissertation, I show the effectiveness of accounting for such relations between the entities with unknown labels and propose models that can leverage them in different domains.

Many common predictive tasks in different real-world network-based domains can be modeled as *predictions about nodes* and *predictions about links* [1]. Figure 1.1 shows an schematic view of these two general tasks. Many *predictions about nodes* can be mapped to a classification task. Identifying spammers from non-spammers in a social network is an example of node classification or labeling. In this dissertation, I focus on node classification in Chapter 2 and provide solutions to model multi-relational nature of the network for collective classification.

Predictions about links can be about the existence or absence of a link in the network, or about a value assigned to a link if the link were to exist. Predictions about links that are not already observed in the network can provide information about the probability of interactions between two entities in the future (i.e., link prediction), or the strength of their future relationship (i.e., link regression). A few example domains of link prediction are the connection between two people on a social network, the interaction of a user with a link or an advertisement, the interaction of a drug with a target protein, and a citation from a researcher to another researcher. An example of the relationship strength prediction is in recommender systems, where the goal is to predict how much a person would like a movie or an article. I focus on predictions about links in Chapter 3.

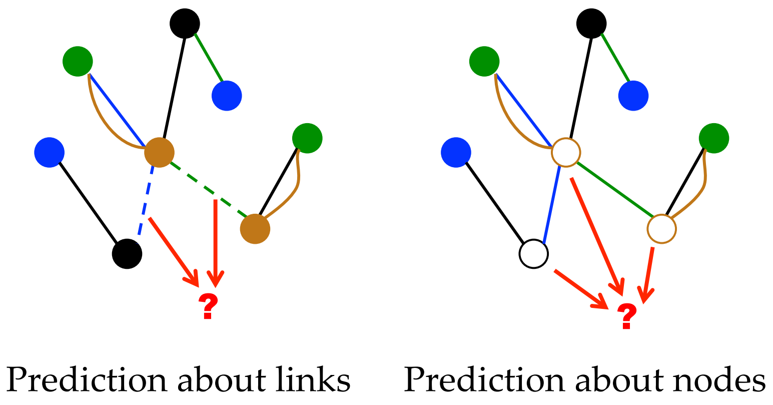


Figure 1.1: Schematic view of predictive tasks in multi-relational networks

The following sections formally define the concepts mentioned above.

1.1 Multi-Relational Networks

Let a multi-relational network or graph be defined as $\mathcal{G} \triangleq \langle \mathcal{V}, \mathcal{E} \rangle$, where $\mathcal{V}$ is a set of vertices and $\mathcal{E}$ is a set of directed or undirected edges. Vertices are identified by their type and a set of features for each type, formally $v_i \in \mathcal{V}$ and $v_i \triangleq \langle t_v, \mathbf{x}_v \rangle$ where $t_v \in \mathbf{t_v}$ shows the type of the vertex and $\mathbf{x}_v$ are a set of features associated with each vertex of this type. Edges are defined by their source and destination vertices, formally $e_i \in \mathcal{E}$ and $e \triangleq \langle v_{src}, v_{dst}, t_e, \mathbf{x_e} \rangle$ where $t_e \in \mathbf{t_e}$ shows the type of the vertex and $\mathbf{x}_e$ are a set of features associated with each edge of this type. Figure 1.2 shows a simple schematic example of traditional graphs where all nodes and edges are of a single type versus a simple example of multi-relational networks where different colors in edges and nodes represent different types.

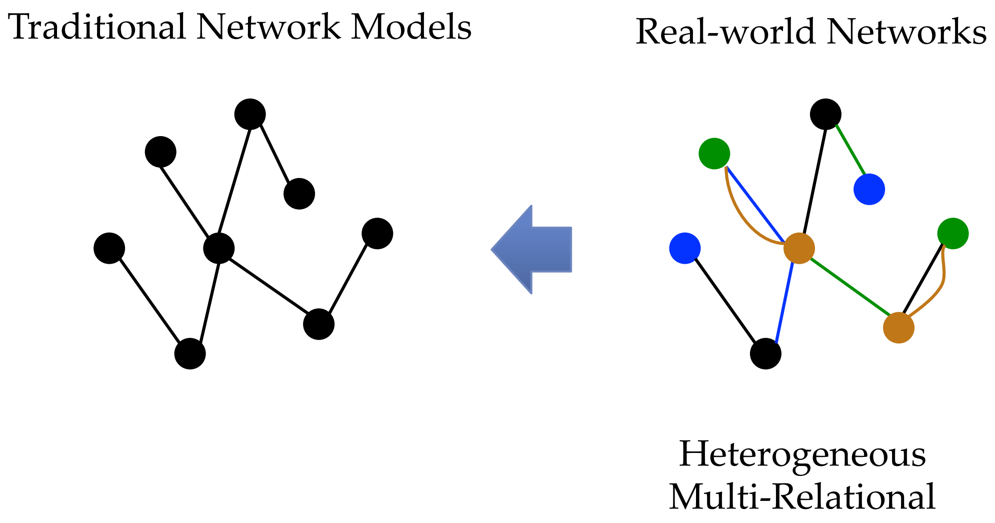


Figure 1.2: Traditional network models versus real-world multi-relational networks.

1.2 Collective Prediction

Collective predictions are based on joint reasoning on a set of interlinked unknown entities where the model's prediction for an entity is affected by the model's predictions about other related entities. For example, observed or inferred information about a persons friends' interests could inform us about that person's interests.

Traditional machine learning approaches tend to make two disjoint sets of train and test instances and assume independence between the instances within and between each set. In other words, they assume that the instances are independent and identically distributed (i.i.d.). However, networks are inherently interconnected and making such assumptions could cause loss of information, and potentially hurt the performance of the model. Figure 1.3 shows an abstract example of traditional machine learning methods where the unseen instances are assumed to be independent and the connection between them are ignored, and an example of collective prediction where the unseen instances are inter-related and the connections between them are leveraged. More formally traditional machine learning methods make predictions via the following method:

$$\hat{y}_i = f(\mathbf{x}_i, \mathbf{y_o}, \mathbf{X_o}; \boldsymbol{\omega}) \tag{1.1}$$

where $\hat{y}_i \in \mathbf{y_u}$ is the predicted label for the ith instance from the *unobserved* set $\mathbf{y_u}$, $\mathbf{x}_i \in \mathbf{X}$ represent features of the ith instance, $\mathbf{X_o}$ and $\mathbf{y_o}$ are the features and labels of instances in the *observed* training set, and $\boldsymbol{\omega}$ represents the model parameters.

Models that perform collective inference [2] based on both known and unknown labels find an optimal state of all unknowns variables by optimizing an objective function g over all unknown labels $\mathbf{y_u}$ and jointly assigning values to all of them. Formally:

$$\hat{\mathbf{y}}_\mathbf{u} = \underset{\mathbf{y_u}}{\arg\operatorname{opt}}\, g(\mathbf{y_u}, \mathbf{y_o}, \mathbf{X}; \boldsymbol{\omega})$$

e.g., in a probabilistic setting:

$$\hat{\mathbf{y}}_{\mathbf{u}} = \arg\max_{\mathbf{y}_{\mathbf{u}}} \mathrm{P}(\mathbf{y}_{\mathbf{u}} | \mathbf{y}_{\mathbf{o}}, \mathbf{X}; \boldsymbol{\omega})$$

where $\mathbf{X}$ are the features for all instances.

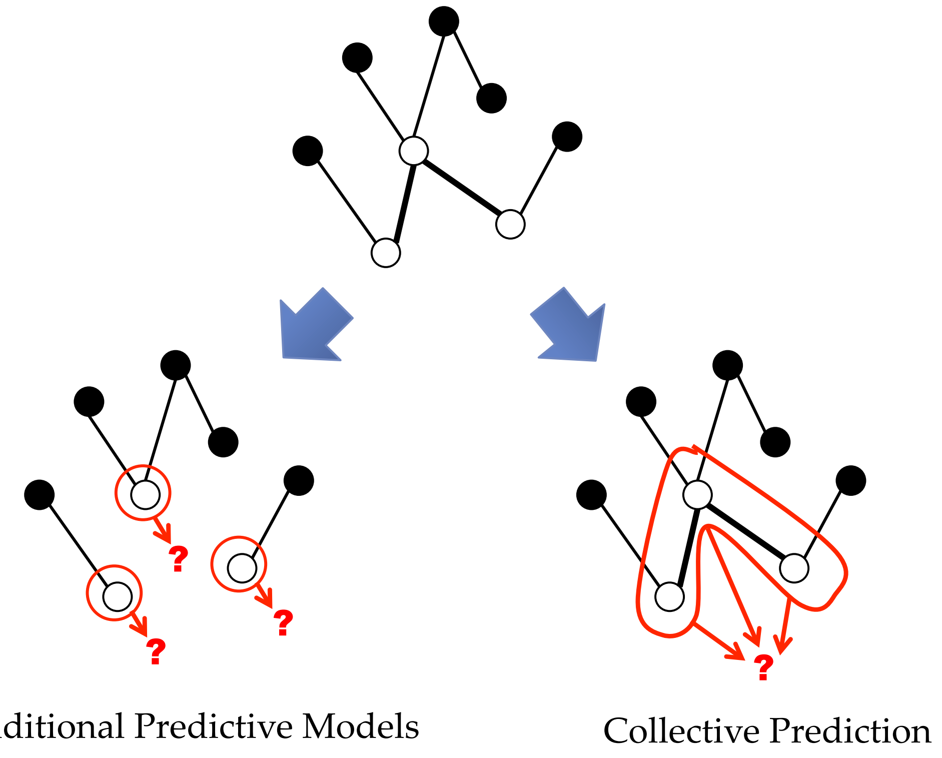


Figure 1.3: Independent predictions on unseen data points versus collective predictions.

1.3 Summary of Contributions

In this dissertation, I propose a set of template models to perform collective predictions on different multi-relational networks and show the importance of both collective predictions and consideration of multi-relational characteristics in predictive modeling of networks. I focus on link prediction and regression and node classification.

First, I propose three methods for node classification in a multi-relational network; I motivate my methods and empirically show their effectiveness in a social network with several types of time-stamped edges between users. I use the multi-relational structure of the network and sequential properties of the network formation to predict spammers, and show the effectiveness of collective propagation of reputation for this task. Using these approaches I show the effectiveness of leveraging the multi-relational nature of the network and collective classification of the nodes.

I then show that data from different domains can be modeled via a common bipartite network and present a set of template models for predicting information about the links in this common augmented bipartite network structure to support multi-relational characteristics of the network and make collective predictions. I show the effectiveness of the template model in pharmaceutical and recommender systems domains and outperform the state-of-the-art methods in both domains via my general link inference framework.

Finally, I show that the collective link prediction task is an instance of a general graph-based prediction model that relies on a *neighborhood graph* for predictions, and highlight the challenges of using a common method of pre-processed *neighborhood graph* selection. I propose a framework that can dynamically adapt the *neighborhood graph* based on the state of variables from intermediate inference results, as well as structural properties of the relations connecting them to improve the performance of the model.

Chapter 2: Predictions on Nodes

Many predictive tasks in networks can be modeled as node classification. Identifying the influential users in a social network, inferring the political affiliation of a user, finding users with fraudulent or malicious behaviors, and predicting a protein's function are examples in this category.

In this chapter, I propose models for node classification based on a motivating example of spammer detection in a social network. Social networks are generally multi-relational and evolve over time. A social network can be represented as a directed time-stamped multi-relational graph $\mathcal{G} \triangleq \langle \mathcal{V}, \mathcal{E} \rangle$, where $\mathcal{V}$ is the set of vertices of the form $v_i \in \mathcal{V}$, and $v_i \triangleq \langle \mathbf{x}_v \rangle$ representing users and their demographic features $\mathbf{x}_v$, and $\mathcal{E}$ is the set of directed edges of the form $e_i \in \mathcal{E}$ and $e \triangleq \langle v_{src}, v_{dst}, t_e, x_t \rangle$ representing their interactions, relation type t_e, and a time-stamp x_t.

The social spammer detection task is to classify node v_i with an unknown label to spammer or not spammer, based on the given network $\mathcal{G}$ and a set of observed labels for already identified spammers and not spammers. We are interested in assigning a score (e.g., a probability) to each user, in order to rank them from the most to the least probable spammer in the system: $c : v_i \rightarrow [0, 1]$.

In this chapter, I propose three approaches for node classification based on different characteristics of the network; *the network static structural features*, *the sequence of edge generation*, and *collective classification using user credibility.* Using these characteristics, I empirically show the effectiveness of leveraging the multi-relational nature of the network both in methods based on static structure of the network and the edge generation sequence. Then, using the third approach, I empirically show the effectiveness of collective classification compared to independent predictions about the nodes.

The following sections describe the task motivation, related work in this area, the details of the proposed methods, and the experimental validations.

2.1 Social Spammer Detection

Unsolicited or inappropriate messages sent to a large number of recipients, known as "spam", can be used for various malicious purposes, including phishing and virus attacks, marketing of objectionable materials and services, and compromising the reputation of a system. From printed advertisements to unsolicited phone calls, spam has been a perennial problem in modern human communication. With the emergence of the Internet, spammers have found a cost-effective medium to reach a broader audience than was previously possible. Email spam is almost as old as the Internet itself. The first email spam was sent in 1978 to all several hundred users of ARPANET [3].

More recently, social media has given spammers a new and effective medium to spread their content. Using social media platforms, spammers can disguise themselves as legitimate users and engage in realistic looking interactions. They can use these platforms to send messages to users, leave spam comments on popular pages, and reply to legitimate comments using spam content. Such diversity of choice has often increased spammers' ability to conceal their intentions from traditional spam filters. According to a study by Nexgate [4], social spam grew by more than 355% between January to July of 2013, one in 200 social messages contain spam, and 5% of all social apps are spammy.

While content-based approaches have been shown to be effective in stopping spam in email and the web, they can be manipulated by sophisticated spammers via incorporating content randomness. Unlike in email and the web, social media enables spammers to split their content across multiple messages in order to bypass spam filters. Link-based approaches that leverage the connectivity of the entities, have been combined with content-based methods to build more effective methods. While it is easier to pass traditional content-based filters, behavioral patterns and graph properties of the users' interactions are harder to manipulate. Furthermore, many social networks can not monitor all the generated content due to privacy and resource concerns. Content-independent frameworks, such as the one proposed in this chapter, can be applied to systems that provide maximum user privacy with end-to-end encryption.

Perhaps the most important difference between social networks and email or web graphs is that social networks have a multi-relational nature, where users have relationships of different types with other users and entities in the networks. For example, they can send messages to each other, add each other as friends, "like" each other's posts, and send non-verbal signals such as "winks" or "pokes." Figure 2.1 shows a representation of a social network as a time-stamped multi-relational graph. The multi-relational nature provides more choices for spammers, but it also empowers detection systems to monitor patterns across activity types, and time. In this chapter, we propose a content-independent framework which is based on the multi-relational graph structure of different activities between users, and their sequences.

Our proposed framework is motivated by data from *Tagged.com*, a social network for meeting new people which was founded in 2004 and has over 300 million registered members. However, the framework is applicable to any multi-relational social network. Our goal is to identify sophisticated spammers that require manual or semi-automated intervention by the administrative security team. These spammers have already passed initial classifiers and know how to manipulate their accounts and contents to avoid being caught by automatic filters. We show that our framework significantly reduces the need for manual administration to control spam.

Our framework consists of three components. First, we extract graph structure features for each of the relations and show that considering the multi-relational nature of the graphs improves the performance. Second, we consider the activity sequence of each user across these relations and extract k-gram features and employ mixtures of Markov models to label spammers. Third, we propose a statistical

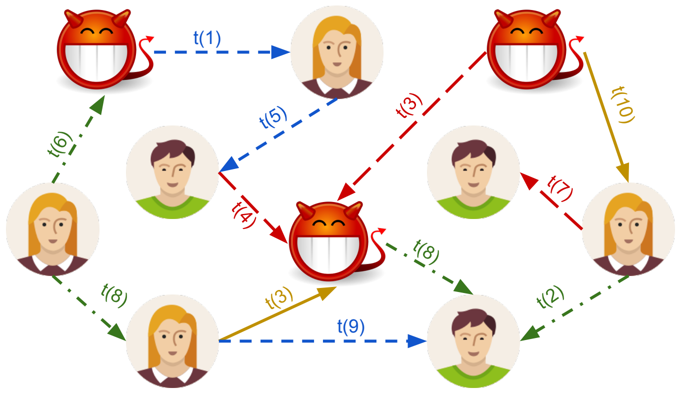


Figure 2.1: A time-stamped multi-relation social network with legitimate users and spammers. Each link $\langle v_1, v_2 \rangle$ in the network represents an action (e.g. profile view, message, or poke) performed by v_1 towards v_2 at specific time t.

relational model based on hinge-loss Markov random fields to perform collective reasoning using signals from an abuse reporting system in the social network.

The following sections formally define the problem and our solution framework along with an experimental validation of our approach on internet-scale data from *Tagged.com*.

2.1.1 Related Work in Spam Detection

Spam detection in email [5] and the web [6] have been extensively studied, and various methods and features have been proposed for them. Network-based approaches are more closely related to our proposed framework. These methods can be generally categorized based on feature construction and label propagation. Shrivastava et al. [7] generalized the network-based spam detection to random link attacks and showed that the problem is NP-complete. Tseng and Chen [8] used network features to identify email spammers, and incrementally updated the SVM classifier to capture the changes in spam patterns. Oscar and Roychowdbury [9] used a network representation of the emails where nodes were email addresses and links between them indicated a sender-receiver relationship. They used clustering properties of the network to build white and black lists of email addresses and identify spammers. Becchetti et al. [10] proposed a link-based classification for web spam detection, and later combined it with content-based features and used graph topology to improve performance [11]. Since spammers tend to form clusters on the web (unlike in social networks), the authors leveraged clustering and label propagation, to further improve their predictions.

A group of methods are based on label propagation and influenced by PageRank. TrustRank [12] for example, used reputable sites as seeds and propagated reputations through the network. There are multiple variations which propagate dis-trust. Similar to this work, Chirita et al. [13] proposed MailRank which ranked the trustworthiness of a sender based on the network representation of the mail environment. Abernethy et al. [14] proposed a method based on graph regularization and used regularizers that is based on the intuition that linked pages are somewhat similar.

The research focus on spam detection in social networks is relatively more recent. Heymann et al. [15] surveyed different countermeasures to address the spam issue in social networks, and categorized them into methods based on detection, demotion, and prevention. Hu et al. [16] combined information from email, text messages (SMS), and web with Twitter content to detect spammers, and showed improvements in results. Tan et al. [17] proposed an unsupervised spam detection method that focused on identifying a white list of non-spammers from the social network. They argued that legitimate users show more stable patterns in social blogs.

Stein et al. [18] described the spam filtering system in Facebook. They highlighted that attacks on social media use multiple channels, and an effective systems must share feedback and feature data across channels. Gao et al. [19] studied messages between users in Facebook, and used clustering to detect spam campaigns. They identify multiple clusters associated with several campaigns.

Markines et al. [20] studied multiple features and classifiers to detect spam in social tagging systems. Benevenuto et al. [21] used content such as presence or absence of a URL in the post, and user social behaviors such as number of posts to detect spam on Twitter. Lee et al. [22] used honeypots in Twitter and MySpace to harvest deceptive spam profiles. They then used content, posting rate, number of friends, and user demographics such as age and gender as features in their classifier.

Zhu et al. [23] reported that unlike email and web, in social networks, spammers do not form clusters with other spammers, and their neighbors are mostly non-spammers. They use matrix factorization on user activity matrix of data extracted from Renren[1] and use the latent factors as features for classification.

Evolving social networks are of high interest to researchers and have been studied for different purposes [24]. Jin et al. [25] modeled a social network as a *time-stamped heterogeneous network* and used a clustering method to identify spammers. They also used active learning to refine their model. Zhang et al. [26] identified

[1] A social network in China: `http://renren.com`

spam campaigns on Twitter by linking accounts with similar malicious URLs in their posts.

Laorden et al. [27] used collective classification to filter spam messages based on their text, to reduce the number of necessary labeled messages. They used implementations in WEKA for collective classification in their evaluation. Geng et al. [28] used a semi-supervised learning algorithm to reduce the labeled training data requirement for web spam detection. Torkamani and Lowd [29] proposed a method to robustly perform collective classification against malicious adversaries that change their behavior in the system.

2.1.2 Our Model

In our framework, we focus on three different mechanisms to identify spammers and malicious activities. We first create networks from the user interactions and compute network structure features from them. As these are evolving networks, each user generates a sequence of actions with the passage of time. Mining these sequences can provide valuable insights into the intentions of the user. We use two methods to study these sequences and extract features from them. We use the output of these methods as features to classify spammers. We then employ a collective model to identify spammer accounts only based on the signals from the abuse reporting system ($\mathcal{G}_{\text{report}}$) as a secondary source to reassure predictions. Figure 2.2 shows an overview of the framework we have proposed for this problem and the following sections discuss our framework and extracted features in more details.

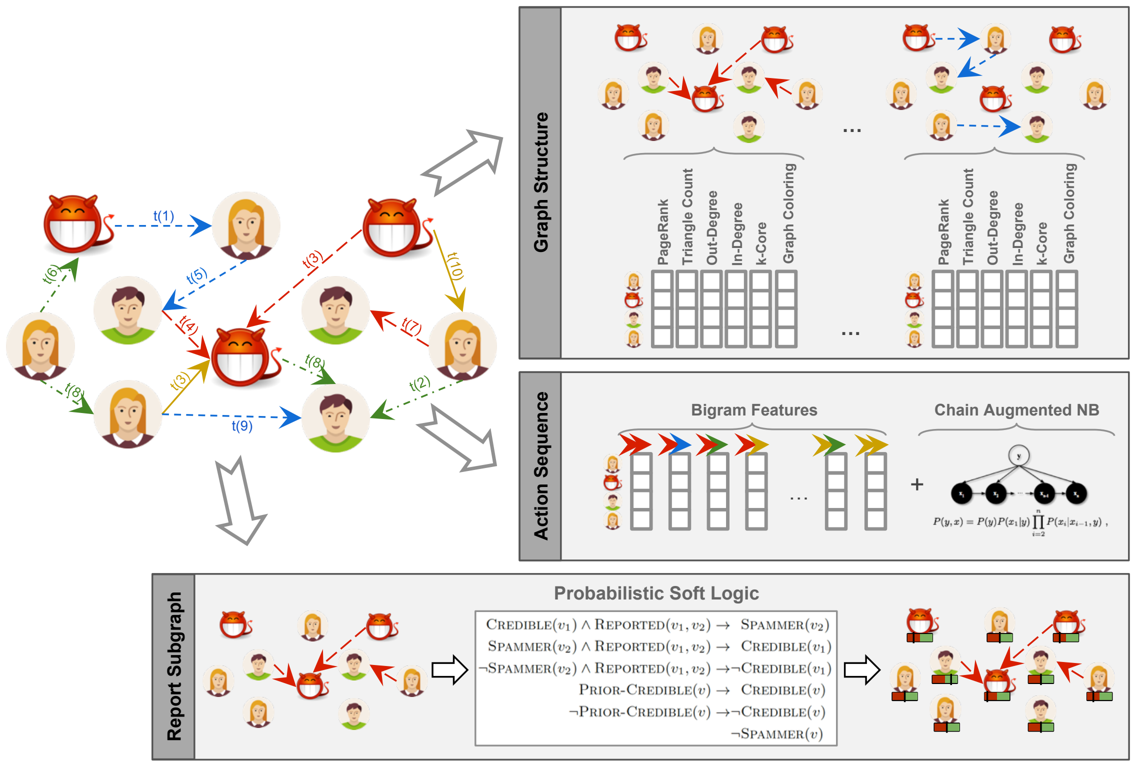

Graph Structure
PageRank
Triangle Count
Out-Degree
In-Degree
k-Core
Graph Coloring
Action Sequence
Bigram Features
Chain Augmented NB
$P(y,x) = P(y)P(x_1|y)\prod_{i=2}^{n} P(x_i|x_{i-1},y)$
Report Subgraph
Probabilistic Soft Logic
CREDIBLE(v_1) ∧ REPORTED(v_1,v_2) → SPAMMER(v_2)
SPAMMER(v_2) ∧ REPORTED(v_1,v_2) → CREDIBLE(v_1)
¬SPAMMER(v_2) ∧ REPORTED(v_1,v_2) →¬CREDIBLE(v_1)
PRIOR-CREDIBLE(v) → CREDIBLE(v)
¬PRIOR-CREDIBLE(v) →¬CREDIBLE(v)
¬SPAMMER(v)
t(1)
t(2)
t(3)
t(4)
t(5)
t(6)
t(7)
t(8)
t(9)
t(10)

Figure 2.2: Social spammer detection framework: It includes graph structure features, action sequence features and model, and a collective model to jointly identify credible accounts and spammers.

2.1.2.1 Graph Structure Features ($\mathbf{X}_{\mathcal{G}}$)

We create a directed graph $\mathcal{G}_r = \langle \mathcal{V}, \mathcal{E}_r \rangle$ for each relation r in the social network, where vertices $\mathcal{V}$ consist of users, and edges $\mathcal{E}_r$ represent interactions of type r between users, e.g. if user$_1$ sends a message to user$_2$ then $\mathcal{G}_{\text{message}}$ will contain v_1 and v_2 representing the two users, and $e_{1,2}$ representing the relation between them.

We have ten different graphs each containing the same users as vertices but different actions as edges.

We use six graph analytics methods (m_i) to compute the features. Using each m_i we create a set of features for each relation graph $\mathcal{G}_r$ as following:

$$\mathbf{X}^{m_i}_{\mathcal{G}_\mathbf{r}} = \left[\begin{array}{ccc} \mathbf{X}^{m_i}_{\mathcal{G}_{r_1}} & \dots & \mathbf{X}^{m_i}_{\mathcal{G}_{r_n}} \end{array} \right]$$

where m_i is one of the graph analytics methods described below, r_i is one of the relationships considered in the study, and $\mathbf{X}^{m_i}_{\mathcal{G}_r}$ is the matrix corresponding to all the features generated using the m_i method.

We then use these features together to get a complete multi-relational graph feature-set, as the following:

$$\mathbf{X}^{\mathbf{m}}_{\mathcal{G}_\mathbf{r}} = [\begin{array}{ccc} \mathbf{X}^{m_1}_{\mathcal{G}_\mathbf{r}} & \dots & \mathbf{X}^{m_k}_{\mathcal{G}_\mathbf{r}} \end{array}]$$

The graph analytics methods (m_i) we use to extract the features from each relation network are described below. Each of these algorithms provieds different perspectives on the local connectivity of the graph and neighborhood characteristics of each user. Our goal is to capture the structural differences between spammers' and legitimate users' multi-relational neighborhood graph.

PageRank: PageRank [30], is a well known ranking algorithm proposed for ranking websites, and computes a score for each node by considering the number and quality of links to a node. The algorithm is based on the underlying assumption that important nodes receive more links from other nodes.

Degree: We compute the total degree, in-degree, and out-degree of each node for each relation, which correspond to the total number of activities a user has been involved in, the number of communications (or actions) a user received, and the number of actions the user performed.

***k*-core:** k-core [31] is a centrality measure that is based on the graph decomposition via a recursive pruning of the least connected vertices. The value each vertex receives depends on the step in which the vertex is eliminated from the graph. e.g, vertices removed on the third iteration receive the value three.

Graph Coloring: Graph coloring [32] is an assignment of colors to elements (here vertices) of a graph, such that no two adjacent vertices share the same color. Using a greedy implementation, we obtain the color identifier of each vertex as a feature.

Connected Components: A connected component [33] is a group of vertices with a path between each vertex and all other vertices in the component. A weakly connected component is a maximal set of vertices such that there is an undirected path between any two vertices in the set. We compute the weakly connected component on each graph and extract the component identifier and size of the component that the vertex participates in as features.

Triangle Count: The triangle count [34] of a vertex is the number of triangles (a complete subgraph of three vertices) in the graph the vertex participates in. Such number is an indication of the connectivity of the graph around that vertex.

2.1.2.2 Sequence-Based Features ($\mathbf{X}_{\mathcal{S}}$)

Sequence classification is used in many domains, including biology and health-informatics, anomaly detection, and information retrieval [35]. In dynamically evolving multi-relational social networks, each user v_i generates a sequence of edges via their actions as the following:

$$\mathcal{S}^{v_i} = \langle r_p, \ldots, r_q \rangle$$

Our hypothesis is that spammers are typically bots and follow specific actions in the network, and it is likely that their sequence of actions diverge from the norm. In this section we study these sequences and provide two different solutions for classifying users based on their activity sequences. It is important to note that such an approach would not be possible if the network were not multi-relational.

Sequential *k*-gram Features: The simplest way to represent a sequence with features is to count each element in the sequence independently. However, the order of the sequence cannot be captured with this approach. Furthermore, in our scenario the values of these features will be the same as the out-degree for each vertex, which we previously computed in the graph-based features. A better approach would be to use a short sequence segment of k consecutive actions, called a k-gram to capture the order of events [35]. The sequence can be represented as a vector of the frequencies of the k-grams. To keep the feature space computationally manageable we chose bigram sequence features where $k = 2$. For example, the number of times a user v_i sent a *message* after performing a *profile view*, would be the value for the feature $x^{v_i}_{\text{profileview-message}}$. The bigram feature set for the sequence $\mathcal{S}$ will be the following:

$$\mathbf{X}_{\mathcal{S}_B} = \left[\, \mathbf{X}_{r_1 r_1} \; \ldots \; \mathbf{X}_{r_p r_q} \; \ldots \; \mathbf{X}_{r_n r_n} \, \right]$$

where r_i is one of the relationships considered in the study, $\mathbf{X}_{r_p r_q} = \left[x^{v_1}_{r_p r_q} \ldots x^{v_m}_{r_p r_q} \right]^\intercal$, and $x^{v_i}_{r_p r_q}$ is the total number of times user v_i performed an action of type r_q consecutively after performing r_p.

Chain-augmented Naive Bayes Model: While k-gram features capture some aspects of the order of elements in the sequence, they may miss patterns in longer sequences. Increasing k will rapidly increase the feature space, introducing computational barriers and estimation challenges due to feature sparsity. Instead, to capture the salient information from longer sequence chains, and to study the predictive power of this information, we construct a simple generative model for sequence data. The model is equivalent to the chain-augmented naive Bayes model of [36], a special case of the tree-augmented naive Bayes model [37] which has been shown to be effective in language modelling. The model posits that each user's actions are generated via a mixture of Markov models. In more detail, each class (spammer or not spammer) is associated with a mixture component y. Conditional on the class (mixture component) y for a user, that user's sequence of actions are assumed to be generated from a Markov chain specific to that class. The joint probability for a user's class y and action sequence $x_i, \ldots, x_n$ is given by

$$P(y, x) = P(y)P(x_1|y)\prod_{i=2}^{n} P(x_i|x_{i-1}, y) \ ,$$

which we summarize with a directed graphical model diagram in Figure 2.3. We place symmetric Dirichlet priors on the parameters of the discrete distributions $P(y)$, $P(x_1|y)$, and $P(x_i|x_{i-1}, y)$, and compute maximum a posteriori (MAP) estimates of them, which are readily obtained as the proportion of each outcome in the training data, with the counts first adjusted by adding the Dirichlet smoothing parameter $\alpha = 1$. Finally, at test time we compute the posterior probability of the user's class label given the observed action sequence x via Bayes rule, $P(y|x) \propto P(x|y)P(y) = P(y, x)$.

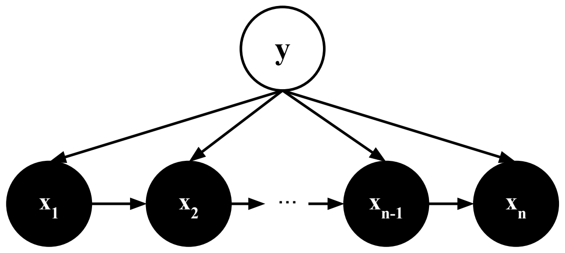


Figure 2.3: Chain-augmented naive Bayes model, for one user. In the diagram, y indicates the label (spammer or not) and x_i represents the ith action performed by the user.

There are multiple methods to incorporate the predictions from this model into our framework. We simply use the ratio of posterior probabilities and their logarithmic forms as a small feature-set ($\mathbf{X}_{\mathcal{S}_M}$) for our classifier.

2.1.2.3 Collective Classification with Reports

Most websites that enable users to publish content also provide an *abuse reporting* mechanism for other users to bring malicious behavior to the system's attention. However, these systems do not necessary offer clean signals. Spammers themselves often randomly report other users (spammers and legitimate users) to increase the noise, legitimate users often have different standards for malicious behaviors, and users may report others for personal gains such as censorship or blocking an opponent in a (social) game from accessing the system. A model that can extract sufficient information from the relational *report* feature, can enhance the administrative team's performance by focusing their attention, and can also provide an additional feature or parallel mechanism for spam classification.

We propose a model based on Hinge-loss Markov Random Fields (HL-MRFs) [38] to collectively classify spammers within the reported users, and assign credibility scores to the users offering feedback via the reporting system. Using this model a better ranking of the reported users based on their probability of being spammers can be provided to the security administration team. The hinge-loss formulation has the advantage of admitting highly scalable inference, regardless of the structure of the network.

HL-MRFs Collective Model for Reports: The goal of this model is to use reports to identify spammers. We study three HL-MRFs models to incorporate the reporting users' credibility into the reporting system and improve the predictability of the reports. We show that collective reasoning over credibility of the reporting user and the probability of the reported user being an spammer, increases the classification performance of the system.

Our collective HL-MRFs model uses the *report* relation graph ($\mathcal{G}_{\text{report}}$), and is based on the intuition that the credibility of a user's abuse reporting should increase when they report users that are more likely to be spammers. Hence, if a user reports other users whom there are other evidence supporting them being spammers, the credibility of that person should increase. On the other hand, if the user reports another user that is unlikely to be a spammer, the credibility of the reporting user should decrease.

$$\begin{aligned}
\textsc{Credible}(v_1) \land \textsc{Reported}(v_1, v_2) \rightarrow\ & \textsc{Spammer}(v_2)\\
\textsc{Spammer}(v_2) \land \textsc{Reported}(v_1, v_2) \rightarrow\ & \textsc{Credible}(v_1)\\
\neg\textsc{Spammer}(v_2) \land \textsc{Reported}(v_1, v_2) \rightarrow & \neg\textsc{Credible}(v_1)\\
\textsc{Prior-Credible}(v) \rightarrow\ & \textsc{Credible}(v)\\
\neg\textsc{Prior-Credible}(v) \rightarrow & \neg\textsc{Credible}(v)\\
& \neg\textsc{Spammer}(v)
\end{aligned}$$

Figure 2.4: Collective HL-MRFs model to predict spammers based on the reports from other users.

We propose the model shown in Figure 2.4 to capture the collective intuition. We incorporate prior credibility of the reporting users based on the past reporting behavior into the model. The negative prior on *SPAMMER* is included in the model to complement the first rule that increases the score of users being spammers. To study the effect of each part of the model, we experimentally compare the proposed collective model with two simpler HL-MRFs models that do not contain the collective reasoning and credibility priors in section 2.1.3.5.

2.1.3 Experimental Validation

We performed four sets of experiments to evaluate the proposed methods. First we study the graph structure properties and compare the multi-relational approach with only considering a single relation. We also study using one graph analytics algorithm as a feature, comparing to having features from multiple methods. We then study the effectiveness of sequence mining features and combine them with graph-based methods to measure the overall performance enhancements. We then include only three demographics features for each user to measure their influence on the performance. Finally we perform collective reasoning over *abuse reports* and measure the improvement of the predictions with this method.

For our experiments we used *Graphlab Create^{TM}* and the Java-based open-source *Probabilistic Soft Logic (PSL)*,[2] on a single Ubuntu machine with 32GB RAM and 3.2GHz CPU (4 cores). For classification, we used *Gradient-Boosted Decision Trees* which is a collection of decision trees combined through a technique called gradient boosting [39].

The deployment options of the framework and what actions are planned to be taken on the identified spammer accounts determine which performance metrics are

[2]`http://psl.umiacs.umd.edu`

more appropriate for this task. High precision lets the spam accounts be blocked without manual intervention, and without concerns of the system harming legitimate users. High recall allows the system to identify the legitimate users with more confidence and clear the environment via deploying measures such as CAPTCHA and additional account verifications for the users with suspicious status. Hence, the appropriate metric to measure the performance of this system is the *Precision-Recall* curve. The *ROC* curve could also be useful, however, due to the high class-imbalance, it would not provide much insight, and unless properly adjusted, it would result in over-optimistic estimates. We report the area under *Precision-Recall* curve (AUPR) and the area under the *ROC* curve (AUROC) for the experiments. We used 10-fold cross-validation to estimate the performance of each method and feature-set. Unless stated otherwise, the reported numbers represent *mean* and *standard deviation* over 10-fold cross-validation.

2.1.3.1 Dataset

The dataset[3] was collected from the *Tagged.com* social network website, which is a network for meeting new people, and has multiple methods for users to make new connections. *Tagged* has various methods to deal with spam. It uses several registration and activation filters to identify and block spam accounts based on traditional methods such as content and registration information and patterns. *Tagged* also employs a reporting mechanism that users can report spammers to the system.

[3]An anonymized sample of the multi-relational part of the dataset along with our code for the experiments can be found here: `http://github.com/shobeir/fakhraei_kdd2015`.

An administrative security team monitors the network for malicious behaviors and manually blocks spammers, which we used as ground truth labels. Our goal in this study is to identify sophisticated spammers that require manual intervention by the security team. These spammers have already passed initial classifiers and know how to manipulate their content to avoid being caught by automatic filters.

The purpose of the social network affects its susceptibility to spam. A social network which is designed for connecting the users who already know each other, can control spam by limiting the communications between users who are not already connected in the network. However, a social network that promotes finding new connections may like to impose minimum limitations on how users interact. *Tagged*, which is a social network for meeting new people, has multiple venues for users to communicate without much restriction.

Another challenge with identifying spammers in multi-purpose social networks such as *Tagged* is that users join the network for different reasons. For example, users may come to *Tagged* to play social games such as *Pets* and *MeetMe*, to find romantic relationships, or simply to spend time with virtual connections. Not only they will generate different behavioral patterns, they will use security measures such as *abuse reporting* mechanism differently and introduce noise to it.

In our experiments, all the users who had at least one activity in the sampling time frame were included in the sample dataset. More formally our initial sample included the following elements:

$$\begin{aligned}\mathcal{V} &= \{\ v \mid \exists\ e = \langle v, v_*, r_*, t_k \rangle \in \mathcal{E}_{\text{all}}\ \wedge\ t_{\text{b}} \leq t_k \leq t_{\text{e}}\} \\ \mathcal{E} &= \{\ e = \langle v_i, v_j, r_*, t_k \rangle\ \mid \exists\ v_i, v_j \in \mathcal{V}\ \wedge\ t_{\text{s}} \leq t_k \leq t_{\text{e}}\}\end{aligned}$$

where v_* indicates any user in the network, r_* indicates any type of action in the study, $\mathcal{E}_{\text{all}}$ indicates all the edges in the *Tagged* network, and t_{b} and t_{e} indicate the time of the beginning and the end of the sampling period.

To perform a retrospective study, we chose t_{b} and t_{e} such that enough time had passed since the sampling period by the time we accessed the data (t_{access}), so that most of the spam accounts were identified and labeled. We then removed the users who had deactivated their accounts themselves by t_{access}, because we could not determine their labels. The remaining users were labeled as spam if their accounts has been manually canceled by a security team by t_{access}. Although the security team cancels accounts for multiple reasons, not just spam, most of the canceled accounts are due to malicious activities. For simplicity, we labeled all the canceled accounts as spammers. Ten different activities on the website were selected during the sampling time frame. The activities included in the study are: viewing another user's profile, sending friend requests, sending messages, sending *luv*, sending *winks*, buying or wishing others in the *Pets* game, clicking yes or no in the *MeetMe* game, and reporting other users for abuse.

There are more effective ways to sample the network in order to conserve its characteristics [40, 41, 42]. However, for practical reasons and ease of deployment, we have chosen the simple time-based sampling method. Further performance improvements may be achieved via better sampling employments. The spammer accounts that were selected for this study could initially bypass *Tagged* deployed preventative measures and successfully perform at least one action in the network. Although they could be identified within a short period of time after their activity, their identification required a manual or semi-automated procedure by the members of the security team. Not only are these spammers harder to identify, they are also very rare in the dataset, causing a huge class imbalance.

Table 2.1 shows some statistics from the sample we used. These numbers do not represent the statistics of the *Tagged* social network, as they have been altered by limiting the number of action types in the study as well as eliminating users with deactivated accounts at t_{access} (which is later than the sample period). Furthermore, only the users who performed an action in the sampling period were included in the dataset.

Table 2.1: Data Sample Statistics.

Entity	**Count**
$\lvert\mathcal{V}\rvert$ (total users)	5,607,454
$\lvert\mathcal{E}\rvert$ (total actions)	912,280,409
$\max(\lvert\mathcal{E}_r\rvert)$ (number of actions that are most frequent action type)	350,724,903
$\min(\lvert\mathcal{E}_r\rvert)$ (number of actions that are least frequent action type)	137,550
total users labeled as spammers	(%3.9) 221,305

All of our experiments are based on the relational data in the following form:

$$\langle t_i, v_{\text{src}}, v_{\text{dst}}, r_j \rangle$$

where t_i is the time stamp, v_{src} is the user who initiated the action, v_{dst} is the user the action was towards, and r_i categorizes the type of action.

2.1.3.2 Graph Structure Features Performance

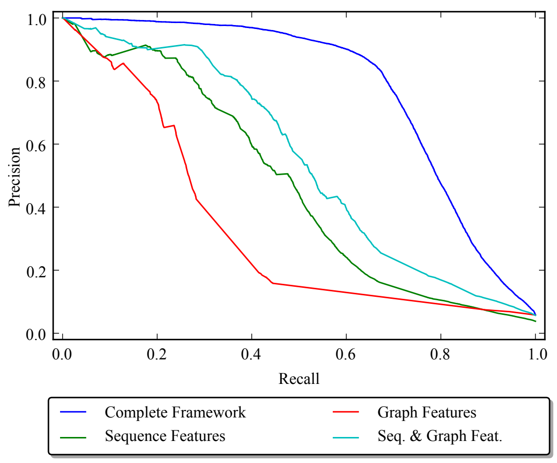
Table 2.2 shows the average results of classification via graph-based features. The first row indicated the best results from using a single relation with features from all the graph-based algorithms. The second row shows the best graph-based feature with all the relations. Comparing the results from the two rows suggests that combining different relations is more effective than combining features from different algorithms on a single relation. Using all algorithms to compute features on all relation graphs results in the best performance for graph-based methods.

Table 2.2: Classification with graph-based features.

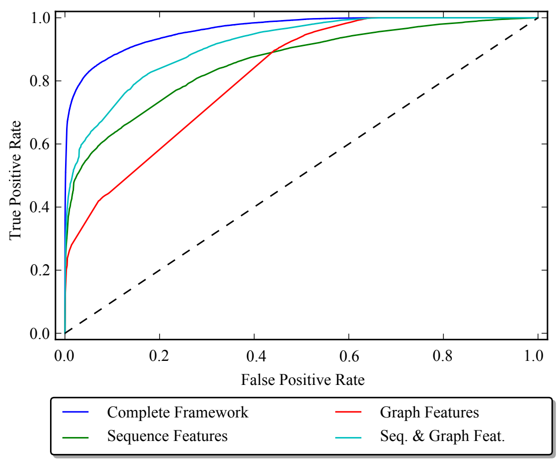
Experiment		**AUPR**	**AUROC**
$\left[\mathbf{X}^{\mathbf{m}}_{\mathcal{G}_{r_i}}\right]$	1 Relation, k Methods	$0.187_{\pm 0.004}$	$0.803_{\pm 0.001}$
$\left[\mathbf{X}^{m_i}_{\mathcal{G}_{\mathbf{r}}}\right]$	n Relations, 1 Method	$0.285_{\pm 0.002}$	$0.809_{\pm 0.001}$
$\left[\mathbf{X}^{\mathbf{m}}_{\mathcal{G}_{\mathbf{r}}}\right]$	n Relations, k Methods	$0.328_{\pm 0.003}$	$0.817_{\pm 0.001}$

2.1.3.3 Sequence-based Features Performance

Next, we experimentally evaluated the sequence-based features. First, we study their effectiveness independently, and then we measure their performance in combination with the graph-based features. To compute the bigram features, we first sorted all of the activities in our dataset based on user IDs and timestamps via



(a) Precision-Recall Curve



(b) ROC Curve

Figure 2.5: *Precision-Recall* curve and *ROC* for models with k-gram and graph-based features. Using both features with user demographics significantly improve the results.

the standard external sort function in Linux. We did a single pass on the sorted file to compute the bigram features.

Table 2.3: Classification with k-gram features.

Experiment	**AUPR**	**AUROC**
$\begin{bmatrix}\mathbf{X}_{\mathcal{S}_B}\end{bmatrix}$ k-gram features	$0.471_{\pm 0.004}$	$0.859_{\pm 0.001}$
$\begin{bmatrix}\mathbf{X}_{\mathcal{S}_B} & \mathbf{X}^{\mathbf{m}}_{\mathcal{G}_\mathbf{r}}\end{bmatrix}$ k-gram & graph features	$0.543_{\pm 0.005}$	$0.914_{\pm 0.001}$

Table 2.3 shows the results of classification using the bigram features. The second row suggests that a model that uses both graph-based and k-gram features outperforms the ones that use them independently. *Precision-Recall* and *ROC* curves from graph-based and k-gram features are shown in Figure 2.5.

We further study the sequence-based classification with the Mixture of Markov Models (MMM) approach. We did a single pass on the sorted file we already generated for the bigram features to compute the probabilities for this model. We then used the probabilities generated from this model in logarithmic and ratio forms as features for classification.

Table 2.4: Classification with mixture of Markov models.

Experiment	**AUPR**	**AUROC**
$\left[\mathbf{X}_{\mathcal{S}_M}\right]$ MMM	$0.246_{\pm 0.009}$	$0.821_{\pm 0.003}$
$\left[\mathbf{X}_{\mathcal{S}_M}\ \mathbf{X}_{\mathcal{S}_B}\right]$ MMM & k-gram	$0.468_{\pm 0.012}$	$0.860_{\pm 0.002}$
$\left[\mathbf{X}_{\mathcal{S}_M}\ \mathbf{X}_{\mathcal{S}_B}\ \mathbf{X}^{\mathbf{m}}_{\mathcal{G}_{\mathbf{r}}}\right]$ MMM & k-gram & graph	$0.550_{\pm 0.005}$	$0.914_{\pm 0.002}$

The results from Table 2.4 shows the classification performance with these features, which suggests minimal improvement employing longer sequence models. This may suggest that the bigram features can incorporate enough signal to capture spam activity in a multi-relational network. However, computing the Mixture of Markov Models does not impose much overhead when extracting bigram features, and can be done within the same process.

2.1.3.4 Demographic Information Performance

Many people use *Tagged* to find new relationships. We anticipate that in such environment users behave differently based on their demographics. To capture this point, we added three features ($\mathbf{X}_{\mathcal{D}}$) to our model: age, gender, and time since registration. Age and gender highly improved the classification results as they tend to be most discriminative of behavioral patterns. Another feature that we included in our model is the time past since registration. As mentioned earlier we labeled all the cancelled accounts for malicious activities as spammers. However, these users have different behavioral patterns, where spammers who mainly mass advertise, may use much newer accounts, in contrast to users who have been blocked due to misbehaviors, and have been active in the system much longer.

Table 2.5 shows the significant improvements of the results when including these features in different models. Figure 2.5 shows the *Precision-Recall* and *ROC* curves of the complete framework.

Table 2.5: Classification when including user demographics information.

Experiment	**AUPR**	**AUROC**
$\begin{bmatrix}\mathbf{X}_{\mathcal{D}} & \mathbf{X}_{\mathcal{S}_B}\end{bmatrix}$ Demo. & *k*-gram	$0.689_{\pm 0.006}$	$0.935_{\pm 0.001}$
$\begin{bmatrix}\mathbf{X}_{\mathcal{D}} & \mathbf{X}^{\mathbf{m}}_{\mathcal{G}_\mathbf{r}}\end{bmatrix}$ Demo. & graph	$0.701_{\pm 0.002}$	$0.950_{\pm 0.001}$
$\begin{bmatrix}\mathbf{X}_{\mathcal{D}} & \mathbf{X}_{\mathcal{S}_B} & \mathbf{X}^{\mathbf{m}}_{\mathcal{G}_\mathbf{r}}\end{bmatrix}$ Demo. & *k*-gram & graph	$0.778_{\pm 0.001}$	$0.963_{\pm 0.001}$
$\begin{bmatrix}\mathbf{X}_{\mathcal{D}} & \mathbf{X}_{\mathcal{S}_M} & \mathbf{X}_{\mathcal{S}_B} & \mathbf{X}^{\mathbf{m}}_{\mathcal{G}_\mathbf{r}}\end{bmatrix}$ Demo. & MMM & *k*-gram & graph	$0.779_{\pm 0.002}$	$0.963_{\pm 0.001}$

2.1.3.5 Collective Classification with Reports

The reporting system can have useful information to detect spammers. We studied the effectiveness of our proposed collective model (in Figure 2.4) to extract useful signals from this relation. We first designed a baseline model shown in Figure 2.6a to only use the reports to detect spammers. This model gives similar results to assigning total count of the reports for each user as their score of being a spammer. We then designed the model shown in Figure 2.6b to use the reports and prior credibility of the reporting user to detect spammers. This model gives similar results to assigning total *weighted* count of the reports for each user as their score of being a spammer, where reports are weighted by the credibility of the reporting users.

$$\begin{aligned} \textsc{Reported}(v_1, v_2) \rightarrow \; & \textsc{Spammer}(v_2) \\ & \neg\textsc{Spammer}(v) \end{aligned}$$

(a) HL-MRFs model that only uses the reports to detect spammers. This model would give similar results to assigning total count of the reports for each user as their score of being a spammer.

$$\begin{aligned} \textsc{Credible}(v_1) \wedge \textsc{Reported}(v_1, v_2) \rightarrow \; & \textsc{Spammer}(v_2) \\ \textsc{Prior-Credible}(v) \rightarrow \; & \textsc{Credible}(v) \\ \neg\textsc{Prior-Credible}(v) \rightarrow & \neg\textsc{Credible}(v) \\ & \neg\textsc{Spammer}(v) \end{aligned}$$

(b) HL-MRFs model that uses the reports and prior credibility of the reporting user to detect spammers. This model would give similar results to assigning total *weighted* counts of the reports for each user as their score of being a spammer.

Figure 2.6: Simple HL-MRFs models to compare with the collective model shown in Figure 2.4.

To perform the experiments we have only used $\mathcal{G}_{report}$, which is a sparse graph. Our collective model is aimed to propagate information between the reported users' likelihood of being spammer, through the credibility of the reporting users. In order for information to propagate in the model, each reporting user should at least have reported two other users. Hence, we removed the vertices with out-degree less than two. We then performed 10-fold cross validation to compare the three models and study the effectiveness of the collective model. We used the ratio of the correctly reported spammers from the training data as a simple prior on credibility for each user. Potentially more effective priors could incorporate the the count and the frequency of the reports as well.

Table 2.6 shows the results from our three experiments. Using the collective model significantly increases the performance of the reports in detecting spammers. These predictions can be added to the overall classification framework as a feature. However, since the report graph was sparse relative to the other relation graphs in our dataset, many users could not be classified with this model. Hence, we did not include these predictions as a feature in our framework. This model can be deployed independently to improve the signal from the reports.

Table 2.6: Classification with collective HL-MRFs model.

Experiment	**AUPR**	**AUROC**
Reports (Figure 2.6a)	0.674$\pm$0.008	0.611$\pm$0.007
Reports & Credibility (Figure 2.6b)	0.869$\pm$0.006	0.862$\pm$0.004
Reports & Credibility & Collective Reasoning (Figure 2.4)	0.884$\pm$0.005	0.873$\pm$0.004

2.1.4 Discussion and Conclusion

We have studied the characteristics of time-stamped multi-relational social networks that can be leveraged to detect spammers. We showed that by considering action or relation types and incorporating graph-based features from different relations, one can improve the spammer classification performance. We then showed two sequence mining techniques and their effectiveness to model sequences extracted from time-stamped multi-relational network for spam detection. We also proposed a collective model to refine and improve the signals from the abuse report graph.

Depending on the precision of the results from the model, the security system could either automatically flag a user as spammer and deactivate the account or block its activities in the system, or ask for more verification. Our experimental results show that our model can detect over 65% of the manually detected spammers with higher than 85% precision. These sophisticated spammers had passed the already deployed security measures and performed some activity in the network. Inspecting some of the false positives with the highest spammer probability, we found unlabeled and abandoned spammer accounts, which suggests the real precision of the proposed framework might actually be higher than reported. These results can significantly reduce the manual overhead of the administrative security team. Furthermore, our results show that the precision at 80% recall, is above 50%, suggesting this portion of users can be asked for additional verification (e.g., CAPTCHA) without affecting many legitimate users.

This model can be deployed as an iterative batch module to complement real-time filters. Except for some parameters such as the user *credibility prior* for the report system that should be set and adjusted globally for the user, the model has to be re-trained from a fresh sample of the network to adjust to the adversarial changes in patterns. Computing the parameters of the models on a relatively low-powered single machine for our experiments suggests that the framework could be run on very short intervals depending on the training size and computational power. The features can also be computed in parallel. Using *Graphlab Create*^{TM}, computing the features is highly efficient. To provide an example, for a graph with 5.6 million vertices and 350 million edges computing PageRank on our experiment machine took approximately 6.25 minutes, triangle counting 17.98 minutes, k-core 14.3 minutes, and graph coloring 143 minutes.

To optimize the model for production, it is possible to perform feature selection and reduce the necessary features. Feature selection [43] may also improve the performance of the model. Our method for collectively refining the signals from the report graph can be used independently or as a feature in the framework. Improved precision of the predictions via reports enables the system to take actions with more confidence, and reduces the manual overhead.

Our method should be retrained with every new sample. An online learning method that can incorporate the changes in the dynamic network can effectively improve the usability of our framework. Another approach that could improve the prediction results significantly could be incorporating this framework with content-based models. Furthermore, spam accounts often do not act independently and are part of spam campaigns. Their targets may often not be at random as well. They may use a white list of legitimate users to target. Our initial observations show that spammers make relations with legitimate users disproportionally to the overall population ratios. A multi-relational model that can classify spammer accounts based on their target accounts, and identify campaigns based on their relational information could potentially improve the results.

Chapter 3: Predictions on Links

Another important task in multi-relational networks is predicting information about links. The prediction could be about the existence or absence of a link or relation in the network. For example “will two users become connected in the future?”, “will a user click on a link?”, “will a researcher cite the work of another researcher?”, or “will a drug interact with a particular target? or another drug?”. These tasks that can be designed as a binary prediction about the existence of a link are often referred to as *link prediction* [44].

Another set of predictive tasks on links is predicting a value (e.g., strength) for the links. A common example is recommender systems, where the question is, “what is the rating a user would give to an item such as a movie or a product?”. In this case, we mostly care about the strength or weight of the link or relation. This task is sometimes called *link regression* [45].

Like any predictive task, predictions on links can be evaluated via different metrics. The plan for using these predictions determines their evaluation methods. Many of the predictions in multi-relational networks will be used as a basis for recommendation for further actions. For example, recommending a product to a user, or showing a list of related links, pages, articles, and other users to a user to connect to, or prioritizing which potential interactions in a biological network look more promising for further experiments are all predictions that can be used for further actions. Therefore, link prediction methods are often evaluated with a ranking measure like precision, recall, and the ROC curve [46].

Many common data mining tasks can be mapped to a link prediction on a bi-typed network structure. The following sections describes this common structure and its applications.

3.1 General Augmented Bipartite Structure

A common multi-relational network structure that can model many applications is an augmented bipartite structure also called bi-typed networks [47]. In these structures (shown in Figure 3.1a), we have two types of nodes and links of interest that we want to make prediction about between these different types of nodes. Examples of such structure include networks of drugs and targets and interactions between them, networks of users and items and their interactions, networks of images and keywords and the links between them, and networks of researchers and topics and the authorship relation between them.

Different predictive models have been proposed for the basic form of this structure, from matrix completions [48] to counting paths, generative probabilistic graphical models [49], and neural networks. However, it is highly common that additional information is available in the network that can help with the performance of the model. The common approach to model the augmented bipartite network is to customize one of the general approaches, not specifically designed for multi-relational networks, based on the type of information available [50].

Figure 3.1 shows the different types of information that can augment this bipartite structure. Some cases are the following:

1. Additional non-relational features for each node (Figure 3.1b). Examples are demographic features of users and content information for items in recommender systems.

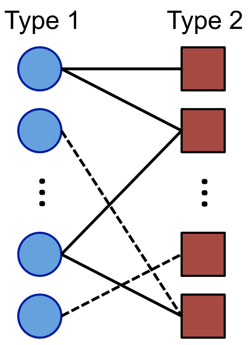
2. Additional relational features (links) for each node (Figure 3.1c). Examples include similarities between two drugs based on their chemical structure, and similarities between two targets based on their sequences in a drug-target interaction network.

3. Additional relational features (links) for each node with external nodes (Figure 3.1d). Examples are author-institute and paper-conference relationships for author-paper citation network.

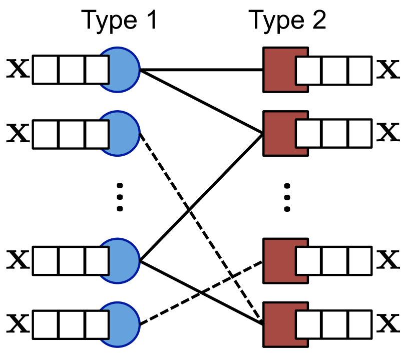
4. Additional non-relational features for each link (Figure 3.1e). An example of this is context features such as time in recommender systems.

5. Additional relational features for each link (Figure 3.1f). An example of this is adding the download relationship between authors and papers to the citation network.

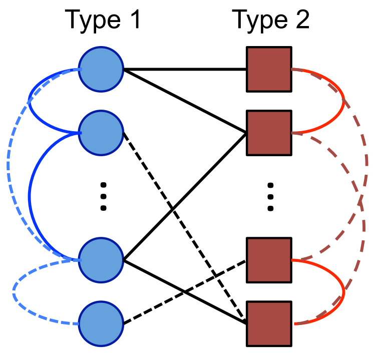
These structures often can be converted to other forms to prepare them to best fit the desired model. Specifically, most of these structures can be converted



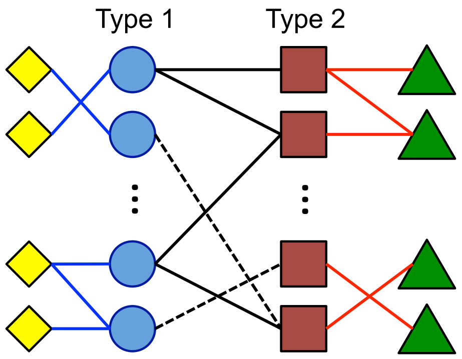
(a) Simple bipartite graph with two types of nodes.



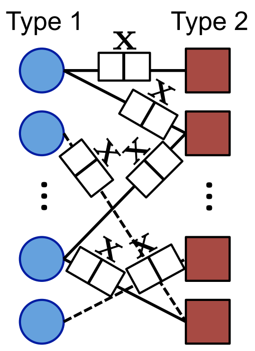
(b) Bipartite graph with additional features for each node.



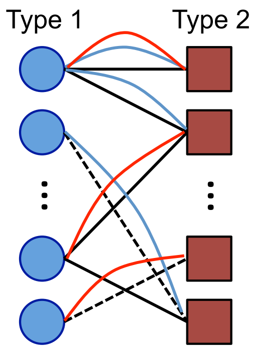
(c) Bipartite graph with augmented relational features (links) for each node.



(d) Bipartite graph with augmented relational features (links) with external nodes for each node.



(e) Bipartite graph with augmented features for each link.



(f) Bipartite graph with augmented relational features (links) for each link.

Figure 3.1: Augmented bipartite multi-relational network structures.

to the form shown in Figure 3.1c. Figure 3.2 shows an example of such case where 3.1d can be converted to 3.1c by computing Jaccard or Cosine similarity, and 3.1b to 3.1c by computing the distances (or similarities) in feature spaces.

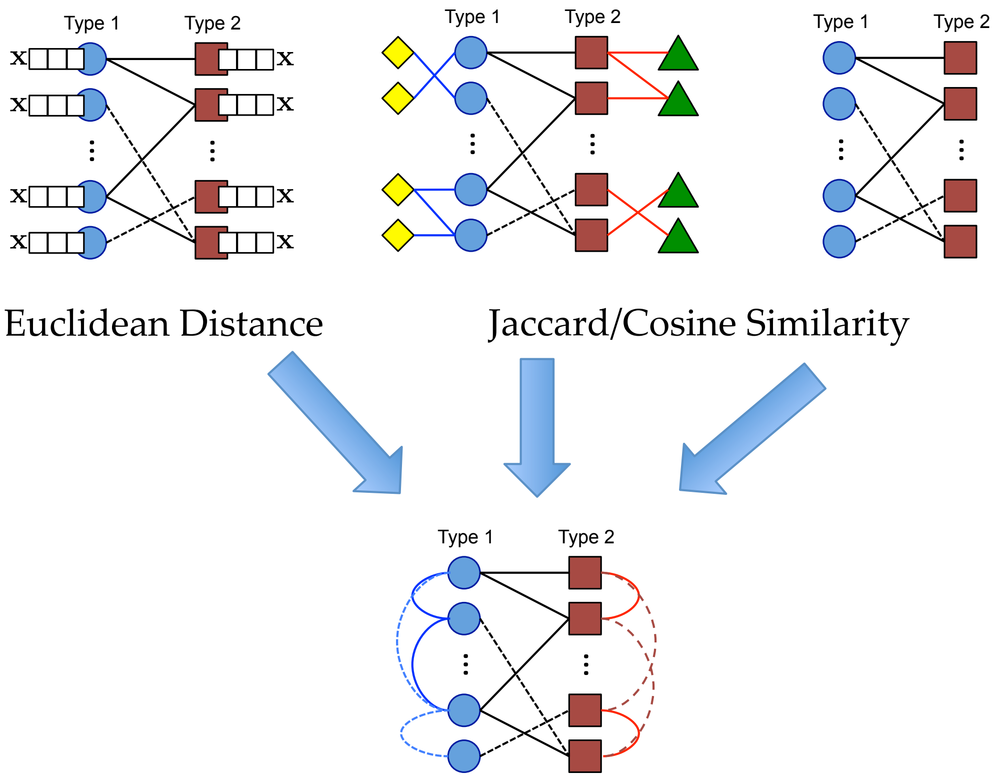


Figure 3.2: Converting different augmented bipartite multi-relational network structures to a single form.

We propose a general method to make predictions on the links for the augmented form of the bi-types structure, using a special form of probabilistic graphical model. We have used this method to achieve state-of-the-art performance for link prediction in drug-target interaction networks [51] and recommender systems [52]. In this chapter, we discuss the application of our approach in drug-target interactions networks and its extension to recommender systems. The following section describes our general solutions for link inference, and its application in different domains.

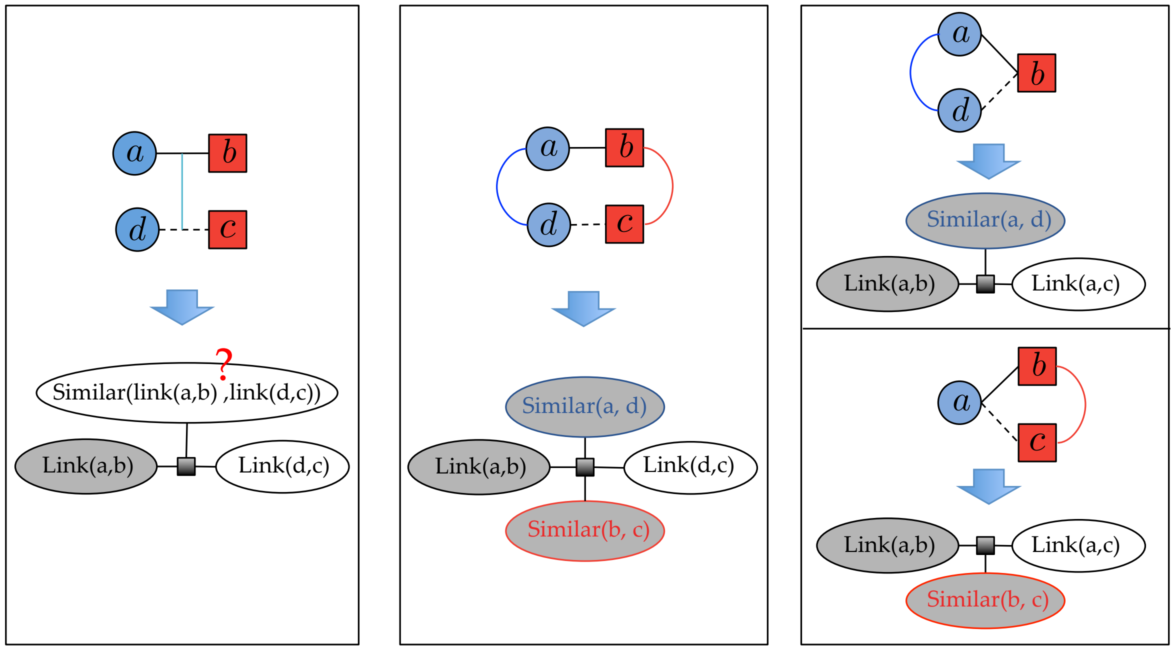
3.2 Proposed General Link Inference Model

As we have empirically shown in Chapter 2, both modeling the multi-relational nature of the network and collective predictions improve the predictive performance. In our method for predictions on links, *link inference* for short, the aim is to leverage both of these findings. In addition, we want to be able to incorporate any prior information we would have about the network and links into our models. To capture all these, we propose a similarity-based link inference method, that we implement via a special probabilistic graphical model described in Appendix 6. The proposed methods can be used as a template for link inference in several domains with different kinds of information. Our template model is based on two main components; *similarity-based reasoning* and *priors*. The following sections describe each of the components in more details.

3.2.1 Similarity-based Reasoning

Our method is based on the assumptions that *similar links have similar values*. One challenge is to define the similarity between links, which is not directly observed in the data. We consider the similarity between two links to be the similarity between the two end nodes of the links. In the common structure shown in Figure 3.1c, the similarity between the two end nodes of the two links are captured in the network.

Figure 3.3 shows the model to consider the similarity of two links, and the way to model the similarity of their end nodes. As shown in the corresponding factor graphs for each of the scenarios, we do not observe the link similarity in Figure 3.3a, but we can observe the similarity of the nodes used in Figure 3.3b. A special case of our method is when we only consider the similarity of the nodes on one side. This situation resembles the common triad structure used in graph mining tasks to capture concepts such as *homophily*. Figure 3.3c shows the corresponding factor graph of the triad structure.



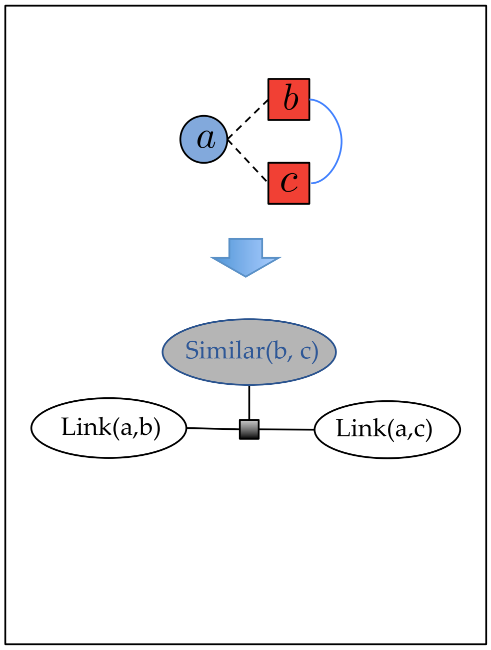
(a) Directly modeling the similarity between links.

(b) Modeling similarity of the links as similarity of nodes.

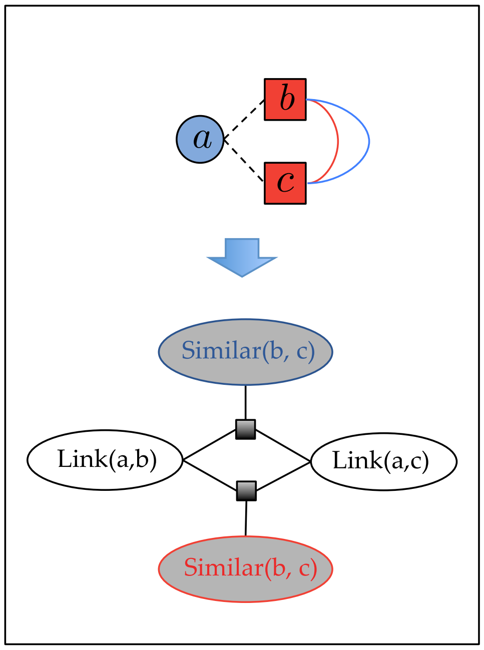
(c) Special case of traid structure.

Figure 3.3: Template model for multi-relational similarity-based link inference, where similar links should have similar values.

Our similarity-based model supports the two main characteristics of real-world networks; multi-relational nature and collective predictions. Multi-relational nature is modeled via the possibility of using multiple similarities in our factor model shown in Figure 3.4b, and the the collective predictions are supported via connecting two instances with unknown labels in our factor model shown in Figure 3.4a.



(a) Performing collective predictions or joint inference via connecting the instances with unknown labels (i.e., unobserved variables) in the factor graph.



(b) Supporting multiple similarities between nodes in the factor graph via including an observed variables for each type of similarity.

Figure 3.4: Leveraging multi-relational nature of the network and collective predictions in our template model.

3.2.2 Priors

Often additional prior information about the links are available that we want to include in our model. We identified three types of prior information that are commonly available in different networks:

Sparsity: Often we know that network is sparse and we want to incorporate that information in the model, and enforce such prior on the model. The strength of such prior could be set manually or learned from training data.

Distribution mean: In *link regression* we may know the mean value corresponding to the group of links of interest. For example, in recommender systems, we may want to incorporate the average rating that a user gives to items or the average rating an item receives from all users in our model. We refer to such constraints the *distribution mean* priors.

Predictions from other models: We can also incorporate predictions from other methods in our model as priors and use our model as an ensemble approach. Other methods such as factorization-based approaches may not be able to incorporate the multi-relational nature for the graph or perform collective predictions, but they may provide signals based on other aspects to improve the performance.

3.2.3 Implementation via HL-MRFs

The proposed template link inference model can be implemented via different probabilistic graphical model frameworks. Due to computational efficiency and modeling convenience of using *Probabilistic Soft Logic (PSL)* we chose *Hinge-loss Markov Random Fields (HL-MRFs)* to implement our model. The details of HL-MRFs are provided in Appendix 6. This section includes details to implement our template model with HL-MRFs, and how our modeling requirements is satisfied by it.

Similarity-based reasoning: To implement a similarity-based link inference model presented in Figure 3.3b we use the following logical representation with PSL:

$$\lambda : \textsc{Link}(A, B) \wedge \textsc{Similar}(B, C) \wedge \textsc{Similar}(A, D) \rightarrow \textsc{Link}(D, C) \tag{3.1}$$

And to implement the special case of the similarity-based link inference template presented in Figure 3.3c we use the following logical representation with PSL:

$$\lambda : \textsc{Link}(A, B) \wedge \textsc{Similar}(B, C) \rightarrow \textsc{Link}(A, C) \tag{3.2}$$

As described in Appendix 6, to infer the values corresponding to the MAP state in HL-MRFs, the following loss is minimized over all random variables.

$$\ell = \big[\max\{0, val(r_{\text{body}}) - val(r_{\text{head}})\}\big]^p$$

where a rule is in the form of $r_{\text{body}} \longrightarrow r_{\text{head}}$ and $p \in \{1, 2\}$ defines the loss to be linear or quadratic.

For example, rule 3.2 translates to a potential function in our factor graph that results in minimizing the following loss when $(\textsc{Link}(a, b) + \textsc{Similar}(b, c) > 1)$:

$$\ell = \left[\max\{0, \textsc{Link}(a,b) - (1 - \textsc{Similar}(b,c)) - \textsc{Link}(a,c)\}\right]^p$$

where a,b,c are instances that can instantiate the rule.

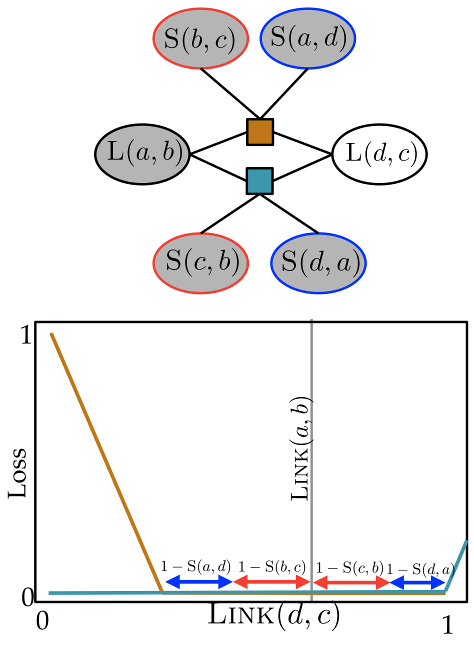
Figure 3.5b shows the schematic loss for this rule in linear form where $\textsc{Link}(a,b)$ and $\textsc{Similar}(b,c)$ are observed and $\textsc{Link}(a,b)$ is not observed. Intuitively, the plateau region corresponding to $1-\textsc{Similar}(b,c)$ is the uncertainty on the closeness of the values of two links. For example, in the extreme case where $\textsc{Similar}(b,c) = 1$ any divergence between the two values of the links results in a penalty. The loss corresponding to the general form of the rule 3.1 is shown in Figure 3.5a. Similar intuitive arguments holds for the general form as well.

Priors: We use two types of PSL templates to model the three set of priors mentioned earlier; *negative priors* to enforce sparsity and *double-sided prior* to model distribution mean and predictions from other models.

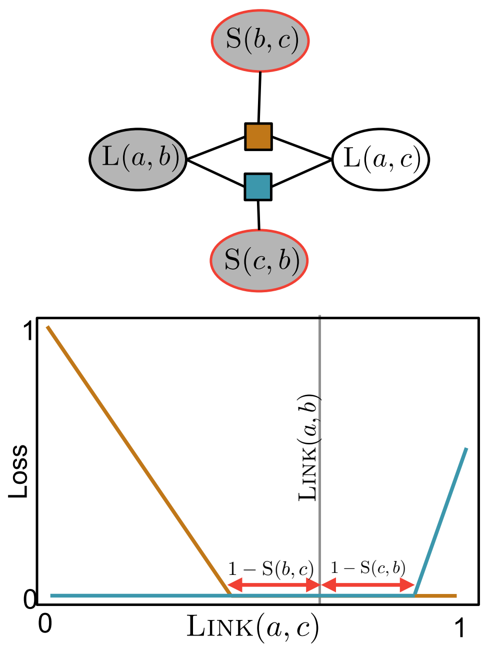
Negative priors implemented via the following rule simply state that the link should not exist. The corresponding loss to the negative prior is shown in Figure 3.6a.

$$\lambda : \neg\textsc{Link}(A,B) \tag{3.3}$$

The double-sided priors implemented via the following rule constrain the value of the link to be close to the value of the prior. The corresponding loss to the double-sided prior is shown in Figure 3.6b.



(a) Tetrad-based structure where we have similarities between nodes on both ends of the links.



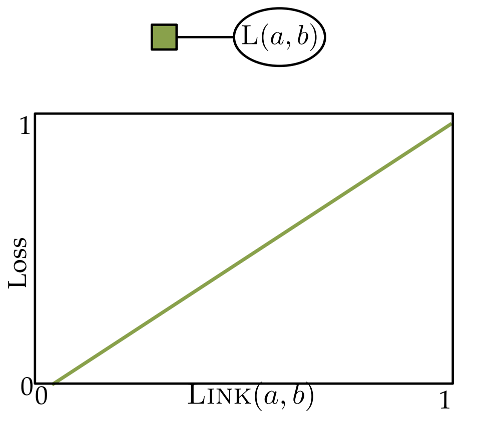
(b) Triad-based structure where we have similarities between nodes on one end of the link.

Figure 3.5: Corresponding losses to each template. $\mathrm{S}(a,d)$ and $\mathrm{L}(a,b)$ indicate $\textsc{Similar}(a,d)$ and $\textsc{Link}(a,b)$.

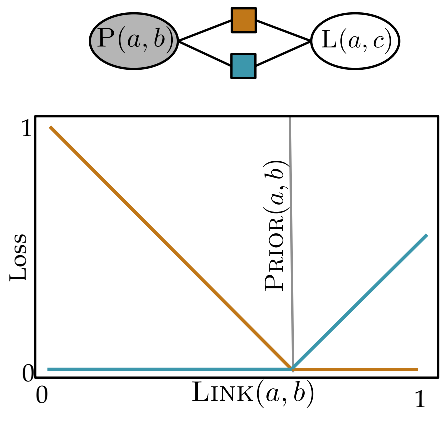
$$\lambda : \textsc{Prior}(A,B) \rightarrow \ \textsc{Link}(A,B) \tag{3.4}$$

$$\lambda : \neg\textsc{Prior}(A,B) \rightarrow \ \neg\textsc{Link}(A,B) \tag{3.5}$$

As shown in Figure 3.6, the negative prior enforces the values of the links to be close to zero unless there is evidence to push the value higher and justify the penalty of the negative prior. Similarly in the double-sided prior unless there is evidence to push the value of the link to diverge from the value of the prior, the penalty will force the two values to be similar.



(a) The negative prior on a link results in a penalty if the value of the link increases from zero.



(b) The double-sided prior results in a penalty if the value of the link diverges from the values of the prior.

Figure 3.6: Corresponding losses to each prior template.

We have proposed solutions based on our link inference framework for drug-target interaction prediction and hybrid recommender systems. The following sections describe the details of the solutions.

3.3 Drug-Target Interaction Prediction

The cost of successful novel chemistry-based drug development often reaches billions of dollars, and the time to introduce the drug to market often comes close to a decade. Most new compounds fail during clinical trials or show adverse side effects. Because of the high failure rate of drugs during this process, the trial phase is often referred to as the "valley of death" [53].

Most drugs[1] affect multiple targets, and *Polypharmacology*, the study of such interactions, is an area of growing interest [54]. These multi-target interactions potentially result in both unintentional therapeutic and adverse side effects. Predicting side effects during the drug developmental phase can reduce the high cost of clinical trials and is crucial for the commercial success of new drugs. Moreover, due to the high cost and low success rate of novel drug development, pharmaceutical companies are particularly interested in *drug repositioning* or *repurposing*, which involves finding new therapeutic effects of pre-approved drugs.

Sildenafil—originally developed for pulmonary arterial hypertension treatment—is a famous drug repurposing example. In clinical trials, it was discovered by chance to have a side effect of treating erectile dysfunction in men, and it was eventually re-branded as *Viagra* [55].

Drug-target interaction identification is an essential step of drug repurposing and drug adverse effect prediction. *In vitro* identification of drug-target associations

[1]Organic molecules that bind to bio-molecular targets and inhibit or activate their functions.

is a labor-intensive and costly procedure. Hence, *in silico* prediction methods are promising approaches for focusing *in vitro* investigations [56].

There are several methods to model the drug-target interaction prediction task [57], many of which use a network representation [58]. We can construct a bipartite interaction network where nodes represent drugs and targets, and edges denote interactions. Drug-drug and target-target similarities can augment this network on each side. Data from multiple publicly accessible datasets can be integrated toward building these networks [59]. The similarities between drugs and between targets have different semantics. For example, targets can have similarity measures based on their sequences and their ontology annotations [60]. Figure 3.7 shows a schematic overview of a drug-target interaction network.

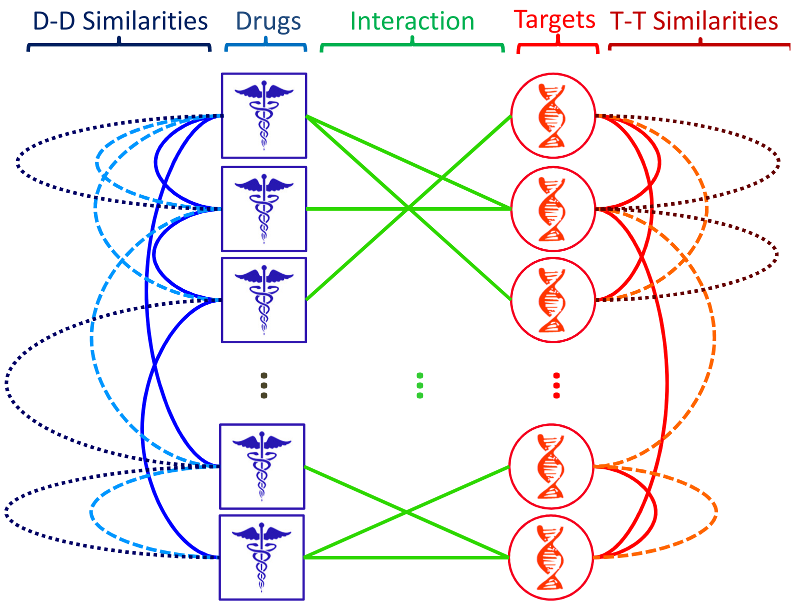


Figure 3.7: A schematic overview of a drug-target interaction network. Edges between drugs and between targets represent different similarities.

A link prediction method can predict new potential drug-target interactions in this setting [1, 61]. However, traditional link prediction methods often ignore the multi-relational characteristics of this drug-target interaction network (i.e., nodes and edges with different semantics) or make oversimplifying assumptions that neglect key, interdependent phenomena during prediction.

The structure of the network and the multi-relational aspects make it challenging to convert such knowledge into the (flat) data formats that are typically used with standard prediction algorithms. Attempts to make such conversions often rely on potentially ad-hoc feature engineering approaches [60, 62]. Such methods may sometimes yield good prediction performance, but they suffer from low interpretability and loss of information. Our approach is based on the premise that links depend on the similarities between their endpoints and on other interactions. Hence, a collective approach is more appropriate than standard machine learning models that make simplifying independence assumptions. As described in Section 3.3.4, in a collective setting, the presence or absence of interactions are studied interdependently.

In this section we present a drug target prediction framework based on *probabilistic soft logic* (PSL) [63]. We reason collectively over the unknown interactions using a structured representation that captures the multi-relational nature of the network. We design a PSL model for drug-target interaction prediction that reasons over structured rules. We consider two types of structured rules, triad rules and tetrad rules, and consider a variety of similarity metrics. We propose a similarity selection method to manage the large computational cost of inference in this task and perform experimental studies on different aspects of link prediction in the drug-target interaction domain with this model. We compare the relative improvement provided by each type of structural rule and find that triad-based rules enable more accurate predictions. We also experiment with the effect of using different similarity metrics and show that combining all similarity metrics in a single probabilistic model produces the most effective model. We additionally test the importance of collective inference in such models by comparing against an analogous model that makes independent predictions. Our PSL based solution outperforms the state-of-the-art drug-target interaction prediction method proposed by Perlman et al. [60]. We further validate that our PSL models can outperform Perlman et al. [60] on a set of new interactions that were not considered in the original evaluation of Perlman et al. [60].

3.3.1 Related Work in Drug-Target Interaction Prediction

In the *similarity ensemble approach* (SEA), Keiser et al. [56] use ligands to predict interaction. They use ligands for target representation and chemical similarities between drugs and ligand sets as potential interaction indicators. In CMap, by Lamb et al. [64, 65], mRNA expressions are used to represent diseases, genes, and drugs. They compare up- and down-regulations of the gene-expression profiles from cultured human cells treated with bioactive molecules and provide cross-platform comparisons. They predict new potential interactions based on opposite-expression profiles of drugs and diseases. Chang et al. [66, 67] proposed a method for predicting drug targets for a given phenotype predicting phenotypes given specific genetic perturbations.

A number of methods reason about network structures to predict interactions. Cockell et al. [55] describe how to integrate drugs, targets, genes, proteins, and pathways into a network for different tasks. They present a hypothesis that similar targets interact with the same drugs, and similar drugs tend to interact with the same targets. Lee et al. [59] describe drug repurposing, multi-agent drug development, and estimation of drug effects on target perturbations via network-based solutions.

Yildirim et al. [58] explain trends in the drug-discovery industry over time using a network-based analysis and show the effect of sequencing the genome on drug development. They also discuss different structural aspects of this network including preferential attachment and cluster formation.

Network-based approaches integrate drug-drug and target-target similarities extracted via different methods (e.g. SEA and CMap) with the drug-target interactions network. The following methods use a single similarity measure for drugs and targets to predict interactions: Cheng et al. [68] predict potential interactions using drug-drug and target-target similarities and a bipartite interaction graph. Using SIMCOMP [69], they compute the 2D chemical drug similarities and sequence similarities for targets via the Smith-Waterman score. They use the following three link-prediction methods: drug-based similarity inference (DBSI)—only considering similarities between drugs; target-based similarity inference (TBSI)—only considering target similarities; network-based inference (NBI)—combining both similarities.

Alaimo et al. [70] extend this approach by proposing a DT-hybrid method that integrates prior domain-dependent knowledge.

Yamanishi et al. [62] propose the following three methods for interaction prediction: a nearest neighbor approach; weighted k-nearest neighbors; and space integration. For the space integration method, they describe a genomic space, using the Smith-Waterman score for targets, and the SIMCOMP score for drugs. They propose a method to integrate drugs and targets in a unified latent *pharmacological space*, and they predict interactions in that space based on the proximity of drugs and targets. They separate out four categories of targets, namely enzymes, ion channels, GPCR, and nuclear receptors for their experiments. They also report that similar drugs tend to interact with similar targets and vise versa.

Bleakley and Yamanishi [71] extend this method and construct local models for graph inference. They classify each interaction twice and combine the results to provide a prediction. First, they build a classifier based on drugs and then based on targets. They use the similarities as the *support vector machine* (SVM) kernels. Extending this method, Mei et al. [72] propose to infer training data from neighbors' interaction profiles to make predictions for new drug or target candidate that do not have any interactions in the network. Wang and Zeng [73] propose a method based on restricted Boltzmann machines for drug-target interaction prediction.

More advanced methods predict interactions based on multiple similarities. Chen et al. [74] reason about the possibility of a drug-target interaction in relation with other linked objects. They use distance, shortest paths, and other topological properties in the network to assess the strength of a relation. They assign scores to paths between drugs and targets and combine path scores for each drug-target pair.

Perlman et al. [60] propose a feature-engineering method based on combinations of drug-drug and target-target similarities and use classification to predict interactions. They build their method based on five drug-drug and three target-target similarities. They evaluate their model using cross validation over three online datasets and validate their predication on a fourth dataset. They show significant performance improvement over Bleakley and Yamanishi [71] and Yamanishi et al. [62] that only use one type of drug-drug and target-target similarities for prediction. To the best of our knowledge Perlman et al. [60] method is the state-of-the-art mutli-similarity based approach for drug-target interaction prediction.

Gottlieb et al. [75] extend their method in [60] to drug-disease domain and propose a personalized medicine approach, representing diseases via their genetic signatures. This method can predict the most effective compound for a genetic signature of an unknown disease.

3.3.2 Our Model

Our proposed drug-target prediction framework uses *probabilistic soft logic* (PSL) [63]. We provided more details about hinge-loss Markov random fields (HL-MRF) and probabilistic soft logic (PSL) in appendix 6. Here we present the models that we use for drug-target interaction prediction based on our general link inference template with HL-MRF.

We design a PSL program using rules that capture domain knowledge about the drug-target interaction problem. Our rules model the idea that similarity among drugs may imply similar interactions with targets, and similarity among targets may imply similar interactions with drugs. We incorporate many types of similarities into a single joint probabilistic model, simultaneously reasoning about the various possible interactions.

Triad-based rules For drug-target interaction prediction, many established methods are based on triangles or triads between drugs and targets. These triads occur between two similar targets and a drug that interacts with both of them, or two similar drugs and a target that both drugs interacts with. The hypothesis is that similar targets tend to interact with the same drug and that similar drugs tend to interact with the same target [55, 62, 68]. Figure 3.8 depicts the triad-based prediction of interactions for drugs and targets.

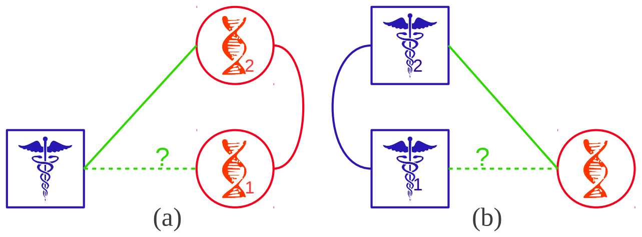


Figure 3.8: Similar targets tend to interact with the same drug (a), and similar drugs tend to interact with the same target (b).

The following rule captures the triad shown in Figure 3.8(a) for targets:

$$\textsc{SimilarTarget}_{\beta}(T_1, T_2) \wedge \textsc{Interacts}(D, T_2) \rightarrow \textsc{Interacts}(D, T_1) \tag{3.6}$$

And the following rule captures the triad in Figure 3.8(b) for drugs:

$$\textsc{SimilarDrug}_{\alpha}(D_1, D_2) \wedge \textsc{Interacts}(D_2, T) \rightarrow \textsc{Interacts}(D_1, T) \tag{3.7}$$

where T denotes a target, D indicates a drug, predicate $\textsc{SimilarTarget}_{\beta}$ represents a specific target-target similarity metric.

For each similarity metric, we add an instance of rule (3.6) to the PSL model. Our model is capable of integrating any set of similarities with these rules. As described in Section 3.3.5, we include three instances of this rule, where β is *sequence-based*, *PPI-network-based*, or *gene ontology-based*. Predicate $\textsc{SimilarDrug}_{\alpha}$ represents a specific drug-drug similarity measure. We consider five instances of rule (3.7), where α is *chemical-based*, *ligand-based*, *expression-based*, *side-effect-based*, or *annotation-based*.

Tetrad-based rules In addition to triads, we also consider more general templates for reasoning about both drug and target similarities to predict interactions. Specifically, when a drug interacts with a target, we may expect another similar drug to interact with another similar target. Figure 3.9 illustrates this hypothesis. We encode this hypothesis using the following *tetrad rules*:

$$\begin{split}\textsc{SimilarDrug}_{\alpha}(D_1, D_2) \wedge \textsc{SimilarTarget}_{\beta}(T_1, T_2) \wedge \textsc{Interacts}(D_2, T_2) \\ \rightarrow \textsc{Interacts}(D_1, T_1)\end{split} \tag{3.8}$$

where α and β are drug-drug or target-target similarity measures as discussed earlier. We include multiple instances of triad and tetrad rules corresponding to the three drug-drug and five target-target similarity measures.

To further enhance our model, we also experiment with an extension of the tetrad-based rules which we call *exclusive tetrad* rules. The idea behind this extension is to exclusively ground rules for the tetrad structures that do not include any triads inside them:

$$\begin{aligned}\neg\textsc{Interacts}(D_1, T_2) \wedge \neg\textsc{Interacts}(D_2, T_1) \wedge \textsc{SimilarDrug}_\alpha(D_1, D_2)\\ \wedge\textsc{SimilarTarget}_\beta(T_1, T_2) \wedge \textsc{Interacts}(D_2, T_2)\\ \rightarrow \textsc{Interacts}(D_1, T_1)\end{aligned} \tag{3.9}$$

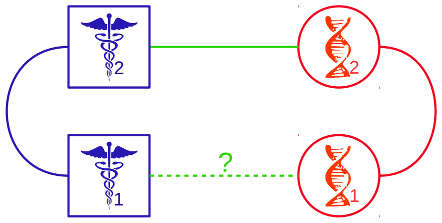


Figure 3.9: If one of the drugs interacts with one of the targets, the other drug may interact with the other target as well.

Negative prior We also include a negative prior indicating that the Interacts predicate is most likely false, accounting for the natural sparsity in the drug-interaction network. The negative prior rule is as follows:

$$\neg \textsc{Interacts}(D, T). \tag{3.10}$$

The similarity predicates $\textsc{SimilarDrug}_\alpha$ and $\textsc{SimilarTarget}_\beta$ represent observed values, and the interaction predicate $\textsc{Interacts}$ represents values that are partially observed. These rules all combine to form a complex, structured model that captures a large number of dependencies between unknown $\textsc{Interacts}$ values that we aim to predict. In the next section, we discuss techniques to manage the high complexity of this model.

3.3.3 Similarity Selection

Because PSL inference considers all possible substitutions for the rules, the number of ground rules can be extremely large. Let $|D|$ denote the number of drugs, $|\alpha|$ the number of different similarities between them, $|T|$ the number of targets, and $|\beta|$ the number of different similarities between targets. Then each potential link can be involved in $O(|D| \times |\alpha|)$ instances of rule (3.7), $O(|T| \times |\beta|)$ instances of rule (3.6). For tetrad-based rules, the situation is even worse because the number of possible substitutions is even greater. In addition, since there are $O(|D| \times |T|)$ potential interactions, the total number of ground rules is $O(|D||T|(|D||\alpha|+|T||\beta|))$. Running inference on such a massive number of ground rules is too computationally expensive for many practical settings.

To limit the number of ground rules, we prevent some of the rules from being grounded by reducing the number of triads and tetrads that are considered for each potential link. To reduce this number, we essentially ignore some of the less similar drugs and targets pairs. This strategy is reminiscent of *blocking* [76, 77, 78], which is a term that refers to the process of limiting the number of links considered. Typically, blocking decisions are done to avoid the quadratic costs inherent in link prediction settings.

There are several ways to approach blocking in our problem; the most basic strategy simply uses a fixed threshold for all similarities and sets the values below that threshold to zero. However, although the similarities are normalized to $[0, 1]$, the distribution of the values tends to be highly varied such that a fixed-threshold approach can ignore most of the values in some similarity measures or include most of the values from another. Figure 3.10 plots the distribution of similarities in our dataset, illustrating the diversity in the shapes of distributions these similarity measures generate.

Another method is to choose a different threshold for each similarity measure. While potentially better than the previous approach, similar issues occur due to variability in individual target and drug similarity distributions. Similarities for each target or for each drug can have highly variable values, and choosing a fixed threshold will include too many similarities for some particular drugs or targets and very few for others. Figure 3.11 shows the annotation-based similarity for drugs and demonstrates an instance of this situation.

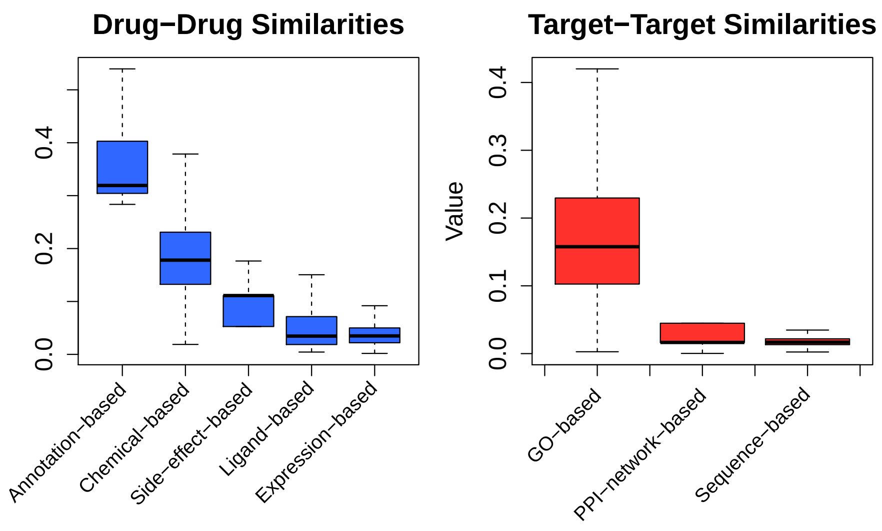


Figure 3.10: Distribution variation of different similarity values between drugs and between targets. Similarities with values of zero or one are omitted in this plot.

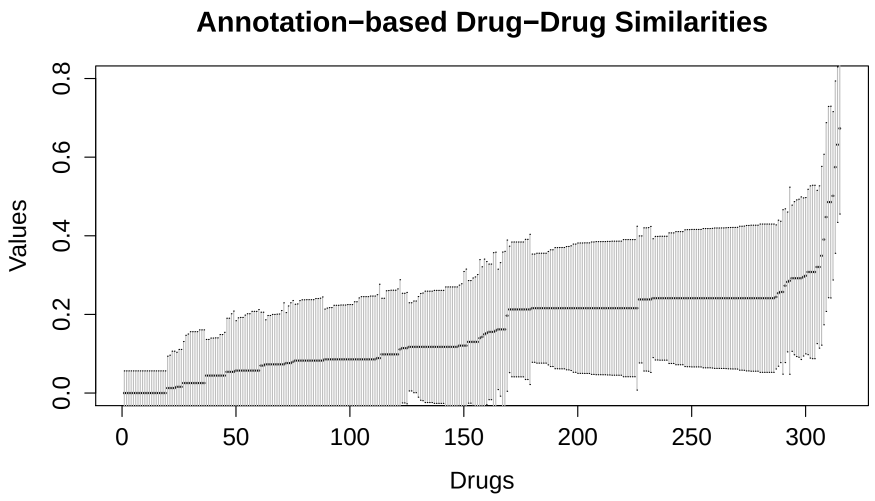


Figure 3.11: Distribution of annotation-based similarity values for 315 drugs, where dots indicate mean similarity value between each drug and all others, and lines demonstrate standard deviation of the values. Similarities with values of zero or one are omitted in this plot. e.g., the mean of all annotation-based drug similarities with drug #200 is about 0.2 with standard deviation of 0.15.

Instead, our approach uses k-nearest-neighbors to ensure that every drug and every target considers at least a few values from each similarity. In this approach, we preserve the k-highest values in each similarity for each drug and each target and set the others to zero. However, depending on the method used for calculating the similarities, there are many cases that similarity values between multiple drugs or targets are the same. Hence, the k-th nearest neighbor of a drug or target can have the same similarity as a large set of other drugs or targets. To address the possibility of ties, we consider the drugs or targets with similarities greater than the k-th nearest neighbor. In other words, we only include the similarities from k-1 drugs or targets. Formally, the *selected* set of similarity predicates are as follows:

$$\textsc{Similar}_\lambda^{\text{selected}} = \begin{cases} \textsc{Similar}_\lambda(x_i, x_j) & \text{if } \textsc{Similar}_\lambda(x_i, x_j) > \textsc{Similar}_\lambda(x_i, x_k); \\ 0 & \text{otherwise.} \end{cases} \tag{3.11}$$

where λ is any drug-drug or target-target similarity and x_k is the k-th nearest neighbor of x_i.

3.3.4 Insights into Collective Link Inference

Traditional machine learning approaches often predict outputs independently, separating examples into distinct, unrelated instances. For example, in the drug-target interaction prediction setting, the presence or absence of each interaction is determined based on the evidence and independent of the other interactions. However, actual interactions may be interdependent.

Classification problems that consider interdependencies are known as *collective classification* [79]. Algorithms that perform collective classification exploit global information propagation through networks defined over the data. Since PSL performs MPE inference on the interpretation I over the whole network, interaction predictions propagate and influence the prediction of other interactions. Thus, PSL models perform collective classification and can reason about new interactions using other predicated interactions.

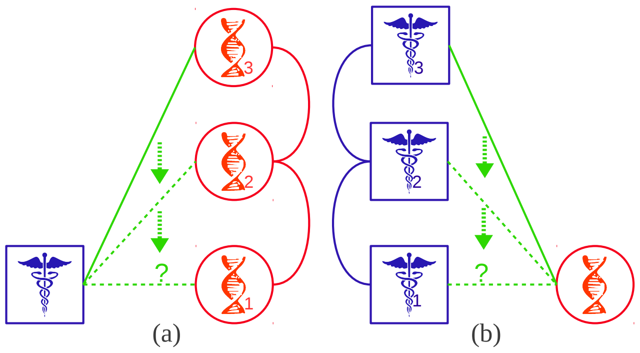


Figure 3.12: Predicted interactions can be used for other inferences using target (a) or drug similarities (b), or both.

Specifically, rules (3.6) and (3.7) adopt a *collective inference* approach, using inferred links to imply the existence of other links, that results in global information propagation through the network. Figure 3.12 shows a situation where a predicted interaction is used in predicting other interactions.

We designed a model to experiment with the potential detrimental effect of making independent predictions. We use rules analogous to (3.6) and (3.7) that do not allow collective inference:

$$\textsc{SimilarTarget}_{\beta}(T_1, T_2) \wedge \textsc{ObservedInteracts}(D, T_2) \rightarrow \textsc{Interacts}(D, T_1) \tag{3.12}$$

$$\textsc{SimilarDrug}_{\alpha}(D_1, D_2) \wedge \textsc{ObservedInteracts}(D_2, T) \rightarrow \textsc{Interacts}(D_1, T) \tag{3.13}$$

We ground ObservedInteracts with the observed interactions from the dataset and use predicate Interacts for predictions. In contrast to rules (3.6) and (3.7), here only observed interactions imply the presence of new interactions, whereas in our full joint model, inferred interactions can imply other interactions. Hence, predictions using these new rules are only made based on observed evidence. We report the comparative results of the collective and non-collective models in the experimental analysis section.

3.3.5 Experimental Validation

We perform an extensive evaluation of our PSL based method on a dataset that was obtained from Perlman et al. [60]. We first report on the behavior of the PSL based method[2] for different configurations as follows:

- **Rule structure:** We first compare the effectiveness of triad-based (3.6 & 3.7) and tetrad-based rules (3.8). This study serves as an example to test different domain assumptions for this task.

- **Similarity Selection:** We show the effectiveness of our proposed similarity selection strategy by showing the speedup and performance stability of our method.

- **Weight learning:** We measure the effect of weight learning on performance by comparing models with and without weight learning.

- **Collective inference:** We show the strength of models with collective inference, by comparing our collective model versus the non-collective version of our model.

- **Combining similarities:** Finally, we measure the effectiveness of combining information from different similarities. In this study, we compare models with all similarities against models using a single similarity.

[2]Our code and data we used for our experiments along with our implementation of the approach from Perlman et al. [60] can be obtained from: https://github.com/shobeir/fakhraei_tcbb2014

We then compare the (best) performance of our method against the method of Perlman et al. [60].

3.3.5.1 Dataset

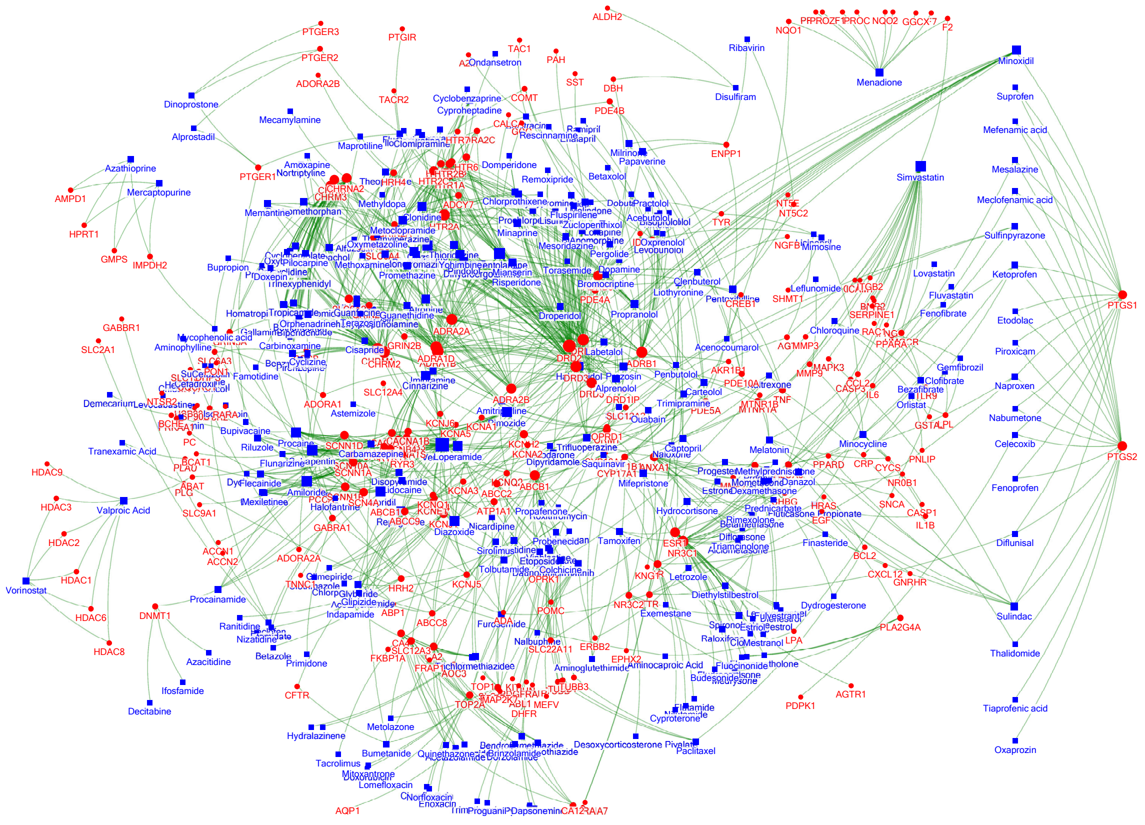


Figure 3.13: Network of drug-target interactions in the dataset. Drugs are shown with blue squares and targets with red circles, where size of the node represent their degree. Similarities are not shown to simplify the graph.

We obtained our dataset from Perlman et al. [60]. The interactions between drugs and targets are obtained from DrugBank [80], KEGG Drug [81], DCDB (Drug Combination database) [82], and Matador [83]. We also use the same five drug-drug and three target-target similarities used by Perlman et al. [60].

We filtered the dataset to remove the drugs and targets that do not have any computed similarities. The final dataset includes 315 drugs, 250 targets, and 1,306 interactions.

Using NodeXL [84], we calculated graph statistics and visualize the graph. The graph contains 16 connected components, and the largest component includes 518 vertices and 1280 edges. The average geodesic distance[3] in the graph is 5.31 with a maximum of 15. The vertices' degrees range from 1 to 37, with average of 4.6. Figure 3.13 shows an overall visualization of the drug-target interactions in the dataset, where drugs are drawn as blue squares and targets as red circles.

This section includes a brief description of the methods used for similarity calculation and how they were computed by Perlman et al. [60]. Drug-drug similarities include the following:

Chemical-based Using the chemical development kit (CDK) [85], Perlman et al. [60] computed the hashed fingerprint of each drug based on the canonical SMILES[4] obtained from Drugbank. Considering each fingerprint as a set of elements, they computed the Jaccard similarity of the fingerprints. The Jaccard similarity score between two sets X and Y is

$$\text{Jaccard}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

[3]The number of edges in a shortest path between two vertices.

[4]Simplified Molecular Input Line Entry Specification

Ligand-based Drugs' canonical SMILES obtained from Drugbank are compared against a collection of ligand[5] sets using the similarity ensemble approach (SEA) search tool [56]. A list of relevant protein-receptor families are obtained for each drug, and they computed Jaccard similarity between the corresponding sets of receptor families for each drug pair.

Expression-based The Spearman rank correlation coefficient of gene expression responses to drugs retrieved from the Connectivity Map Project [64, 65] are used as a similarity measure between drugs. The Spearman rank correlation coefficient between two sets X and Y is calculated as

$$\text{Spearman}(X, Y) = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}}$$

where x_i and y_i are ranked elements of X and Y.

Side-effect-based Similarities between drugs are calculated using the Jaccard score between their common side-effects obtained from SIDER [86].

Annotation-based Drugs' ATC codes are obtained from DrugBank and matched against the World Health Organization ATC classification system [87], where drugs are categorized based on different characteristics. They calculated the similarities using the semantic similarity algorithm of Resnik [88].

Target-target similarities include the following:

[5] A substance that binds with a biomolecule to serve a biological purpose.

Sequence-based Perlman et al. [60] compute sequence-based similarities using the Smith-Waterman sequence alignment scores, normalized via the method suggested in [71]—which divides the pairwise score by the geometric mean of the alignment scores of each sequence against itself.

Protein-protein interaction network-based Using an all-pairs shortest path algorithm, they calculated the distance between pairs of genes using their corresponding proteins in the human protein-protein interactions network.

Gene Ontology-based Using the method of Resnik [88], they calculated the semantic similarity measure between Gene Ontology annotations, downloaded from UniProt [89].

3.3.5.2 Evaluation Criteria

We use ten-fold cross validation, where each fold randomly leaves out 10% of the positive and negative (unknown) interactions for testing. We infer interactions and compare against the held-out interactions, measuring performance using the area under the ROC curve (AUC), area under the precision-recall curve ($AUPR$) of the positive class, and the precision of the top n predictions ($P@n$) where $n = 130$ (i.e., the number of positive links held out in each fold) for our evaluations.

AUC is the most commonly reported measure in our related publications and it allows us to compare against the published results of other methods [60, 62, 71] on the same dataset. Lichtnwalter and Chawla [46] discuss different link prediction evaluation methods.

ROC curves are created by plotting the true positive rate versus the false positive rate at various thresholds [90]. Precision-recall (PR) curves are created by plotting the precision (or positive predictive value) versus the recall (or true positive rate) at various thresholds. ROC and PR curves are visually different but they are highly correlated [91], and PR curves are more informative in settings with heavy class imbalance, such as link prediction [46].

Due to this high class imbalance (130 positive to 7,744 negative examples), AUC changes are subtle. We also report AUPR performance and precision of the top 130 predictions which can highlight the importance of each model modification more clearly. This metric is of importance in practice as well, since only the top portion of the predicted interactions are typically actionable for domain experts to further evaluate.

To evaluate our model's performance with cross-validation we used the common method of random sampling of the interactions for hold-outs in each fold. However, we have to assign a value to all the possible grounding to perform weight learning on our PSL model. In order to avoid assigning an arbitrary value to the held-out interactions in each fold we add a dummy predicate (IGNOREDINTERACTS) to the rules only for weight learning. We can avoid grounding of the rules for the held-out interactions using this predicate. For example, we change rule (3.6) to the following form:

$$\begin{aligned}\neg\textsc{IgnoredInteracts}(D, T_2) \land \neg\textsc{IgnoredInteracts}(D, T_1)\\ \land\textsc{SimilarTarget}_{\beta}(T_1, T_2) \land \textsc{Interacts}(D, T_2)\\ \rightarrow \textsc{Interacts}(D, T_1).\end{aligned} \tag{3.14}$$

Although this change negatively affects the performance of our model in the cross-validation setting, we believe it provides an unbiased evaluation and avoids an arbitrary assignment to the held-out variables. This is only an artifact of the cross validation evaluation setting.

3.3.5.3 Analysis Results

We report the results of our five experimental analysis in this section.

Rule Structure We first study the effectiveness of each assumption for predictions. We compare the rules based on triads (3.6 & 3.7) and the rules based on tetrads (3.8). We compare four different settings: the model with only the triad-based rules, the model with only the tetrad-based rules, model with both set of rules, and model with triads and exclusive tetrads. We set the selection parameter (k) to 5 in all models to control the growth of tetrad-based rules, and we learn the weights using separate held-out set of interactions (equal to the size of cross-validation hold-outs) in each fold.

As Table 3.1 shows, the rules inspired by triads are more predictive of the interactions compared to the rules that are based on tetrads. It may be the case that, in a collective setting, triad-based rules capture the effect of tetrad-based rules and perform the best. This experiment not only provides insight into the behavior of prediction using triads and tetrads in this domain, but it also demonstrates how we can easily test such assumptions using PSL's flexibility. One can easily generalize this to quickly evaluate different hypothesis about interactions.

Table 3.1: Comparison of triad-based and tetrad-based rules with $k = 5$

Rules	**AUC**	**AUPR**	**P@130**
Triad-based only	0.920$\pm$0.016	**0.617$\pm$0.048**	**0.616$\pm$0.035**
Tetrad-based only	0.775$\pm$0.023	0.188$\pm$0.029	0.250$\pm$0.033
Triad & tetrad	0.909$\pm$0.015	0.416$\pm$0.047	0.443$\pm$0.025
Triad & excl. tetrad	**0.924$\pm$0.013**	0.560$\pm$0.048	0.588$\pm$0.036

Similarity Selection Next, we study the effect of similarity selection on performance by varying the number of neighbors (k) and measuring the effect on the PSL model based on the triad rules (3.6 & 3.7). We measure the completion time in two settings: first, a setup where we only perform inference, and second, a setting where we run weight learning and inference. Table 3.2 lists the average computation time of ten-fold cross-validation experiments on computers with a (2×4) 2.66 GHz Intel processor and 48GB of RAM.[6] The results show that similarity selection causes significant improvement in processing time.

[6]We used machines with slightly different specifications and under different loads, so the reported times are approximate.

Our similarity selection achieves this speed-up with no significant performance loss. The columns of Table 3.3 lists the performance as we change the selection parameters. The insignificant performance change as we select more restrictively suggests that, even with limited number of similarities (i.e., $k = 5$), our model can produce accurate predictions. AUPR results in Table 3.3 (Inference + W. learning) show that similarity selection sometimes even helps performance and suggests that the similarities with higher values are most predictive of interactions. These results suggest that, in the drug-target interaction domain, models that rely on sparse similarities with high value are often more predictive than the ones that include many similarities with low values. Perlman et al. [60] report relatively similar findings with their own model.

Table 3.2: Speedup with triad-based rules

Condition	**Time to Complete**		
	k=5	k=15	k=30
Inference only	14mins	3h	9.5h
Inference + Weight learning	30mins	6h	22h

Weight learning We study the effect of weight learning by running experiments under two conditions: with all weights set to 5 (arbitrarily hand-tuned) and with weights being learned from a set of observed interactions. Table 3.3 lists the performance improvement of the models with weight learning with different k values for similarity selection.

Table 3.3: Performance variations under the effect of weight learning with triad-based rules

Condition	**AUC** k=5	k=15	k=30
Inference only	0.917±0.017	**0.933±0.014**	0.928±0.016
Inference + W. learning	0.920±0.016	0.931±0.016	0.924±0.019
Condition	**AUPR** k=5	k=15	k=30
Inference only	0.563±0.047	0.578±0.067	0.504±0.061
Inference + W. learning	**0.617±0.048**	0.579±0.062	0.486±0.063
Condition	**P@130** k=5	k=15	k=30
Inference only	0.580±0.042	0.585±0.045	0.532±0.051
Inference + W. learning	**0.616±0.035**	0.594±0.039	0.515±0.037

It is also notable that the performance improvement caused by weight learning is more significant than increasing the number of similarities used as evidence. Although weight learning improves the results, AUC changes are subtle and AUPR and the precision at the top predictions show the improvement more clearly.

Weight learning performance improvement in AUPR and AUC (for $k = 5$) is statistically significant ($p < 0.005$).[7] Figure 3.14 plots the average precision of the top 130 interaction predictions (i.e., P@n) over all ten folds with and without weight learning. It demonstrates how weight learning improves the precision of the predictions, providing steady improvement for the top 130 predictions.

[7] We performed paired one-tailed t-test on the corresponding values of the ten folds.

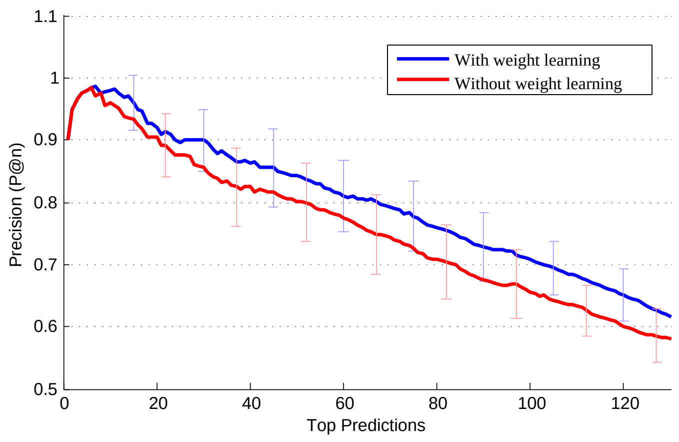


Figure 3.14: Average precision of the top 130 interaction predictions for all 10 folds with $k = 5$.

Figure 3.15 illustrates the average relative weights assigned by PSL to triad-based rules for each similarity. We normalize the weights by dividing their value by the learned prior weight, so the resulting quantity represents how much more heavily the rule is weighted than the prior. It is important to note that neither the absolute nor the relative value of the weights provide precise insight into the predictive power of the rules, since the features and predictions are dependent. Nevertheless, they provide some hints as to how the PSL model makes its joint prediction.

An example of low rule weight and high prediction performance is PPI-network-based similarity (Figure 3.15 and Table 3.4), which produces high accuracy for a single-similarity-based model, but has a low normalized weight. A more accurate method of measuring the effectiveness of each rule (and similarity) for prediction is building models with single rules (as described in the next section) and directly measuring their prediction performance. Even in models with single similarities, rule weight does not correlate with prediction power. Figure 3.15 also shows the rule weights of models with single similarities, which follows the previous trend.

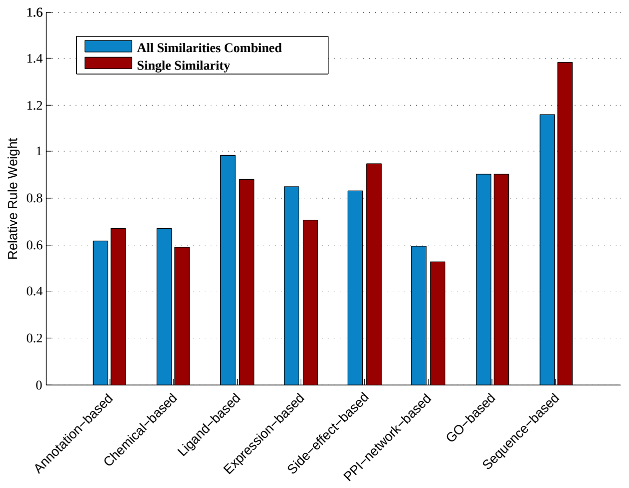


Figure 3.15: Relative triad-based rule weights in models with all similarities included and models with only one similarity.

Combining similarities We study the effect of incorporating multiple heterogeneous drug-drug and target-target similarities in PSL models. Different similarities can replace $\textsc{SimilarTarget}_\beta$ and $\textsc{SimilarDrug}_\alpha$ in rules (3.6) and (3.7) as described in previous sections. For each similarity metric, we add an instance of the rule (3.6) and (3.7) to the PSL model correspondingly. We study the situation where PSL models predict new interactions using only one drug-drug or target-target similarity versus when they are all combined and the results are shown in Table 3.4.

Ligand-based drug-drug similarity and Sequence-based target-target similarity generate the best performance among models using a single similarity. Nevertheless, there is a significant difference between the best single similarity setting (AUC=0.811 ± 0.026 and AUPR=0.516 ± 0.062) and the all-similarities-combined

Table 3.4: Prediction based on one similarity and all similarities combined.

	Similarity	AUC	AUPR	P@130
Drugs	Annotation-based	0.660±0.017	0.224±0.026	0.319±0.026
	Chemical-based	0.670±0.023	0.234±0.042	0.289±0.032
	Ligand-based	0.713±0.023	0.270±0.035	0.337±0.037
	Expression-based	0.540±0.025	0.031±0.009	0.069±0.026
	Side-effect-based	0.631±0.016	0.209±0.032	0.271±0.023
Targets	PPI-network-based	0.781±0.021	0.389±0.047	0.480±0.041
	GO-based	0.611±0.023	0.103±0.027	0.213±0.039
	Sequence-based	0.811±0.026	0.516±0.062	0.574±0.055
	All Similarities	**0.920**±**0.016**	**0.617**±**0.048**	**0.616**±**0.035**

setting (AUC=0.920 ± 0.016 and AUPR=0.617 ± 0.048). This study clearly shows that considering multiple similarities is critical for optimal prediction accuracy and that PSL can efficiently consider the multi-similarity nature of the problem.

Collective inference We list the results from comparing the collective versus non-collective PSL models in Table 3.5. Due to high class-imbalance, AUC does not reflect the change in performance as well as the other measures.

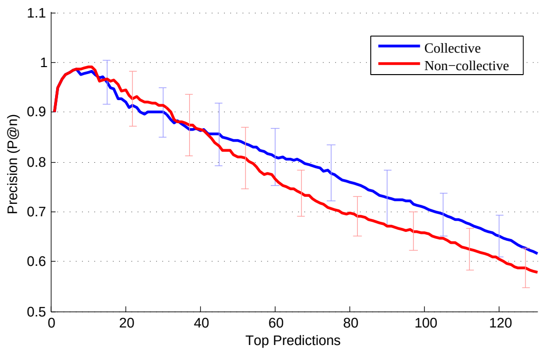
Table 3.5: Effect of collective inference

Condition	AUC	AUPR	P@130
Non-collective inference	0.916±0.016	0.556±0.039	0.577±0.039
Collective inference	**0.920**±**0.016**	**0.617**±**0.048**	**0.616**±**0.035**

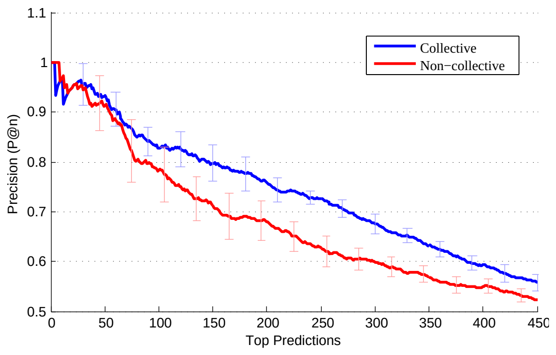
Collective inference performance improvement in AUPR and AUC is statistically significant ($p < 0.005$).[8] Figure 3.16(a) highlights the effect of collective model

[8]We performed paired one-tailed t-test on the corresponding values of the ten folds.

by showing the average (over ten folds) precision of the top 130 predictions. Collective modeling improves the performance in this setting, as it generates a higher overall precision in the top 130 predictions. It is notable that non-collective model is only more effective for the first few predictions. This may be the results of those interactions being predicted based on direct observed evidence. However, collective inference outperforms non-collective setting for higher number of predictions. In addition, collective setting may be more effective when there are more missing links to predict. The significance of collective inference improvement is more clear in Figure 3.16(b) which shows the same experiment with three-fold cross validation and 450 top prediction.



(a) Top 130 prediction via ten-fold cross validation.



(b) Top 450 prediction via three-fold cross validation.

Figure 3.16: Collective vs. non-collective average precision of the top predictions.

3.3.5.4 Predictions Evaluation

Finally, we can compare our results with the reported results of other state-of-the-art methods. Perlman et al. [60] report experimental evaluation on the same dataset. They report state-of-the-art performance and show that their method significantly outperforms those of Yamanishi et al. [62] and Bleakley and Yamanishi [71].

Thus, we compare our model's performance with Perlman et al. [60][9] using the same experimental setting. We report the results based on our folds that contain 130 positives and 7,744 negative links.[10] Table 3.6 lists the results of different PSL models comparing to the Perlman et al. [60].

Table 3.6: Comparison with Perlman's method using ten-fold cross validation

Methods	**AUC**	**AUPR**	**P@130**
Perlman et al. [60]	**0.937**$\pm$**0.018**	0.564$\pm$0.050	0.594$\pm$0.040
PSL triads $k = 5$	0.920$\pm$0.016	**0.617**$\pm$**0.048**	**0.616**$\pm$**0.035**
PSL triads $k = 15$ & excl. tetrads $k = 5$	**0.937**$\pm$**0.012**	0.585$\pm$0.056	**0.616**$\pm$**0.039**

PSL models with triads and selection parameter $k = 5$ score a higher AUPR and P@n. This shows that our predictions have higher precision. Although we argue that AUC in such highly imbalanced settings is not as important, we can match the AUC of the previous state-of-the-art by using a more complicated PSL model with triads and exclusive tetrads and using two different selection parameters of k for each set of rules. Since tetrads generate more groundings, we set k to 5 for tetrads and set k to 15 for triads. Figure 3.17 plots the precision of the top 130 predictions of the PSL model with triad rules and k set to 5 in comparison the predictions of

[9]We implemented the method of Perlman et al. [60] in correspondence with the authors and reproduced their results. Our implementation of their method is also available for download.

[10]The difference between our AUPR results and the ones reported in [60] is due to the downsampling of the unobserved interactions for testing in their paper. We choose our testing folds based on the real ratio of positives to negatives

Perlman et al. [60]. The results show that the PSL model with simple, triad-based rules improves the AUPR and P@n prediction performance of the state-of-the-art methods. We achieve statistically significant improvements in AUPR using the PSL triad model with $k = 5$ over the Perlman's method ($p < 0.005$), and we match the AUC performance of their method using our second PSL model with no significant difference ($p > 0.49$).[11]

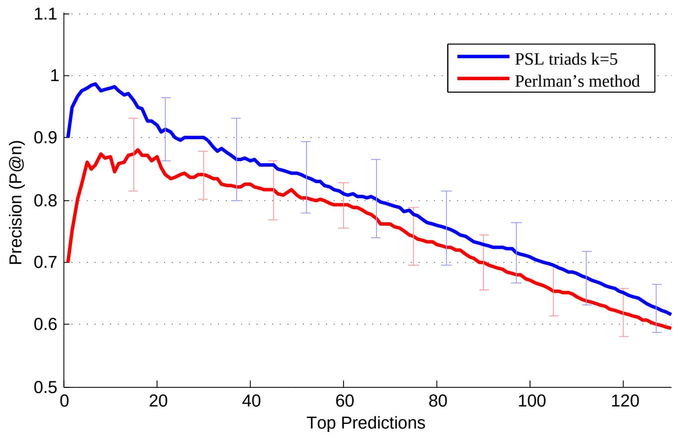


Figure 3.17: Comparing Perlman's method with PSL's top 130 predictions using ten-fold cross validation.

New interaction predictions Additionally, we aimed to evaluate our new interaction predictions by comparing them with new interactions that were not in our initial dataset. Using NodeXL's [84] motif clustering tool, we find targets (or drugs) that share several drugs (or targets). One example is *PTGS1* and *PTGS2*, which is shown on the right side of Figure 3.13.

[11]We performed paired one-tailed t-test on the corresponding values of the ten folds.

Three unobserved interactions in this structure are *Minoxidil-PTGS2*, *Tiaprofenic acid-PTGS1*, and *Oxaprozin-PTGS1*. Our model ranks *Oxaprozin-PTGS1* and *Tiaprofenic acid-PTGS1* as the 46th and 158th most probable interactions out of the 77,444 total interactions, placing them in the top 0.2 percentile of all possible interactions. Since the time the original dataset was collected by Perlman et al. [60], the online databases have been updated. Examining the latest version of Drugbank, the database now contains these two interactions.[12][13] Our model ranked *Minoxidil-PTGS2* significantly lower than the other two at 1,807, and we could not find any indication in the drug-target interaction dataset that this interaction exists.[14]

The two targets are shared between all three drugs in this structure, hence, only the target-target similarities were discriminative. The results show that PSL effectively uses different similarities between *Oxaprozin* and *Tiaprofenic acid*, and the other targets in the structure to rank their interactions higher than the one involving *Minoxidil*.

Furthermore, there are 197 interactions that were added to the Drugbank database since the dataset was collected. We used these newly reported interactions to further evaluate the performance of our models. We generate ten folds consisting of these new interactions and samples of the unobserved interactions, to rank these new interactions against all the other possible predictions. Table 3.7 lists the performance scores of our models and Perlman et al. [60] on the newly reported interactions.

[12]http://www.drugbank.ca/drugs/DB00991
[13]http://www.drugbank.ca/drugs/DB01600
[14]http://www.drugbank.ca/drugs/DB00350

Table 3.7: Comparison with Perlman's method using new interactions

Methods	**AUC**	**AUPR**	**P@130**
Perlman et al. [60]	0.921$\pm$0.016	0.309$\pm$0.014	0.393$\pm$0.018
PSL triads $k = 5$	0.881$\pm$0.001	0.324$\pm$0.008	0.456$\pm$0.017
PSL triads $k = 15$ & excl. tetrads $k = 5$	**0.926$\pm$0.001**	**0.344$\pm$0.018**	**0.460$\pm$0.010**

Although our more complex model demonstrates superior numbers on all performance measures, the predictions made by the PSL model with triads and $k = 5$ are more actionable due to higher precision at the top of the predictions list. That portion of the predictions are the most critical for domain experts to further evaluate. Our more complex PSL model with triads and tetrads achieves higher AUPR due to better recall. Figure 3.18 plots the precision of the top 150 prediction of the PSL model with triads with $k = 5$, the PSL model with triads with $k = 15$ and exclusive tetrads with $k = 5$, and Perlman et al. [60]. The simpler PSL model with triads and $k = 5$ ranks the newly reported interactions higher and performs significantly better than Perlman's method, especially beyond the top 40 predictions. Since there is potential bias in which interactions have been explored *in vitro*, these results, while encouraging, should be interpreted with discretion.

3.3.6 Discussion and Conclusion

In this section, we proposed a model for drug target interaction prediction based on link inference template model described in section 3.2. We used a knn-based similarity selection method and demonstrated how we could combine different similarities and perform collective inference to produce state-of-the-art prediction

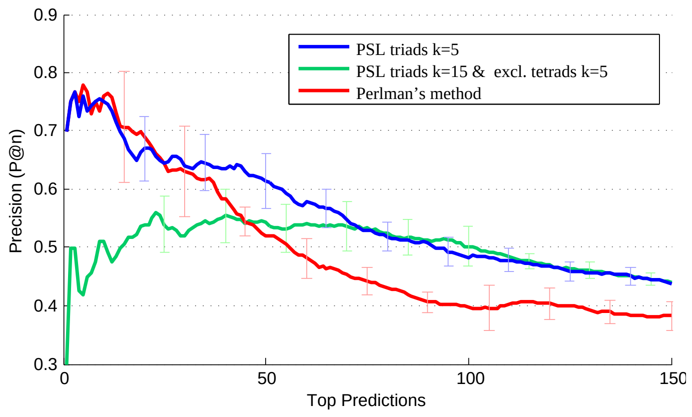


Figure 3.18: Comparing Perlman's method with PSL's top 150 predictions using new interactions.

accuracy for this task. In our experimental evaluation, we studied the effects of collective inference, similarity selection, the combination of similarities, and weight learning in drug target interaction prediction performance prediction. Our results indicate that each of these components plays a positive role, and contribute to improving the performance and efficiency of our model.

3.4 Hybrid Recommender Systems Model

Recommender systems is another examples that we can model as the augmented bipartite structure, where we have users and items as nodes and links between them represent the ratings. Figure 3.19 shows an example of this structure, similarities between the items and between the users can be extracted from different sources. This section describes our proposed hybrid recommender systems framework based on our link inference template model. We show that our link inference template model can use different sources of information in recommender systems.

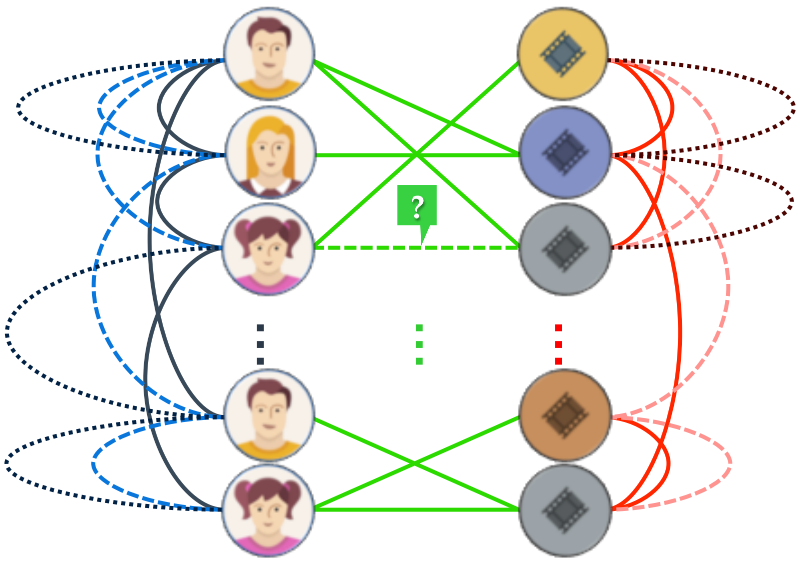


Figure 3.19: Recommender systems as an augmented bipartite structured multi-relational graph.

Our model can encode dependencies between users and items, and ratings, additional information on content and social relations. The model also provides the flexibility to incorporate prior predictions, such as mean-centering priors and the results of other recommender algorithms. The following rules define the model for our hybrid recommender systems framework.

3.4.1 Similarity-based Triad Rules

We use the triad-based structure for our link inference model, due to its superior performance over the general tetrad form. Below are the correspondence of these rules to different methods and intuitions in recommender systems.

User-based Collaborative Filtering: Using the triad structure for similarities in the user side, we can define PSL rules of the following form:

$$\textsc{SimilarUsers}_{sim}(u_1, u_2) \wedge \textsc{Rating}(u_1, i) \Rightarrow \textsc{Rating}(u_2, i) \tag{3.15}$$

This rule captures the intuition that similar users give similar ratings to the same items. The similarities can be calculated with any similarity measure *sim*.

Item-based Collaborative Filtering: Likewise, we can define PSL rules to capture the intuition of item-based collaborative filtering methods, namely that similar items should have similar ratings from the same users:

$$\textsc{SimilarItems}_{sim}(i_1, i_2) \wedge \textsc{Rating}(u, i_1) \Rightarrow \textsc{Rating}(u, i_2) \tag{3.16}$$

The predicate $\textsc{SimilarItems}_{sim}(i_1, i_2)$ is binary, with value 1 iff i_1 is one of the k-nearest neighbors of i_2 (using similarity measure sim), while $\textsc{Rating}(u, i)$ represents the normalized value of the rating of user u to item i. Several methods are available to compute similarities between entities for user-based and item-based methods, and they capture different notions of similarity. We use Pearson's correlation and cosine similarity to calculate similarities between users and items. To improve robustness of the similarities, inspired by Hoff et al. [92], we compute similar users and items in the low-dimensional latent space generated via matrix-factorization using cosine and Euclidean measures. The user similarities are captured in the following rules:

$$\begin{aligned}
\textsc{SimilarUsers}_{cosine}(u_1,u_2) \wedge \textsc{Rating}(u_1,i) &\Rightarrow \textsc{Rating}(u_2,i)\\
\textsc{SimilarUsers}_{pearson}(u_1,u_2) \wedge \textsc{Rating}(u_1,i) &\Rightarrow \textsc{Rating}(u_2,i)\\
\textsc{SimilarUsers}_{\substack{latent\\cosine}}(u_1,u_2) \wedge \textsc{Rating}(u_1,i) &\Rightarrow \textsc{Rating}(u_2,i)\\
\textsc{SimilarUsers}_{\substack{latent\\euclidean}}(u_1,u_2) \wedge \textsc{Rating}(u_1,i) &\Rightarrow \textsc{Rating}(u_2,i)
\end{aligned} \tag{3.17}$$

Using Additional Sources of Information: Incorporating other sources of information about the items, the users, or the respective ratings to our framework is straightforward using the similarity-based template model.

$$\textsc{SimilarItems}_{Content}(i_1,i_2) \wedge \textsc{Rating}(u,i_1) \Rightarrow \textsc{Rating}(u,i_2) \tag{3.18}$$

In this rule, the predicate $\textsc{SimilarItems}_{Content}(i_1,i_2)$ represents items that have similar content-based features (e.g. in the movie recommendation domain such features are the genre, actor, director, etc.), instead of similar ratings. The framework can also incorporate social information when available, and use them as indicators of similarity between users. For instance, we use social network friendship links in the following form:

$$\textsc{Friends}(u_1, u_2) \wedge \textsc{Rating}(u_1, i) \Rightarrow \textsc{Rating}(u_2, i) \tag{3.19}$$

We can also compute similarities based on demographic information and use the following rule:

$$\textsc{SimilarUsers}_{Demo}(u_1, u_2) \wedge \textsc{Rating}(u_1, i) \Rightarrow \textsc{Rating}(u_2, i) \tag{3.20}$$

3.4.2 Ratings Priors

We include two types of priors; mean-centering ratings, and predictions from other methods.

Mean-Centering Priors: We include rules that penalize the divergence of the value of the ratings from the corresponding averages, for each user and each item:

$$\begin{gathered}
\textsc{AverageUserRating}(u) \Rightarrow \textsc{Rating}(u, i) \\
\neg\textsc{AverageUserRating}(u) \Rightarrow \neg\textsc{Rating}(u, i) \\
\textsc{AverageItemRating}(i) \Rightarrow \textsc{Rating}(u, i) \\
\neg\textsc{AverageItemRating}(i) \Rightarrow \neg\textsc{Rating}(u, i)
\end{gathered} \tag{3.21}$$

where $\textsc{AverageUserRating}(u)$ represents the average of the ratings for the set of items that user u provided in the training set. Similarly, $\textsc{AverageUserRating}(i)$

represents the average of the user ratings an item i has received. In order to capture cases where we have no information about a user or an item, we use a general prior on ratings centered at the mean value of all of the ratings in the system (i.e. the average over all items rated by all users) with the following rules:

$$\begin{aligned} \textsc{PriorRating} &\Rightarrow \textsc{Rating}(u, i) \\ \neg\textsc{PriorRating} &\Rightarrow \neg\textsc{Rating}(u, i) \end{aligned} \tag{3.22}$$

were the predicate PRIORRATING represents the average of all ratings.

Prections From Other Methods: We can also use predictions from other methods as priors for our model. Therefore, instead of selecting a single recommendation algorithm, we can incorporate the predictions from different methods into our unified model. For example, the predictions from matrix factorization (optimizing regularized squared error via stochastic gradient descent) (MF), Bayesian probabilistic matrix factorization (BPMF) [93], and item-based collaborative filtering can be incorporated in the model via the following rules:

$$\begin{aligned} \textsc{Rating}_{MF}(u, i) &\Rightarrow \textsc{Rating}(u, i) \\ \neg\textsc{Rating}_{MF}(u, i) &\Rightarrow \neg\textsc{Rating}(u, i) \\ \textsc{Rating}_{BPMF}(u, i) &\Rightarrow \textsc{Rating}(u, i) \\ \neg\textsc{Rating}_{BPMF}(u, i) &\Rightarrow \neg\textsc{Rating}(u, i) \\ \textsc{Rating}_{\substack{item\\based}}(u, i) &\Rightarrow \textsc{Rating}(u, i) \\ \neg\textsc{Rating}_{\substack{item\\based}}(u, i) &\Rightarrow \neg\textsc{Rating}(u, i) \end{aligned} \tag{3.23}$$

3.4.3 Experimental Validation

We compared our model with the state-of-the-art recommender systems on two standard datasets. We used a subset of the Yelp academic dataset[15] and the Last.fm dataset[16]. Our goal in the Yelp dataset is to recommend local businesses to users by predicting the missing (held out) ratings of businesses based on previous ratings. For the Last.fm dataset, our goal is to recommend artists to users. As Last.fm does not provide explicit user-artist ratings we use the number of times a user has listened to an artist to construct implicit ratings. Table 3.8 provides detailed information about the dataset.

Table 3.8: Dataset Description

Dataset	**Yelp**	**Last.fm**
No. of users	34,454	1,892
No. of items	3,605	17,632
No. of ratings	99,049	92,834
Content	514 business categories	9,719 artist tags
Social	81,512 friendships	12,717 friendships
Sparsity	99.92%	99.72%

We evaluate the performance of different models with 5-fold cross-validation and report the average cross-validated error.

We study the performance of our model in comparison to several state-of-the-art models. We considered the following baselines:

[15]https://www.yelp.com/academic_dataset
[16]http://grouplens.org/datasets/hetrec-2011/

Table 3.9: Overall Performance of Different Recommender Systems on Yelp and Last.fm.

		Yelp		Last.fm	
	Model	**RMSE (SD)**	**MAE (SD)**	**RMSE (SD)**	**MAE (SD)**
Base models	Item-based	1.216 (0.004)	0.932 (0.001)	1.408 (0.010)	1.096 (0.008)
	MF	1.251 (0.006)	0.944 (0.005)	1.178 (0.003)	0.939 (0.003)
	BPMF	1.191 (0.003)	0.954 (0.003)	1.008 (0.005)	0.839 (0.004)
Hybrid models	Average	1.179 (0.003)	0.925 (0.002)	1.067 (0.004)	0.857 (0.004)
	BPMF-SRIC	1.191 (0.004)	0.957 (0.004)	1.015 (0.004)	0.842 (0.004)
	Our model	**1.173** (0.003)	**0.917** (0.002)	**1.001** (0.004)	**0.833** (0.004)

- Item-based [94] collaborative filtering, using Pearson's correlation with a mean-centering correction, implemented in Graphlab Create.[17]

- Matrix factorization (MF) using stochastic gradient descent [48], implemented in Graphlab Create.

- Bayesian probabilistic matrix factorization (BPMF): The Bayesian variant of probabilistic matrix factorization, trained using Gibbs sampling [93].

- A Naive hybrid method where the predictions of the models are simply averaged.

- BPMF with social relations and items' content (BPMF-SRIC), which is a hybrid model that extends BPMF with social and content information [95].

The performance of our model is statistically significantly better than the baselines at $\alpha = 0.05$ for both datasets and evaluation metrics when using paired

[17] http://www.dato.com

t-test. We denote the numbers that are statistically significantly better with a bold font. Table 3.9 shows the performance of different methods on the two datasets.

Chapter 4: Dynamic Similarity Selection for Link Inference

As discussed in section 3.3.3 the number of grounded rules in the triad-based rules with HL-MRFs and PSL for link prediction and recommender systems, increases rapidly with the number of similarities. In section 3.3.5 I showed that selecting the right number of similarities to include in the model is essential in improving the performance of the model and reducing the inference and learning time. I already discussed the limitations of using a simple threshold in section 3.3.3, and used the k-highest similarities in the drug-target interaction prediction model and the recommender systems setting. This approach is used in traditional neighborhood-based link prediction and collaborative filtering.

However, in the collective setting, similarities play a significant role in information propagation in the network between the links, even through the different kinds of similarities. In sparse networks, this role becomes more important because most links are not connected to observed evidence, and predictions about them are mostly based on inferred information for other links. Selecting which similarities to include in the model for each node is not a trivial task. In this chapter I provide a generalization of our link inference template based on *neighborhood graphs* and propose a dynamic similarity selection framework for it.

4.1 Neighborhood Graph-based Methods

Many predictive models use the dependencies between instances to perform inference. The simplest model of this form is the *k-nearest neighbors* classifier which uses similarity as a simple form of dependency between instances, and assigns a label to an unknown instance based on the most similar instances with known labels. Graph-based semi-supervised learning methods extend this approach to propagate predictions using the similarities between the instances with unknown labels as well. More complex multi-relational models can incorporate different types of dependencies between all instances and perform joint inference on their unknown labels.

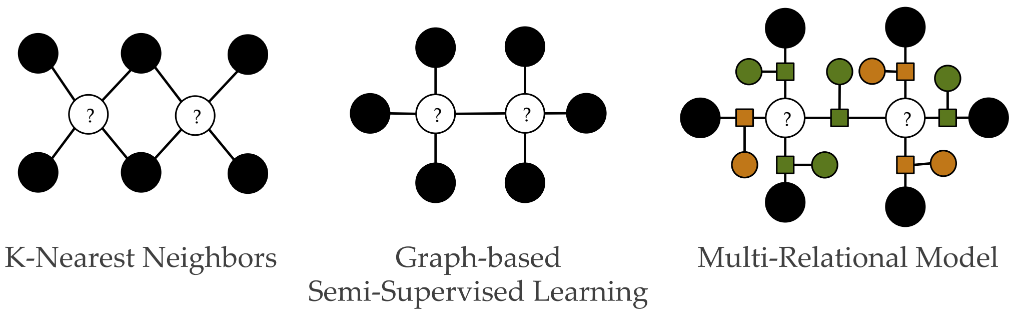


Figure 4.1: Schematic view of neighborhood graph-based methods.

These types of models are used in several domains such as collaborative filtering, link prediction, and image classification. For example, when determining characteristics of individuals, the characteristics of the people who are most similar to them, their friends, family, and co-workers, can all influence a model's prediction. *Neighborhood graphs* which capture interdependencies between instances, are the underlying structure of reasoning in many of these predictive models. In these models the data is represented as a *neighborhood graph*, where nodes are instances and weighted edges represent relations (e.g., similarities) between them.

Most methods that make predictions based on a neighborhood graph follow three basic operations [96]: Graph *Generation*, *Selection* and *Inference*. The first step, is the *candidate graph generation*, which often includes defining the relations or similarities between instances. The process of constructing the candidate graph is generally problem-specific. If the original input data is *relational*, or in the form of a graph (which we call *data graph*), some of the explicit relations such as relationships in a social network, or adjacencies in an image may be used as an approximation of the affinity or dependency of instances. When the original input data includes feature vector representation of instances, a similarity or kernel function is defined to estimate their pairwise affinities.

Similar to k-nearest neighbors classifiers where only the k most similar instances are useful in the model, not all of the dependencies generated in the candidate graph are helpful. The abundance of pairwise similarities or relations often hinders a model's scalability as well as its predictive performance [97] and makes the candidate graph generally unsuitable for modeling approaches. Hence, the next step is *selection*, reducing the candidate graph by pruning similarities or relations to a more manageable neighborhood graph. Examples of these methods that are often considered a pre-processing step to inference include k-nearest neighbors, ϵ-neighborhood selection, and b-matching [96].

The *third step* is performing inference using the neighborhood graph. Methods such as Mincut, graph random walk, Gaussian random fields, local and global consistency, spectral graph transducer, manifold regularization, and label propagation are examples that perform inference on the neighborhood graph [98].

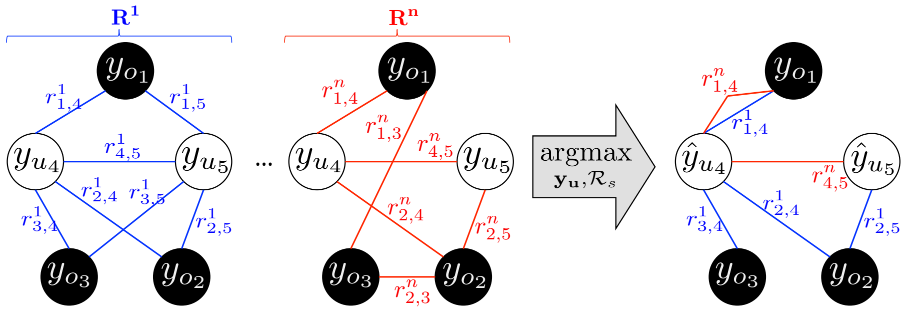


Figure 4.2: Neighborhood graph construction from candidate graph with several relations. For simplicity of notation, items are represented only with their labels y. Observed labels y_o are shown as black circles and instances with unobserved labels y_u are shown with white circles and annotated accordingly. Relations $r_{\mathbf{x}_i,\mathbf{x}_j}$ are noted as $r_{i,j}$. The neighborhood graph shown on the right has a subset of selected relations with different types and also the inferred values ($\hat{y}$) for the unobserved labels.

4.2 Neighborhood Graph Construction

There are several challenges in this sequential process for constructing the neighborhood graph especially for *multi-relational* models and data:

- Most methods are designed to handle a single similarity, dependency, or relation type when constructing the neighborhood graph. However, there are often multiple relations than each can serve as a noisy approximation for the affinity that is important for the predictive task at hand. Constructing a *multi-*

relational neighborhood graph that can effectively combine different affinity signals from multiple sources is highly important.

- The static nature of the pruning methods is blind to the state of instances with unknown labels. However, the interdependencies between these unknown variables are the essence of the semi-supervised and multi-relational models. Hence, the static neighborhood graphs can not adequately capture the similarities between the most important unknown variables.
- The pruning process often solely relies on the value assigned the similarity or relation and does not consider the characteristics of instances that are being connected in the neighborhood graph. For example a slightly lower similarity to an instance with a rare label can be more informative than a high similarity to an instance of a majority class.

In this chapter, we address these challenge by developing a framework to dynamically construct a neighborhood graph. Our framework interleaves inference and neighborhood graph construction, efficiently using multi-relational data while maintaining scalable performance. We have developed the **LINA** framework, which consists of four parts; **L**earning the relative importance of each relation in multi-relational settings, **I**nferring labels for interrelated instances, **N**ominating instances that can benefit from additional relations in the neighborhood graph, and **A**ctivating new relations between instances in the neighborhood graph to improve the inference results. The main contribution in this chapter include:

- We propose a general unified framework to actively learn multi-relational neighborhood graphs and demonstrate its use for collective link prediction.
- We outline a general multi-relation model used for link prediction and regression [52, 97, 99] as an inference problem on a neighborhood graph.

In Section 4.3 we formally define the problem statement, and in Section 4.4 we describe our proposed framework. Section 4.6 includes discussion on modeling the link prediction task with neighborhood graph. We then present our nomination, activation, and learning method in Section 4.5 and provide experimental results to validate our approach in Section 4.7. Finally, we discuss related methods to our approach in different domains in Section 4.8.

4.3 General Problem Statement

Let candidate graph be represented via $\mathcal{G}_c \triangleq \langle \mathbf{X}, \mathcal{R} \rangle$, where vertices $\mathbf{X}$ is a set of n data points $\mathbf{x}_i = \left[x_i^1, \ldots, x_i^{p-1}, y_i\right]$, and $\mathbf{y}$ represent the set of all labels ($\mathbf{y} = \{y_1, \ldots, y_n\}$) where a subset is observed ($\mathbf{y_o}$) and others are unobserved ($\mathbf{y_u}$), and edges $\mathcal{R}$ are a set of relations $r_{\mathbf{x}_i,\mathbf{x}_j}$ (e.g., pairwise similarities, or connections in social network) connecting pairs of data points $\mathbf{x}_i$ and $\mathbf{x}_j$, based on some notion of affinity. Graph-based methods generally make the assumption that for two data points $\mathbf{x}_i$ and $\mathbf{x}_j$ that are connected in this candidate graph $\mathcal{G}_c$, their labels y_i and y_j are close to each other, where the strength of this assumption depends on the value or weight associated with the relation $r_{\mathbf{x}_i,\mathbf{x}_j}$.

Models that perform collective inference [2] based on both known and unknown labels find an optimal state of all unknowns variables by optimizing an objective function f over all target variables $\mathbf{y_u}$ and jointly assigning values to all of them. Thus, collective methods propagate inferred values of the labels based on the relations on the candidate graph. Formally:

$$\hat{\mathbf{y}}_\mathbf{u} = \arg\operatorname*{opt}_{\mathbf{y_u}} f(\mathbf{y_u}, \mathbf{y_o}, \mathcal{G}_c; \boldsymbol{\omega})$$

e.g., in a probabilistic setting:

$$\hat{\mathbf{y}}_\mathbf{u} = \arg\max_{\mathbf{y_u}} \mathrm{P}(\mathbf{y_u}|\mathbf{y_o}, \mathcal{G}_c; \boldsymbol{\omega}) \tag{4.1}$$

where, $\mathbf{y_u}$ and $\mathbf{y_o}$ represent the unobserved and observed labels, $\hat{\mathbf{y}}_\mathbf{u}$ indicates the inferred value for the unobserved labels, and $\boldsymbol{\omega} = \{\omega_1, \ldots, \omega_m\}$ represents the model parameters.

Furthermore, we are interested in settings where $\mathcal{R}$ can be of different types (i.e., $\mathcal{R}= \{\mathbf{R^1}, \ldots, \mathbf{R^n}\}$), for example we can have pairwise similarities computed based on different methods or features, or observed relations in a social network with different semantic (friendship, follow, etc.).

The task of interest in this section is given an *activation quota* $q_\tau \geq 1$ to dynamically select a subset $\mathcal{R}_s$ (such that $|\mathcal{R}_s| \leq q$) from all known relations $\mathcal{R}$ to improve the performance of inference. We call the reduced candidate graph with less relations a *neighborhood graph* ($\mathcal{G}_n$). We select some of the relations $r_{\mathbf{x}_i,\mathbf{x}_j}$ from each relation type $\mathbf{R}^k$ form candidate graph ($\mathcal{G}_c$) to include in the neighborhood graph ($\mathcal{G}_n = \langle \mathbf{X}, \mathcal{R}_s \rangle$) and discard the rest of relations. Figure 4.2 shows an overview of our main task.

More formally we aim to find an *activation methods* τ (i.e., inclusion function or map) such that

$$\mathcal{R}_s = \tau(\mathcal{R}, \mathbf{y}, q_\tau, \boldsymbol{\omega}) \tag{4.2}$$

and $|\mathcal{R}_s| \leq q_\tau$, so that using a graph $\mathcal{G}_n \triangleq \langle \mathbf{X}, \mathcal{R}_s \rangle$ with $\mathcal{R}_s$ instead of $\mathcal{R}$ improves the inference's result, i.e.,

$$\hat{\mathbf{y}}_\mathbf{u} = \underset{\mathbf{y}_\mathbf{u}, \mathcal{R}_s}{\arg\max}\, \mathrm{P}(\mathbf{y}_\mathbf{u} | \mathbf{y}_\mathbf{o}, \langle \mathbf{X}, \mathcal{R}_s \rangle; \boldsymbol{\omega}) \tag{4.3}$$

Note that we replaced $\mathcal{G}_c$ in (4.1) with $\mathcal{G}_n$. Next, we describe our proposed methods to maximize the objective in (4.3).

4.4 Proposed General Framework

Maximizing the objective in (4.3) is a non-convex combinatorial optimization problem. Hence, we break this objective into three parts and iteratively perform each part; one part to *nominate* instance ($\mathbf{X_n}$) with unknown labels that require more evidence, other part to *activate* or select more relations for those nominated instances ($\mathcal{R}_s$), and the third part to jointly *infer* all unknown labels for instances given the results from the two previous steps. To achieve this we introduce a *nomination method* η such that $\mathbf{X_n} = \eta(\mathbf{X}, \hat{\mathbf{y}}_\mathbf{u}, q_\eta)$ and $|\mathbf{X_n}| \leq q_\eta$ where q_η indicates the *nomination quota*. We introduce the nomination quota to control the number of unknown variables that the selected similarities in each iteration focus on. For simplicity we assume that the *activation quota* is related to the *nomination quota* by a constant κ (i.e. $q_\tau = \kappa \times q_\eta$), which means for each nominated instance we can activate up to a maximum of κ relations r. We then update the activation method in (4.2) to select relations only for the nominated instances such that $\mathcal{R}_s = \tau(\langle \mathbf{X}_n, \mathcal{R}\rangle, \mathbf{y}, q_\tau, \boldsymbol{\omega})$

Then (4.3) will be approximated via three components of nomination, activation, and inference as follows:

$$\begin{aligned}
&\mathbf{X_n} = \eta(\mathbf{X}, \hat{\mathbf{y}}_\mathbf{u}, q_\eta)\\
&\mathcal{R}_s = \tau(\langle \mathbf{X}_n, \mathcal{R}\rangle, \mathbf{y}, q_\tau, \boldsymbol{\omega})\\
&\hat{\mathbf{y}}_\mathbf{u} = \arg\max_{\mathbf{y_u}} \mathrm{P}(\mathbf{y_u}|\mathbf{y_o}, \langle \mathbf{X}, \mathcal{R}_s\rangle; \boldsymbol{\omega})
\end{aligned} \tag{4.4}$$

Due to obvious dependencies between $\mathbf{X_n}, \mathcal{R}_s,$ and $\hat{\mathbf{y}}_\mathbf{u}$, we developed an iterative algorithm to preform these steps and update the assignments. Algorithm 1 shows the overall code performing each step.

Algorithm 1 LINA Framework

```
1: $\mathcal{R}_s \leftarrow \mathcal{R}_{\text{init}}$
2: $\boldsymbol{\omega} \leftarrow$ Learn the model parameters based on $\mathcal{R}_{\text{init}}$
3: $\mathcal{R}_s \leftarrow \emptyset$
4: for $i \in 0 \ldots$ total iterations do
5:     $\mathbf{y_u} \leftarrow \arg\max \mathrm{P}(\mathbf{y_u} | \mathbf{y_o}, \langle \mathbf{X}, \mathcal{R}_s \rangle; \boldsymbol{\omega})$
6:     $\mathbf{X_n} \leftarrow \eta(\mathbf{X}, \mathbf{y_u}, q_\eta)$
7:     $\mathcal{R}_s \leftarrow \tau(\langle \mathbf{X}_n, \mathcal{R} \rangle, \mathbf{y}, q_\tau, \boldsymbol{\omega})$
8: return $\mathbf{y_u}$
```

We discuss the *nomination method* τ and the *activation method* η in Section 4.5. To learn the importance ($\boldsymbol{\omega}$) of each relation type $\mathbf{R}^k$ and perform inference we use *Probabilistic Soft Logic (PSL)* and *Hinge-loss Markov Random Fields (HL-MRFs)* [38] learning and inference methods.

4.5 Methods

The methods introduced in this chapter generally apply to multi-relational probabilistic models that perform joint inference on all unknown variables. We use PSL and HL-MRFs for inference and parameter learning of our framework. Our choice of HL-MRFs comes from technical considerations: MPE inference in HL-MRFs is provably and empirically efficient, in theory growing $O(N^3)$ with the number of potentials, N, but in practice often converging in $O(N)$ time [38]. Models built using HL-MRFs have achieves state-of-the-art performance in a variety of applications [52, 97, 99, 100]. The MPE inference in HL-MRFs is a convex optimization problem solved efficiently with Alternating Direction Method of Multiples algorithm. We use the PSL templating language to built a simple model capturing dependencies between instance labels as following:

$$\omega_{k_1}: \quad \textsc{Label}(\mathbf{x}_i) \wedge \textsc{Relation}_k(\mathbf{x}_i, \mathbf{x}_j) \Rightarrow \quad \textsc{Label}(\mathbf{x}_j) \tag{4.5}$$

$$\omega_{k_2}: \neg\textsc{Label}(\mathbf{x}_i) \wedge \textsc{Relation}_k(\mathbf{x}_i, \mathbf{x}_j) \Rightarrow \neg\textsc{Label}(\mathbf{x}_j) \tag{4.6}$$

The HL-MRFs model built using the above rules assigns values to $\textsc{Label}(\mathbf{x}_i)$ and $\textsc{Label}(\mathbf{x}_j)$ that are close to each other both in boolean and continuous forms. We use one set of rules for each $\textsc{Relation}_k$ in the dataset to support multi-relational data. PSL support different methods to learn the rule weight ω_k including via maximum likelihood estimation [38]. As we discuss in Section 4.5.2.4, we use the learned PSL rule weight ω_k as an approximation of the importance of the $\textsc{Relation}_k$. In this section we discuss our approach for nomination and activation method and explain how we leverage the optimization terms and model parameters in our proposed methods.

4.5.1 Nominating Instances

The nomination phase of our framework selects instances that may benefit from additional evidence. The process of nominating instances is similar to the problem of active learning [101], where instances are labeled based on a utility function. However, in contrast to active learning, nomination does not acquire labels, but selects those instances for which we introduce new relations in the neighborhood graph.

Our general framework is compatible with arbitrary nomination techniques, allowing us to leverage the diverse active learning strategies developed over the past decades. Furthermore, our choice of HL-MRFs for modeling inference problems provides the opportunity to use unique nomination strategies that incorporate partial inference outputs and the variable state. We present a nomination strategy that uses inference context and model features to choose instances.

4.5.1.1 Disagreement-based Uncertainty

A useful signal of uncertainty for an instance label can be when existing relations selected for the instance cause disagreements on the inferred values. For example, an inferred binary label of 0.5 that is the result of two conflicting evidence suggesting the value should be 0 and 1 is much more uncertain than if both evidence suggest it should be 0.5. For a simple k-nearest neighbors classifiers this uncertainty can be measured based on the variance of the evidence.

For HL-MRFs we build on a method from Pujara et al. [102] for online inference, which derives features for variable selection from the optimization process underlying inference. This method identifies features from the underlying consensus optimization algorithm for HL-MRFs inference, the alternating direction method of multipliers (ADMM) [38]. ADMM decomposes the optimization objective into independent subproblems, optimizes each subproblem independently, and introduces a constraint that the all subproblems agree on the optimal value of each inferred variable. This optimization is often expressed using the augmented Lagrangian seen in (4.7). Here, ω_r and ϕ_r are the parameter and potential associated with a given relation r, $\tilde{y}_r$ is the local optimizer of a subproblem and y_r is the consensus estimate. Consensus between subproblems is enforced by introducing a Lagrange multiplier, α_r, associated with the constraint, and increasing the optimization penalty, ρ, associated with violating this constraint to guarantee eventual convergence.

$$\min_{\tilde{y}_r} \; \omega_r \, \phi_r(\mathbf{x}, \tilde{y}_r) + \frac{\rho}{2} \left\| \tilde{y}_r - y_r + \frac{1}{\rho} \, \alpha_r \right\|^2 \tag{4.7}$$

The disagreement between evidence can be expressed in terms of the Lagrange multipliers associated with each optimization term. At convergence, the value of this Lagrange multiplier captures the disagreement of a given optimization term with the consensus estimate. We use the average weighted Lagrange multiplier (AWL) [102], that measures the overall discrepancy between the local and consensus copies of the label as a measure of uncertainty for the inferred labels:

$$\frac{1}{|R|} \sum_{r \in R} \omega_r \alpha_r(i) \tag{4.8}$$

where R indicates all the local copies of a consensus variable.

4.5.2 Activating Relations

In the relation activation step, we select a subset of relations $\mathcal{R}_s$ from the candidate graph $\mathcal{G}_c$ to include in the neighborhood graph $\mathcal{G}_n$. In addition to considering the weight, or value, of each relation edge $r^k_{\mathbf{x}_i,\mathbf{x}_j}$, we consider structural properties of the neighborhood graph. Intuitively, these structural properties measure the informativeness of a relation for inferring multiple unknown instances, and their abilities to effectively propagate the information through $\mathcal{G}_n$. In many applications relations are shared between multiple instances. For example, in drug-target networks, a similarity between two targets can be used to predict interactions for several drugs. We introduce features that considers the number of instances with known and unknown labels incident on a relation. First we introduce and define two additional structural features along with the relation value feature. Then, we combine these features and select the top q_τ relations from multiple relation types.

4.5.2.1 Value Feature

We use strength or value associated with a relation edge $r^k_{\mathbf{x}_i,\mathbf{x}_j}$ as a basic feature. Relations of higher value convey a greater dependence between assignments to labels of instances $\mathbf{x_i}$ and $\mathbf{x_j}$. If $\mathbf{x_i}$ or $\mathbf{x_j}$ is an instance with known label, then a high valued $r^k_{\mathbf{x}_i,\mathbf{x}_j}$ effectively propagates that label to the unknown instance.

4.5.2.2 Nominated Instance Count Feature

For each $r^k_{\mathbf{x}_i,\mathbf{x}_j}$ in $\mathcal{G}_n$, we compute the number of nominated unknown instances that are incident upon $r^k_{\mathbf{x}_i,\mathbf{x}_j}$. We introduce an incident operator $\mathcal{I}(x_\alpha, r_\beta)$ that returns 1 if x_α is an endpoint of $r^k_{\mathbf{x}_i,\mathbf{x}_j}$ and 0 otherwise. Formally, the nominated instance count score for $r^k_{\mathbf{x}_i,\mathbf{x}_j}$ is:

$$\sum_{\mathbf{x}_\alpha \in \mathbf{X}_n} \mathcal{I}\left(\mathbf{x}_\alpha, r^k_{\mathbf{x}_i,\mathbf{x}_j}\right)$$

where $\mathbf{X}_n$ are the nominated instances incident on $r^k_{\mathbf{x}_i,\mathbf{x}_j}$. Intuitively, if many $\mathbf{x}_\alpha \in \mathbf{X}_n$ have endpoints in $r^k_{\mathbf{x}_i,\mathbf{x}_j}$, then the relation will be informative for many predictions and introduce multiple useful dependencies in the inference step.

4.5.2.3 Observed Instance Count Feature

For each relation $r^k_{\mathbf{x}_i,\mathbf{x}_j}$, we also compute the number of instances with *observed* labels (i.e., $\mathbf{y_o}$) that are incident on $r^k_{\mathbf{x}_i,\mathbf{x}_j}$. Connecting more instances with observed labels are important because they can propagate evidence to instances with unknown labels through these activated relations. Formally, the observed feature score for $r^k_{\mathbf{x}_i,\mathbf{x}_j}$ is:

$$\sum_{\mathbf{x}_\alpha, y_\alpha \mid y_\alpha \in \mathbf{y_o}} \mathcal{I}\left(\mathbf{x}_\alpha, r^k_{\mathbf{x}_i,\mathbf{x}_j}\right)$$

For this score we only count the instances with observed labels incident on $r^k_{\mathbf{x}_i,\mathbf{x}_j}$. Hence, relations with higher scores increase connectivity of instances with known labels.

4.5.2.4 Combining Features and Selecting Relations

Each of the proposed features above assigns a score to each relation. In our framework, first for each nominated instance $\mathbf{x}_i \in \mathbf{X}_n$ we consider the set of relations $\mathcal{R}_n = \{r_\beta | \mathcal{I}(\mathbf{x}_i, r_\beta)\}$ to which the nominated instance $\mathbf{x}_i$ is incident on. Then for each $r^k_{\mathbf{x}_j,\mathbf{x}_l} \in \mathcal{R}_n$, we compute the score for each feature and use their products for combination, which we denote as $s^k_{\mathbf{x}_j,\mathbf{x}_l}$. We rank the relations in $\mathcal{R}_n$ by their weighted combined scores $\omega_k \times s^k_{\mathbf{x}_j,\mathbf{x}_l}$ where ω_k is the *parameter*, or importance, of relation type k learned via the HL-MRFs framework. For each nominated instance $\mathbf{x}_i$, we select the top κ relations from $\mathcal{R}_n$. And since $q_\tau = \kappa \times q_\eta$, where q_η is the number of nominated instances, the total activation quota is never exceeded.

4.6 Neighborhood Graph-based Link Prediction

While the framework proposed in this chapter is generally applicable to all neighborhood graph-based models, we focus our arguments on the link prediction task.

To apply a neighborhood graph-based learning method for a link prediction-task, links should be represented as instances (i.e., nodes) in the candidate graph $\mathcal{G}_c$ and similarities between the links should be captured as relations (i.e., edges) between them. In other words, we convert the links of the original data to nodes in the candidate graphs. More formally, let $\mathcal{G}_d \triangleq \langle \mathcal{V}, \mathcal{E} \rangle$ be the original data graph where $\mathcal{V}$ is the set of vertices, and $\mathcal{E}$ is the set of edges or links. In a multi-relational network, vertices and edges can be of different types. For example in a drug-target interaction network, $\mathcal{V}$ is the set of all drugs and protein targets and $\mathcal{E}$ is the set of all the drug-target interactions as well as similarities with different semantics between drugs and between targets. Similarities can be extracted from different sources, for example, based on chemical structures of the drugs, or nucleotide sequence of the targets [97].

In such setting, the candidate graph $\mathcal{G}_c \triangleq \langle \mathbf{X}, \mathcal{R} \rangle$ consists of instances $\mathbf{X}$ which are a subset of links $\mathcal{E}$, and similarities $\mathcal{R}$ which are derived from $\mathcal{G}_d$ based on a modeling decision. For example, Kashima et al. [103] use the *Kronecker sum and product* to derive similarity between links, and we defined the similarities between links based on the similarities between their end nodes in our link inference template via the rules:

$$\begin{aligned} \textsc{Similar}(v_2, v_3) \wedge \quad \textsc{Link}(v_1, v_2) &\Rightarrow \quad \textsc{Link}(v_1, v_3) \\ \textsc{Similar}(v_2, v_3) \wedge \neg\textsc{Link}(v_1, v_2) &\Rightarrow \neg\textsc{Link}(v_1, v_3) \end{aligned} \tag{4.9}$$

Figure 4.3 shows an example of creating a candidate graph for links based on the *data graph* using our method. In this example, the original data graph has three types of edges (e^i, e^j, e^k) and we are interested to infer the values of $e^i_{1,2}$ and $e^i_{1,4}$ based on the observed values of other edges. Note that in this setting a similarity is shared between several links. e.g., $\mathbf{x}_p = e^i_{1,2}$, $\mathbf{x}_q = e^i_{1,4}$ and $r^j_{\mathbf{x}_p,\mathbf{x}_q} = r^j_{e^i_{1,2},e^i_{1,4}} = e^j_{2,4}$.

For $\mathcal{G}_n$ in this link prediction setting, the incident operator $\mathcal{I}(x_\alpha, r_\beta)$ that we use for activation functions will be the following:

$$\mathcal{I}\left(\mathbf{x}_\alpha = e^i_{s,t}, r^j_\beta = e^j_{u,v}\right) = \begin{cases} 1 & \text{if } \{s,t\} \cap \{u,v\} \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$$

Link instance $\mathbf{x}_\alpha$ is incident on r^j_β (i.e.,$e^j_{u,v}$) if it has an end point of either u or v. Intuitively, links are incident on a relations if the share a node.

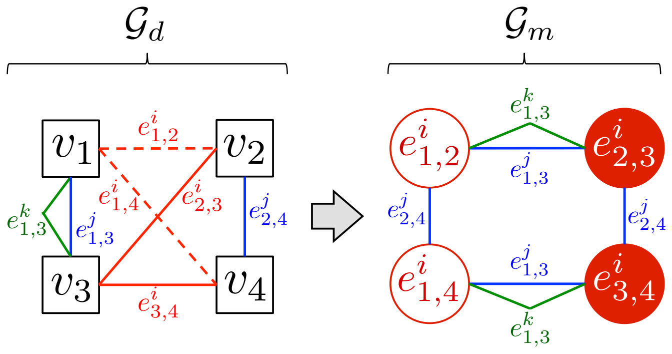


Figure 4.3: *candidate graph* construction based on the *data graph*. Using the logic shown in (4.9), the similarity between two links that share a node are based on the similarity between their other end nodes. For example, similarity between $\mathbf{x}_p = e^i_{1,2}$ and $\mathbf{x}_q = e^i_{2,3}$ that share v_2 is based on the similarities of v_1 and v_3 which are $e^j_{1,3}$ and $e^k_{1,2}$.

4.7 Experimental Validation

In this section we evaluate the main components of our framework, activation and nomination methods, on the link prediction setting. The task is to predict the held out set of interactions in the network, based on the partially observed interactions and the similarities between nodes. We use 10-fold cross validation for our experiments where we hold out 10% of the observed links (i.e., positive class) and use the rest as observed instance to predict their values. We also sample 10% of the absent or missing links (i.e., negative class) and include them in each held out fold. We use the link prediction model presented at Chapter 3 for drug-target interaction prediction. The underlying neighborhood graph was build using a sequential process with a predefined k-nearest neighbors approach. The following sections describes the dataset and present our experimental results.

4.7.1 Dataset

We perform our experiments to predict new interactions between drug compounds and target proteins using the setting of Chapter 3. We follow the link prediction modeling approach described in Section 4.6. We use known interactions and biologically relevant similarity relations to predict held-out interactions. We describe the interactions and similarities used for our experimental evaluation below.

The interactions between drugs and target are gathered from Drugbank, KEGG Drug, Drug Combination Database (DCB) and Matador. The dataset includes 1,306 known interactions between 315 drugs and 250 targets. We use five types of similarity between each pair of drugs and three types of similarity for each pair of targets. We describe each briefly below. For full details, refer to Chapter 3. In this dataset the ratio of positive class (i.e., links presence $y = 1$) to negative class (i.e., link absence $y = 0$) is 1.6%.

Between drug similarities for this dataset include *Chemical-based* using chemical structure of the drug molecules, *Ligand-based* using the closeness between protein-receptor families for each drug, *Expression-based* comparing the gene expression levels in response to the administration of each drug, *Side-effect-based* via comparing the reported side-effects for each drug, and *Annotation-based* via comparing the ontological characterizations of drugs.

Between target similarities for this dataset include *Sequence-based* via measuring the goodness of alignment between the genetic codes of each target, *Protein-protein interaction network-based* using the graph distance between proteins encoded by each target gene on protein-protein interaction network, *Gene Ontology-based* via comparing the semantic similarity between genes based on their ontological classification.

4.7.2 Results

We use the baseline of selecting k relations or similarities for all instances that we used in Chapter 3 to achieve state-of-the-art performance, and increasing the k at each step. It is important to note that this baseline does not have a nomination quota and basically nominates all instance to receive more relations at each step. Our nomination method in contrast is limited by a quota and can not explore the space as freely as the selected baseline.

Figure 4.4a depicts the performance of the AWL nomination method with quota of 10% in comparison with the baseline on the drug-target interaction dataset.

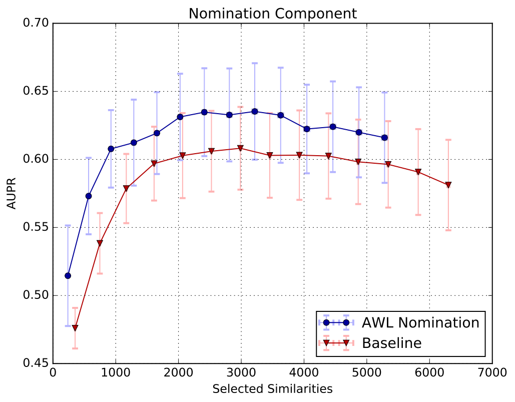
In this setting, limitations imposed on the search space by AWL nomination method improves the link prediction performance. It is also notable that number of selected similarities at each step by using the nomination method is less than the baseline.

Figure 4.4b shows the performance of the activation method in comparison to the baseline method. In this setting although all instance were nominated to get more relations, the relations with higher activation score were prioritized to be included in the neighborhood graph, the performance achieved via this method is even higher than the nomination method. It is also notable that the number of similarities selected in this experiment is less the number of experiments in Figure 4.4a, which can suggest limiting the search space based on the relation can be more restrictive.

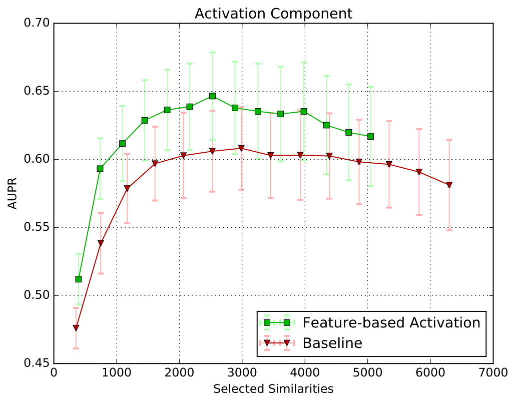
The nomination method limits the search space by prioritizing the instance to get more relations, while the activation method limit the search space by prioritizing which relations to be selected for the neighborhood graph. Figure 4.4c shows using combination of nomination and activation methods could be overly restrictive, This finding is consistent with exploration and exploitation balance principle [104], and suggests some degree of random search in the relation space may be helpful.

4.8 Related Work

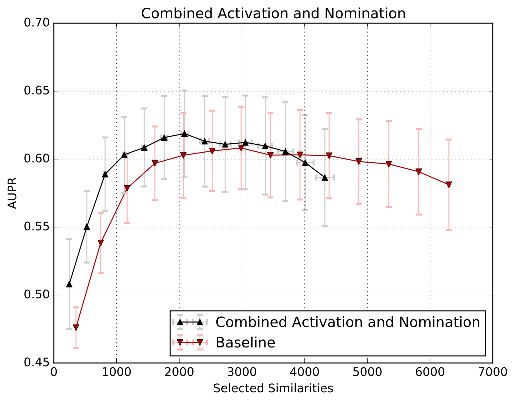
As mentioned earlier, the general problem of constructing a neighborhood graph from a candidate graph of relation is related to multiple areas of interest in the research community. We briefly discuss these perspectives in this section. To



(a) Nomination Component



(b) Activation Component



(c) Combined Nomination and Activation

Figure 4.4: Drug-Target interaction prediction experiments. Horizontal axis show mean and standard deviation (std) of the number of similarities included in the neighborhood graph at each iteration and vertical axis shows the mean and std of the AUPR at each step over ten folds.

improve scalability of the k-nearest neighbors (kNN) classifier, Ougiaroglou et al. [105] propose three early break heuristics for enhancing the neighborhood-graph-based kNN classifier by using an adaptive k-value based on the instance of interest. The first method is based on having a percentage of neighbors voting for one class, the second one is based on the difference in the number of votes for the winning class and others, and the final method is based on finding a set of consecutive neighbors that vote for a class. Their approach reduces the number of neighbors needed for classification, thus decreasing the computational cost associated with higher numbers of k. In our method, we dynamically prune the neighborhood graph by leveraging the state of the inference as well as the structural features of the similarities (i.e., their incidence scores).

Zeng et al. [106] propose a method based on matrix conversion and instance selection to deal with sparsity and scalability problems in collaborative filtering. Their idea for instance selection is that rarely rated items are not very informative and hence can be discarded. In our work, we select relations based on multiple structural properties of the candidate graph.

Olvera-López et al. [107] survey instance selection methods, and similar to feature selection approaches, generally categorize them into two main groups of *Filter* and *Wrapper*-based methods. According to their categorization most wrapper instance selection methods, perform based on *misclassification* information, and explore the neighborhood of those instances that are misclassified. Some methods are also based on sequential search and evolutionary algorithms. Many filter-based methods in their categorization use a variant of clustering. In contrast to wrapper methods, we nominate unknown instances based on methods capturing the uncertainty in inference instead of local misclassifications.

Brighton and Mellish [108] review instance selection methods and define two schemes of competence enhancement and preservation to categorize them. They argue that in competence enhancement, the aim is to improve the performance by removing noisy instances, and in competence preservation the aim to maintain the level of performance and eliminate the instances with redundant information. In our work, we can trade-off these perspectives with our dynamic nomination and activation.

Yu et al. [109] propose an information-theoretic approach to feature and instance selection for collaborative filtering. To find relevant instances, they define and use a notion of rationality for instances based on uncertainty reduction. Our approach can have different nomination features including the uncertainty-based nomination and also structural activation features to iteratively searches the space of neighborhood graphs.

Baltrunas and Ricci [110] propose a locally adaptive method called *best item per overlap* to compute similarities between users for collaborative filtering considering the target user and item. They relate their work to feature selection. They first assign weights to items and select the items with highest weights to compute user similarity for predict ratings for each target item. Their method results in a smaller set of items being selected for the user–user similarity computation and improved performance. However computing a distinct similarity for each user–item pair in the test set is computationally very expensive. In contrast, our nomination step reduces the number of unknown instances that require activation of relations, focusing the search on relevant parts of the space.

Liang et al. [111] define two item sets of *common items* and *similar preference* for two users and argue that not all items rated by both users are indicative of their similar preference. Their method is based on the recursive assumption that if an item is indicative of similar preference its neighbors should also be in the *similar preference* set, and operates by removing each item from the *common set* and measuring its effect on the computed similarity. They show improvements in performance, but their method is computationally expensive.

For link prediction multiple methods have been proposed in different domains [44, 61] such as unsupervised topological methods [112], factorization models [113], latent membership models [49], and neighborhood-based methods [114]. Graph-based semi-supervised methods and collective models extend neighborhood-based methods by taking advantage of the structure between all instances with known and unknown labels and jointly predicting links for all unknown instances. Link propagation method by Kashima et al. [103] which extends label propagation to infer information about links is an example in this category.

Our work published in [52, 97, 99] uses a Markov random field to achieve state-of-the-art performance in drug-target interaction prediction, drug-drug interaction prediction, and hybrid recommender systems for rating predictions with a graph-based collective model. We construct a probabilistic graphical model (e.g., Markov random field) based on observed relations (e.g., similarities) between all known and unknown instances (i.e., links), and find the values of unknown labels that maximize the probability a an exponential family distribution given the observed labels of instances and relations between unknown and known labels, as well as relations between all unknown labels.

4.9 Discussion and Conclusion

In this chapter, we highlight the limitations of sequential neighborhood graph construction and introduce a framework to dynamically construct multi-relational neighborhood graphs during inference. We base our dynamic neighborhood graph construction on the states of variables from intermediate inference results, the structural properties of the relations connecting them, and weight parameters learned by the model. We then lay out the formulation of the general link prediction task as inference on neighborhood graphs, and present our results on drug-target interaction networks showing effectiveness of our methods.

Chapter 5: Discussion and Conclusion

I proposed a set of template models to perform collective predictions on different multi-relational networks and showed the importance of both collective predictions and consideration of multi-relational characteristics in predictive modeling of networks. I focused on two main tasks in predictive models of real-world networks; inferring information about nodes, and inferring information about links.

First, I proposed three methods for node classification in a multi-relational network motivated by a real-world social network with several types of time-stamped edges between users. I used the multi-relational structure of the network, sequential properties of the network formation, and collective propagation of reputation for social spammer detection. Using these approaches I show the effectiveness of leveraging the multi-relational nature of the network and collective classification of the nodes.

I then showed that data from many domains can be modeled via a common bipartite network structure and presented a probabilistic template model for predicting information about the links in this common augmented bipartite network structure to support multi-relational characteristics of the network and make collective predictions. I showed the effectiveness of the template model in pharmaceutical and recommender systems domains and outperformed the state-of-the-art methods in both domains via my general link inference framework.

Finally, I showed that the collective link prediction task is an instance of a general graph-based prediction model that relies on a neighborhood graph for predictions and highlighted the challenges of pre-selecting a neighborhood graph. I proposed a framework that can dynamically adapt the neighborhood graph based on the state of variables from intermediate inference results, as well as structural properties of the relations connecting them to improve the performance of the model.

5.1 Future Directions

Building on the findings of this dissertation that shows the importance of modeling the multi-relational nature of the network as well as performing collective predictions, an interesting area of consideration would be to study the connections between models that could implicitly or explicitly leverage joint inference and the multi-relational characteristics of the data.

Joint inference relies on the basic assumption that the instances that are related to each other influence each others' properties. A range of models use this assumption in different ways; k-nearest neighbors classifiers that simply consider the relations between the labeled instance and unlabeled instances are the most basic method in this category. More sophisticated approaches such as the graph-based semi-supervised learning, also include the relations between unlabeled instances to propagate the information. Statistical relational learning methods extend these capabilities in various ways that include the ability to model several types of relations, heterogeneous types of information, and more complex relations.

Often the time complexity of the models in each category increases with their ability to model more complex relations and dependencies. By studying the connection between methods in related disciplines that could support multi-relational nature of the network and collective predictions, one may find models that could benefit from different aspects of each discipline, and are both scalable and capable of representing common forms of multi-relational dependencies.

Appendix 6: Hinge-loss Markov Random Fields

Hinge-loss Markov random fields (HL-MRFs) are a general class of conditional, continuous probabilistic models [38]. HL-MRFs are log-linear models whose features are hinge-loss functions of the variable states. Through constructions based on *soft logic*, hinge-loss potentials can be used to model generalizations of logical conjunction and implication. A hinge-loss Markov random field P over random variables $\mathbf{Y}$ and conditioned on random variables $\mathbf{X}$ defines a conditional probability density function as the following:

$$P\left(\mathbf{Y}|\mathbf{X}\right) = \frac{1}{Z(\lambda)} \exp\left[-\sum_{j=1}^{m} \lambda_j \phi_j(\mathbf{Y}, \mathbf{X})\right] \ ,$$

where Z is the normalization constant of the form

$$Z = \int_{\mathbf{Y}} \exp\left[-\sum_{j=1}^{m} \lambda_j \phi_j(\mathbf{Y}, \mathbf{X})\right] \ .$$

In the above, ϕ is a set of m continuous potential of the form

$$\phi_j(\mathbf{Y}, \mathbf{X}) = \left[\max\left\{\ell_j(\mathbf{Y}, \mathbf{X}), 0\right\}\right]^{p_j} \ ,$$

where ℓ is a linear function of $\mathbf{Y}$, and $\mathbf{X}$ and $p_j \in \{1, 2\}$.

6.1 Probabilistic Soft Logic

Probabilistic Soft Logic (PSL) [38] uses a first-order logical syntax as a templating language for HL-MRFs. A typical example of a PSL rule is

$$\lambda : P(a, b) \wedge Q(a) \rightarrow R(b),$$

where P, Q, and R are *predicates*, a and b are *variables*, and λ is the weight associated with the rule, indicating its importance. For instance, $P(a, b)$ can represent a relational edge in the graph such as $\textsc{Reported}(a, b)$, and $Q(a)$ could represent a value for a vertex such as $\textsc{Credible}(b)$. Each grounding forms a ground atom, or logical fact, that has a soft-truth value in the range $[0, 1]$. The rules can encode domain knowledge about dependencies between these predicates. PSL uses the *Lukasiewicz* t-norm and co-norm to provide relaxations of the logical connectives AND ($\wedge$), OR ($\vee$), and NOT ($\neg$) as following:

$$\begin{aligned} p \;\wedge\; q &= \max(0,\; p + q - 1), \\ p \;\vee\; q &= \min(1,\; p + q), \\ \neg\, p &= 1 - p. \end{aligned}$$

A ground instance of a rule r ($r_{\text{body}} \longrightarrow r_{\text{head}}$) is satisfied when the value of r_{body} is not greater than the value of r_{head}. ℓ is defined to capture the distance to satisfaction for rules:

$$\ell = \max\{0, val(r_{\text{body}}) - val(r_{\text{head}})\}\ .$$

A full assignment of soft-truth values to a set of ground atoms is also called an *interpretation* (I) of that set. Using the above relaxations and the logical identity $p \to q \equiv \neg\, p \vee q$, a ground instance of a rule $r_{\text{body}} \longrightarrow r_{\text{head}}$ is satisfied (i.e., $I(r) = 1$) when $I(r_{\text{body}}) \leq I(r_{\text{head}})$.

6.2 Inference

In HL-MRFs, inference of a *maximum a posteriori probability* (MAP), which finds the most probable interpretation given evidence (i.e., a given partial interpretation) is computed by maximizing the density function $P\left(\mathbf{Y}|\mathbf{X}\right)$, subject to both the evidence and the equality and inequality constraints. For example, given a drug-target interaction network and interactions between some drugs and some targets, the goal of MAP inference is to output the most likely interactions between all other drugs and targets. Finding the most probable interpretation given a set of weighted rules reduces to solving a convex optimization problem and can be solved very efficiently [38].

6.3 Weight Learning

The PSL rule weights indicate how much an assignment is penalized if a rule is not satisfied. They are measures of importance for each rule. We can set the weights based on prior domain knowledge or, if we have training data, we can learn the weights using a number of different training objective and learning algorithms [38]. In particular, the primary methods for weight learning are voted-perceptron approximate maximum likelihood, maximum pseudo-likelihood, and large-margin estimation.

We mainly used approximate maximum likelihood in our models. We seek to maximize the log-likelihood of the full data, including both the observed data and the training labels. We can do so using gradient ascent, where the gradient of the log-likelihood with respect to a weight λ_j is:

$$\frac{\partial}{\partial \lambda_j} \log P\left(\mathbf{Y}|\mathbf{X}\right) = \mathbb{E}_{\lambda}\left[\phi_j(\mathbf{Y},\mathbf{X})\right] - \phi_j(\mathbf{Y},\mathbf{X}),$$

This gradient is intractable to compute exactly because the expectation term enumerates all possible interpretations, so we approximate the expectation by the values at the MAP solution.

Bibliography

[1] Lise Getoor and Christopher P. Diehl. Link mining: a survey. *SIGKDD Explor. Newsl.*, 7(2):3–12, December 2005.

[2] Ben London and Lise Getoor. Collective classification of network data. In Charu C. Aggarwal, editor, *Data Classification: Algorithms and Applications*. CRC Press, 2013. May differ from the published version.

[3] Wikipedia. History of email spam — Wikipedia, the free encyclopedia, 2014. URL http://en.wikipedia.org/wiki/History_of_email_spam.

[4] Harold Nguyen. 2013 state of social media spam. Technical report, Nexgate. URL http://go.nexgate.com/nexgate-social-media-spam-research-report.

[5] Enrico Blanzieri and Anton Bryl. A survey of learning-based techniques of email spam filtering. *Artificial Intelligence Review*, 2008.

[6] Nikita Spirin and Jiawei Han. Survey on web spam detection: principles and algorithms. *ACM SIGKDD Explorations Newsletter*, 13(2):50–64, 2012.

[7] Nisheeth Shrivastava, Anirban Majumder, and Rajeev Rastogi. Mining (social) network graphs to detect random link attacks. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International conference on*, pages 486–495. IEEE, 2008.

[8] Chi-Yao Tseng and Ming-Syan Chen. Incremental SVM model for spam detection on dynamic email social networks. In *Computational Science and Engineering, 2009. CSE'09. International conference on*, volume 4, pages 128–135. IEEE, 2009.

[9] P Oscar and VP Roychowdbury. Leveraging social networks to fight spam. *IEEE Computer*, 38(4):61–68, 2005.

[10] Luca Becchetti, Carlos Castillo, Debora Donato, Ricardo Baeza-Yates, and Stefano Leonardi. Link analysis for web spam detection. *ACM Transactions on the Web (TWEB)*, 2008.

[11] Carlos Castillo, Debora Donato, Aristides Gionis, Vanessa Murdock, and Fabrizio Silvestri. Know your neighbors: Web spam detection using the web topology. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2007.

[12] Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. Combating web spam with trustrank. In *Proceedings of the thirtieth international conference on very large data bases*, pages 576–587. VLDB Endowment, 2004.

[13] Paul-Alexandru Chirita, Jörg Diederich, and Wolfgang Nejdl. Mailrank: using ranking for spam detection. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 373–380. ACM, 2005.

[14] Jacob Abernethy, Olivier Chapelle, and Carlos Castillo. Graph regularization methods for web spam detection. *Machine Learning*, 81(2):207–225, 2010.

[15] Paul Heymann, Georgia Koutrika, and Hector Garcia-Molina. Fighting spam on social web sites: A survey of approaches and future challenges. *Internet Computing, IEEE*, 11(6):36–45, 2007.

[16] Xia Hu, Jiliang Tang, and Huan Liu. Leveraging knowledge across media for spammer detection in microblogging. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 2014.

[17] Enhua Tan, Lei Guo, Songqing Chen, Xiaodong Zhang, and Yihong Zhao. Unik: unsupervised social network spam detection. In *Proceedings of the 22nd ACM international conference on information & knowledge management*. ACM, 2013.

[18] Tao Stein, Erdong Chen, and Karan Mangla. Facebook immune system. In *Proceedings of the 4th Workshop on Social Network Systems*. ACM, 2011.

[19] Hongyu Gao, Jun Hu, Christo Wilson, Zhichun Li, Yan Chen, and Ben Y Zhao. Detecting and characterizing social spam campaigns. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 35–47. ACM, 2010.

[20] Benjamin Markines, Ciro Cattuto, and Filippo Menczer. Social spam detection. In *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pages 41–48. ACM, 2009.

[21] Fabricio Benevenuto, Gabriel Magno, Tiago Rodrigues, and Virgilio Almeida. Detecting spammers on twitter. In *Collaboration, electronic messaging, anti-abuse and spam conference (CEAS)*, volume 6, page 12, 2010.

[22] Kyumin Lee, James Caverlee, and Steve Webb. Uncovering social spammers: social honeypots+ machine learning. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 435–442. ACM, 2010.

[23] Yin Zhu, Xiao Wang, Erheng Zhong, Nathan N Liu, He Li, and Qiang Yang. Discovering spammers in social networks. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[24] Gueorgi Kossinets and Duncan J Watts. Empirical analysis of an evolving social network. *Science*, 2006.

[25] Xin Jin, Cindy Xide Lin, Jiebo Luo, and Jiawei Han. Socialspamguard: A data mining-based spam detection system for social media networks. *PVLDB*, 2011.

[26] Xianchao Zhang, Shaoping Zhu, and Wenxin Liang. Detecting spam and promoting campaigns in the twitter social network. In *ICDM*, pages 1194–1199, 2012.

[27] Carlos Laorden, Borja Sanz, Igor Santos, Patxi Galán-García, and Pablo G Bringas. Collective classification for spam filtering. *Logic Journal of IGPL*, 2012.

[28] Guang-Gang Geng, Qiudan Li, and Xinchang Zhang. Link based small sample learning for web spam detection. In *Proceedings of the 18th international conference on World wide web*, pages 1185–1186. ACM, 2009.

[29] Mohamadali Torkamani and Daniel Lowd. Convex adversarial collective classification. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 642–650, 2013.

[30] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[31] J Ignacio Alvarez-Hamelin, Luca Dall'Asta, Alain Barrat, and Alessandro Vespignani. Large scale networks fingerprinting and visualization using the k-core decomposition. In *Advances in neural information processing systems (NIPS)*, 2005.

[32] Tommy R Jensen and Bjarne Toft. *Graph coloring problems.* John Wiley & Sons, 2011.

[33] S Pemmaraju and S Skiena. Implementing discrete mathematics: Combinatorics and graph theory with mathematica, 2003.

[34] Thomas Schank. Algorithmic aspects of triangle-based network analysis. *Phd in computer science, University Karlsruhe*, 2007.

[35] Zhengzheng Xing, Jian Pei, and Eamonn Keogh. A brief survey on sequence classification. *ACM SIGKDD Explorations Newsletter*, 2010.

[36] Fuchun Peng, Dale Schuurmans, and Shaojun Wang. Augmenting naive Bayes classifiers with statistical language models. *Information Retrieval*, 2004.

[37] Fei Zheng and Geoffrey I Webb. Tree augmented naive Bayes. In *Encyclopedia of Machine Learning*, pages 990–991. Springer, 2010.

[38] Stephen H Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. Hinge-loss markov random fields and probabilistic soft logic. *arXiv preprint arXiv:1505.04406*, 2015.

[39] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[40] Nesreen K. Ahmed, Jennifer Neville, and Ramana Kompella. Network sampling: From static to streaming graphs. *ACM Trans. Knowl. Discov. Data*, 2013.

[41] Jure Leskovec and Christos Faloutsos. Sampling from large graphs. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006.

[42] Mohammad Al Hasan and Mohammed J. Zaki. Output space sampling for graph patterns. *PVLDB*, 2009.

[43] Shobeir Fakhraei, Hamid Soltanian-Zadeh, and Farshad Fotouhi. Bias and stability of single variable classifiers for feature ranking and selection. *Expert Systems with Applications*, 41(15):6945 – 6958, 2014.

[44] Mohammad Al Hasan and Mohammed J Zaki. A survey of link prediction in social networks. In *Social network data analytics*, pages 243–275. Springer, 2011.

[45] Ajit P Singh and Geoffrey J Gordon. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 650–658. ACM, 2008.

[46] Ryan Lichtnwalter and Nitesh V Chawla. Link prediction: fair and effective evaluation. In *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on*, pages 376–383. IEEE, 2012.

[47] Yizhou Sun and Jiawei Han. Mining heterogeneous information networks: principles and methodologies. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 3(2):1–159, 2012.

[48] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8), 2009.

[49] Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9(Sep):1981–2014, 2008.

[50] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King. Recommender systems with social regularization. In *WSDM*, 2011.

[51] Shobeir Fakhraei, Louiqa Raschid, and Lise Getoor. Drug-target interaction prediction for drug repurposing with probabilistic similarity logic. In *ACM SIGKDD 12th International Workshop on Data Mining in Bioinformatics (BIOKDD)*. ACM, 2013.

[52] Pigi Kouki, Shobeir Fakhraei, James Foulds, Magdalini Eirinaki, and Lise Getoor. Hyper: A flexible and extensible probabilistic framework for hybrid recommender systems. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 99–106. ACM, 2015.

[53] David J. Adams. The valley of death in anticancer drug development a reassessment. *Trends in Pharmacological Sciences*, 33(4):173 – 180, 2012.

[54] Aislyn D.W. Boran and Ravi Iyengar. Systems approaches to polypharmacology and drug discovery. *Current opinion in drug discovery & development*, 13 (3):297, 2010.

[55] S. J. Cockell, J. Weile, P. Lord, C. Wipat, D. Andriychenko, M. Pocock, D. Wilkinson, M. Young, and A. Wipat. An integrated dataset for in silico drug discovery. *J Integr Bioinform*, 7(3):116, 2010.

[56] Michael J. Keiser, Vincent Setola, John J. Irwin, Christian Laggner, Atheir I. Abbas, Sandra J. Hufeisen, Niels H. Jensen, Michael B. Kuijer, Roberto C. Matos, Thuy B. Tran, Ryan Whaley, Richard A. Glennon, Jérôme Hert, Kelan L. H. Thomas, Douglas D. Edwards, Brian K. Shoichet, and Bryan L. Roth. Predicting new molecular targets for known drugs. *Nature*, 462(7270):175–181, November 2009.

[57] Hao Ding, Ichigaku Takigawa, Hiroshi Mamitsuka, and Shanfeng Zhu. Similarity-based machine learning methods for predicting drug–target interactions: a brief review. *Briefings in bioinformatics*, page bbt056, 2013.

[58] Muhammed A Yildirim, Kwang-Il Goh, Michael E Cusick, Albert-Laszlo Barabasi, and Marc Vidal. Drug–target network. *Nature biotechnology*, 25 (10):1119–1126, October 2007. PMID: 17921997.

[59] Soyoung Lee, Keunwan Park, and Dongsup Kim. Building a drug–target network and its applications. *Expert Opinion on Drug Discovery*, 4(11):1177–1189, November 2009.

[60] Liat Perlman, Assaf Gottlieb, Nir Atias, Eytan Ruppin, and Roded Sharan. Combining drug and gene similarity measures for drug-target elucidation. *Journal of Computational Biology*, 18(2):133–145, February 2011.

[61] Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, 2011.

[62] Yoshihiro Yamanishi, Michihiro Araki, Alex Gutteridge, Wataru Honda, and Minoru Kanehisa. Prediction of drug–target interaction networks from the integration of chemical and genomic spaces. *Bioinformatics*, 24(13):i232–i240, July 2008.

[63] Matthias Broecheler, Lilyana Mihalkova, and Lise Getoor. Probabilistic similarity logic. In *Conference on Uncertainty in Artificial Intelligence*, 2010.

[64] Justin Lamb. The connectivity map: a new tool for biomedical research. *Nature Reviews Cancer*, 7(1):54–60, January 2007.

[65] Justin Lamb, Emily D. Crawford, David Peck, Joshua W. Modell, Irene C. Blat, Matthew J. Wrobel, Jim Lerner, Jean-Philippe Brunet, Aravind Subramanian, Kenneth N. Ross, Michael Reich, Haley Hieronymus, Guo Wei, Scott A. Armstrong, Stephen J. Haggarty, Paul A. Clemons, Ru Wei, Steven A. Carr, Eric S. Lander, and Todd R. Golub. The connectivity map: Using gene-expression signatures to connect small molecules, genes, and disease. *Science*, 313(5795):1929–1935, September 2006.

[66] Rui Chang, Robert Shoemaker, and Wei Wang. A novel knowledge-driven systems biology approach for phenotype prediction upon genetic intervention. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 8(5):1170–1182, 2011.

[67] Rui Chang, Robert Shoemaker, and Wei Wang. Systematic search for recipes to generate induced pluripotent stem cells. *PLoS computational biology*, 7(12): e1002300, 2011.

[68] Feixiong Cheng, Chuang Liu, Jing Jiang, Weiqiang Lu, Weihua Li, Guixia Liu, Weixing Zhou, Jin Huang, and Yun Tang. Prediction of drug-target interactions and drug repositioning via network-based inference. *PLoS Comput Biol*, 8(5):e1002503, May 2012.

[69] Masahiro Hattori, Yasushi Okuno, Susumu Goto, and Minoru Kanehisa. Heuristics for chemical compound matching. *GENOME INFORMATICS SERIES*, pages 144–153, 2003.

[70] Salvatore Alaimo, Alfredo Pulvirenti, Rosalba Giugno, and Alfredo Ferro. Drug-target interaction prediction through domain-tuned network based inference. *Bioinformatics*, 2013.

[71] Kevin Bleakley and Yoshihiro Yamanishi. Supervised prediction of drug–target interactions using bipartite local models. *Bioinformatics*, 25(18):2397–2403, September 2009.

[72] Jian-Ping Mei, Chee-Keong Kwoh, Peng Yang, Xiao-Li Li, and Jie Zheng. Drug–target interaction prediction by learning from local information and neighbors. *Bioinformatics*, 29(2):238–245, 2013.

[73] Yuhao Wang and Jianyang Zeng. Predicting drug-target interactions using restricted boltzmann machines. *Bioinformatics*, 29(13):i126–i134, 2013.

[74] Bin Chen, Ying Ding, and David J. Wild. Assessing drug target association using semantic linked data. *PLoS Comput Biol*, 8(7):e1002574, July 2012.

[75] Assaf Gottlieb, Gideon Y. Stein, Eytan Ruppin, and Roded Sharan. PREDICT: a method for inferring novel drug indications with application to personalized medicine. *Molecular Systems Biology*, 7(1), June 2011.

[76] Alvaro E. Monge and Charles Elkan. The field matching problem: Algorithms and applications. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 267–270, 1996.

[77] Hanna Köpcke, Andreas Thor, and Erhard Rahm. Evaluation of entity resolution approaches on real-world match problems. *Proc. VLDB Endow.*, 3(1-2): 484–493, September 2010.

[78] Steven Euijong Whang, David Menestrina, Georgia Koutrika, Martin Theobald, and Hector Garcia-Molina. Entity resolution with iterative blocking. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, SIGMOD '09, pages 219–232. ACM, 2009.

[79] Prithviraj Sen, Galileo Mark Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

[80] David S. Wishart, Craig Knox, An Chi Guo, Dean Cheng, Savita Shrivastava, Dan Tzur, Bijaya Gautam, and Murtaza Hassanali. Drugbank: a knowledgebase for drugs, drug actions and drug targets. *Nucleic acids research*, 36(suppl 1):D901–D906, 2008.

[81] Minoru Kanehisa, Susumu Goto, Miho Furumichi, Mao Tanabe, and Mika Hirakawa. Kegg for representation and analysis of molecular networks involving diseases and drugs. *Nucleic acids research*, 38(suppl 1):D355–D360, 2010.

[82] Yanbin Liu, Bin Hu, Chengxin Fu, and Xin Chen. Dcdb: drug combination database. *Bioinformatics*, 26(4):587–588, 2010.

[83] Stefan Günther, Michael Kuhn, Mathias Dunkel, Monica Campillos, Christian Senger, Evangelia Petsalaki, Jessica Ahmed, Eduardo Garcia Urdiales, Andreas Gewiess, Lars Juhl Jensen, et al. Supertarget and matador: resources for exploring drug-target relationships. *Nucleic acids research*, 36(suppl 1): D919–D922, 2008.

[84] Derek Hansen, Ben Shneiderman, and Marc A Smith. *Analyzing social media networks with NodeXL: Insights from a connected world*. Morgan Kaufmann, 2010.

[85] Christoph Steinbeck, Christian Hoppe, Stefan Kuhn, Matteo Floris, Rajarshi Guha, and Egon L Willighagen. Recent developments of the chemistry development kit (CDK)—an open-source java library for chemo-and bioinformatics. *Current pharmaceutical design*, 12(17):2111–2120, 2006.

[86] Michael Kuhn, Monica Campillos, Ivica Letunic, Lars Juhl Jensen, and Peer Bork. A side effect resource to capture phenotypic effects of drugs. *Molecular systems biology*, 6(1), 2010.

[87] A. Skrbo, B. Begović, and S. Skrbo. Classification of drugs using the ATC system (anatomic, therapeutic, chemical classification) and the latest changes. *Medicinski arhiv*, 58(1 Suppl 2):138, 2004.

[88] Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999.

[89] Eric Jain, Amos Bairoch, Severine Duvaud, Isabelle Phan, Nicole Redaschi, Baris E. Suzek, Maria J. Martin, Peter McGarvey, and Elisabeth Gasteiger. Infrastructure for the life sciences: design and implementation of the UniProt website. *BMC bioinformatics*, 10(1):136, 2009.

[90] Tom Fawcett. ROC graphs: Notes and practical considerations for researchers. *Machine Learning*, 31:1–38, 2004.

[91] Jesse Davis and Mark Goadrich. The relationship between precision-recall and ROC curves. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006. ISBN 1-59593-383-2.

[92] P. D. Hoff, A. E. Raftery, and M. S. Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97, 2001.

[93] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *ICML*, 2008.

[94] C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender Systems Handbook*. Springer, 2011.

[95] J. Liu, C. Wu, and W. Liu. Bayesian probabilistic matrix factorization with social relations and item contents for recommendation. *Decision Support Systems*, 55(3), 2013.

[96] Tony Jebara, Jun Wang, and Shih-Fu Chang. Graph construction and b-matching for semi-supervised learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009.

[97] S. Fakhraei, B. Huang, L. Raschid, and L. Getoor. Network-based drug-target interaction prediction with probabilistic soft logic. *Transactions on Computational Biology and Bioinformatics*, 11(5), 2014.

[98] Xiaojin Zhu, Andrew B Goldberg, and Tushar Khot. Some new directions in graph-based semi-supervised learning. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 1504–1507. IEEE, 2009.

[99] Dhanya Sridhar, Shobeir Fakhraei, and Lise Getoor. A probabilistic approach for collective similarity-based drug-drug interaction prediction. *Bioinformatics*, 2016.

[100] Shobeir Fakhraei, Bert Huang, and Lise Getoor. Collective inference and multi-relational learning for drug-target interaction prediction. In *NIPS Workshop on Machine Learning in Computational Biology (MLCB)*, 2013.

[101] Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[102] Jay Pujara, Ben London, and Lise Getoor. Budgeted online collective inference. In *Uncertainty in Artificial Intelligence*, 2015.

[103] Hisashi Kashima, Tsuyoshi Kato, Yoshihiro Yamanishi, Masashi Sugiyama, and Koji Tsuda. Link propagation: A fast semi-supervised learning algorithm for link prediction. In *SDM*, volume 9, pages 1099–1110. SIAM, 2009.

[104] James G March. Exploration and exploitation in organizational learning. *Organization science*, 2(1):71–87, 1991.

[105] Stefanos Ougiaroglou, Alexandros Nanopoulos, Apostolos N. Papadopoulos, Yannis Manolopoulos, and Tatjana Welzer-Druzovec. Adaptive k-nearest-neighbor classification using a dynamic number of nearest neighbors. In *Proceedings of the 11th East European Conference on Advances in Databases and Information Systems*, ADBIS'07, pages 66–82. Springer-Verlag, 2007.

[106] Chun Zeng, Chun-Xiao Xing, and Li-Zhu Zhou. Similarity measure and instance selection for collaborative filtering. In *Proceedings of the 12th International Conference on World Wide Web*, 2003.

[107] J Arturo Olvera-López, J Ariel Carrasco-Ochoa, J Francisco Martínez-Trinidad, and Josef Kittler. A review of instance selection methods. *Artificial Intelligence Review*, 34(2):133–143, 2010.

[108] Henry Brighton and Chris Mellish. Advances in instance selection for instance-based learning algorithms. *Data mining and knowledge discovery*, 6(2):153–172, 2002.

[109] Kai Yu, Xiaowei Xu, Martin Ester, and Hans-Peter Kriegel. Feature weighting and instance selection for collaborative filtering: An information-theoretic approach*. *Knowledge and Information Systems*, 5(2):201–224, 2003.

[110] Linas Baltrunas and Francesco Ricci. Locally Adaptive Neighborhood Selection for Collaborative Filtering Recommendations. In Wolfgang Nejdl, Judy Kay, Pearl Pu, and Eelco Herder, editors, *Adaptive Hypermedia and Adaptive Web-Based Systems*, volume 5149, pages 22–31. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[111] Zhang Liang, Xiao Bo, and Guo Jun. An approach of selecting right neighbors for collaborative filtering. In *Innovative Computing, Information and Control (ICICIC), 2009 Fourth International Conference on*, pages 1057–1060. IEEE, 2009.

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

[113] Aditya Krishna Menon and Charles Elkan. Link prediction via matrix factorization. In *Machine Learning and Knowledge Discovery in Databases*, pages 437–452. Springer, 2011.

[114] Charu C Aggarwal. Neighborhood-based collaborative filtering. In *Recommender Systems*, pages 29–70. Springer, 2016.