Link Prediction via Community Detection in Bipartite Multi-Layer Graphs

Maksim Koptelov, Albrecht Zimmermann, Bruno Crémilleux, Lina F. Soualmia

Link Prediction via Community Detection in Bipartite Multi-Layer Graphs

Maksim Koptelov
Normandie Univ, UNICAEN, ENSICAEN, CNRS - UMR GREYC, 14000 Caen, France
maksim.koptelov@unicaen.fr

Albrecht Zimmermann
Normandie Univ, UNICAEN, ENSICAEN, CNRS - UMR GREYC, 14000 Caen, France
albrecht.zimmermann@unicaen.fr

Bruno Crémilleux
Normandie Univ, UNICAEN, ENSICAEN, CNRS - UMR GREYC, 14000 Caen, France
bruno.cremilleux@unicaen.fr

Lina Soualmia
Normandie Univ, UNIROUEN, ULH, INSAR - LITIS-TIBS, 76800 Rouen, France
lina.soualmia@chu-rouen.fr

ABSTRACT

The growing number of multi-relational networks pose new challenges concerning the development of methods for solving classical graph problems in a multi-layer framework, such as link prediction. In this work, we combine an existing bipartite local models method with approaches for link prediction from communities to address the link prediction problem in multi-layer graphs. To this end, we extend existing community detection-based link prediction measures to the bipartite multi-layer network setting. We obtain a new generic framework for link prediction in bipartite multi-layer graphs, which can integrate any community detection approach, is capable of handling an arbitrary number of networks, rather inexpensive (depending on the community detection technique), and able to automatically tune its parameters. We test our framework using two of the most common community detection methods, the Louvain algorithm and spectral partitioning, which can be easily applied to bipartite multi-layer graphs. We evaluate our approach on benchmark data sets for solving a common drug-target interaction prediction task in computational drug design and demonstrate experimentally that our approach is competitive with the state-of-the-art.

SAC '20, March 30-April 3, 2020, Brno, Czech Republic

ACM ISBN 978-1-4503-6866-7/20/03…$15.00
https://doi.org/10.1145/3341105.3373874

CCS CONCEPTS

• **Mathematics of computing → Graph algorithms**; • **Information systems → Data mining**; *Recommender systems*; Decision support systems; • **Applied computing →** Bioinformatics;

KEYWORDS

multi-layer graphs, bipartite graphs, graph mining, link prediction, community detection

ACM Reference Format:
Maksim Koptelov, Albrecht Zimmermann, Bruno Crémilleux, and Lina Soualmia. 2020. Link Prediction via Community Detection in Bipartite Multi-Layer Graphs. In *The 35th ACM/SIGAPP Symposium on Applied Computing (SAC '20), March 30-April 3, 2020, Brno, Czech Republic.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3341105.3373874

1 INTRODUCTION

Many real world applications can be modeled as bipartite graphs, vertices of which are divided into two distinct groups [31]. The problem setting that motivates our work is the prediction of links between drug candidates and biological targets, an essential step of computational drug design. But there are other link prediction settings that fall into the same category, for instance user-product recommendation.

The available data on drug-target interaction prediction are of heterogeneous structure, i.e. represented by networks the edges of which have different origins. This makes the straightforward use of most existing link prediction methods impossible. Current solutions are limited by the number or type of networks, often referred to as *layers*, e.g. three layers, with two assumed to be similarity networks [4, 6, 26].

To address these restrictions, we take inspiration from existing methods that use community detection to perform link prediction [17, 29, 33, 36]. While this decouples the problem into how to find communities in multi-layer graphs, and

how to exploit them for prediction, existing link prediction measures [5, 12, 34] are not directly applicable to the bipartite setting. To address this restriction, we extend several of those measures to our problem setting. In addition, we go a step further by proposing alternatives to those measures based on an existing bipartite local model. While we evaluate two concrete approaches for community detection, spectral partitioning and the Louvain algorithm, both of which can be easily applied to multi-layer graphs, we do not require any particular community detection approach. Our long-term contribution is the adaptation of existing link-prediction-by-community-detection measures to the bipartite multi-layer setting, which we evaluate experimentally, and the selection of the best measure and community detection approach combination. In addition, we demonstrate that the parameter settings of the community detection methods can be effectively set via internal cross-validation.

The rest of the paper is organized as follows. Section 2 discusses related work on link prediction and community detection in multi-layer networks. Section 3 provides basic notations and definitions. Section 4 explains how we adapt existing measures for our framework. Section 5 describes the data used for evaluation, the experimental setup and presents the results. Finally, Section 6 concludes and outlines future work.

2 RELATED WORK

Existing methods for link prediction in bipartite multi-layer networks for addressing the drug-target interaction problem can be grouped into three classes: similarity, random-walker, and latent models based. The first group assumes 2 out of 3 possible layers to be similarity networks for drugs and targets respectively, and exploits similarity information to perform link prediction on the third bipartite layer [4, 11]. The second models the behavior of a random-walker to perform link prediction in multi-layer graph using PageRank adaptations [6, 7, 21]. Such methods are dependent on fixing the similarity networks and while the approach was extended to any number of networks in [21], it pays for this flexibility with high computational cost. The last group of methods maps drugs, targets and their interactions into a combined feature space, and performs drug-target interaction prediction using distance functions or regression analysis [35, 37]. The most recent family of methods in this mold is often referred to as graph embeddings [15]. The main disadvantage of this group of methods is a certain lack of interpretability.

The idea to use community information to predict links in graphs is not novel. Clauset *et al.* [8] proposed to exploit a learned hierarchical generative community model to estimate the probabilities of missing links in partially known networks. The authors of [17, 33, 36] combine community detection with existing edge prediction methods to improve prediction accuracy. These methods are based on the hypothesis that vertices in the same community have similar properties, and missing edges are more likely to be found within communities. Missing edges are predicted by node similarity using nearest-neighbor measures [36], Stochastic Block Models [17], or in-group/out-group neighbor similarity measures [33]. Edges can be predicted for vertices belonging to the same community even if there is no path between them within the community [19]. The density of links in a particular community or between two communities can be exploited in a naïve Bayes model [27]. The authors of [1] reimagine communities as groups of edges rather than vertices, and [29] use community detection to modify similarity measures. Finally, there is a set of methods which extend the concept of shared neighborhoods [25] to community neighborhoods [5, 12, 34]. In addition, in [18] neighborhood measures have been extended to multiple layers.

Community detection in multi-graphs can also be performed in different ways: directly, by ensemble-based methods, or by graph flattening. The direct methods perform discovery of communities on the multi-layer network directly, e.g. by adapting objective functions for community detection to the multi-layer setting [10, 22, 32]. Ensemble-based methods perform community detection on each layer separately, and aggregate discovered communities afterwards [32]. Flattening approaches, finally, summarize multiple edges into single ones and use the resulting single-layer network to discover communities by using one of the common community detection approaches such as spectral partitioning [24] or Louvain algorithm [3]. In this work, we use the last type of approach due to their ease of use and potentially low computational complexity.

3 DEFINITIONS AND PROBLEM SETTING

3.1 Basic notations

A *graph* is a tuple $G =< V, E >$, where $V = \{v_1, v_2, ..., v_n\}$ denotes a set of vertices or nodes, and $E \subseteq V \times V$ a set of edges defined by distinct vertex pairs $(u, v) \in V \times V$ with $u \neq v$ (without self-loops). We also use the notion of a *bipartite graph*, which we define as a graph whose vertices can be divided into two classes V_1 and V_2 such that there is no edge between vertices of the same class: $G =< V_1 \cup V_2, E >$, $E \subset V_1 \times V_2$.

We address weighted and unweighted graphs in the same manner. We define a *weighted graph* as one with a labeling function for edges $E \mapsto A_e$ with $A_e \in [0, 1]$, where 0 denotes that there is no interaction between vertices, 1 confirmed interaction, and an intermediate value represents an interaction probability. An *unweighted graph* is one where every edge is labeled by 1.

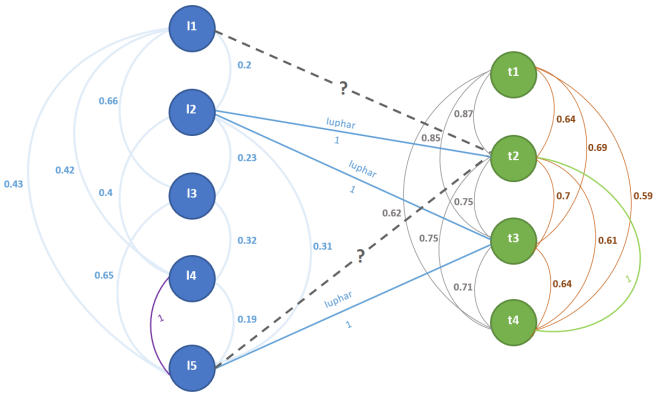

Figure 1: Example of a bipartite multi-layer graph with 6 layers for drug-target prediction problem (blue nodes represent drugs or ligands, green nodes represent targets): drug-target network is in deep blue (bipartite layer based on IUPHAR network), drug networks are in light blue and violet, target networks are in green, grey and brown. New possible interactions are represented by dashed edges with question marks.

To exploit different sources of information in one single structure, we employ multi-layer networks. We define a *multi-layer network* as a weighted graph where more than one edge (u, v) can exist for a pair of vertices u, v. Multi-layer networks can be decomposed into disjunct set of graphs G_l that contain at most a single edge for each pair of vertices, called *layers* or just *networks*. As we wrote above, our original setting is a bipartite one. To combine it with the multi-layer framework, we define a *bipartite multi-layer graph* as a multi-layer network whose vertices can be divided into two classes, and where exactly one of the layers is a bipartite graph (see Fig. 1 for an example). Note, following Kivelä *et al.* [20], the multi-layer networks used in this work are **not** *node-aligned*[1], **not** *layer-disjoint*[2], have *diagonal couplings*[3] which are *categorical*[4], and the number of layers can be any.

We represent graphs as matrices. The *adjacency matrix* A has size $n \times n$, $n = |V|$, and A_{ij} represents the weight of the edge (v_i, v_j). In the case of multi-layer networks, we aggregate the weights of multiple edges between v_i and v_j by summing them up. Note that A has zeros on the main diagonal, because graphs as used in this work have no self-loops. The *degree matrix* D is the diagonal matrix

$$D = \begin{bmatrix} deg(v_1) & 0 & \dots \\ 0 & \ddots & 0 \\ 0 & & deg(v_n) \end{bmatrix}, \; deg(v_i) = \sum_{j=1}^{n} A_{ij}$$

of same size as A, where $deg(v_i)$ represents the degree of vertex v_i. The *degree* of a vertex is the sum of the weights of the edges adjacent to v_i [14].

The last, and arguably most important, matrix used in this paper is the Laplacian matrix. The *Laplacian matrix*, denoted by L, is a matrix of the same dimensionality as A and D, defined as the difference between the degree matrix and the adjacency matrix: $L = D - A$. L has the same values as D on the diagonal, and off the diagonal L_{ij} is equal to $-A_{ij}$.

3.2 Problem setting

We define the problem of link prediction in *bipartite multi-layer graph* addressed in this paper as follows.

For a given bipartite multi-layer graph $G = G_{V_1V_2} \cup G^{(n)}_{V_1} \cup G^{(m)}_{V_2}$ with $1 + n + m$ layers, where:

- $G_{V_1V_2} = \langle V_1 \cup V_2, E_i, \lambda_v, \lambda_{e_{V_1V_2}} \rangle$ is a *bipartite layer* with $u \in V_1$ labeled with identifiers of type V_1, $v \in V_2$ labeled with identifiers of type V_2, $\forall (u, v) \in E, u \in V_1, v \in V_2$ and $\lambda_{e_{V_1V_2}} : E_{V_1V_2} \mapsto \{0, 1\}$,
- $G^i_{V_1} = \langle V_1, E^i_{V_1}, \lambda_v, \lambda_{e^i_{V_1}} \rangle, \lambda_{e^i_{V_1}} : E^i_{V_1} \mapsto [0, 1]$ are layers of type V_1,
- $G^i_{V_2} = \langle V_2, E^i_{V_2}, \lambda_v, \lambda_{e^i_{V_2}} \rangle, \lambda_{e^i_{V_2}} : E^i_{V_2} \mapsto [0, 1]$ are layers of type V_2,

Predict, whether for a given $(u, v) \notin E, u \in V_1, v \in V_2$ $\lambda_e((u, v)) = 1$.

We limit ourselves to predict whether there is an activity or not, leaving the prediction of its *strength* as future work.

4 OUR APPROACH

In our problem setting, we want to predict links between two distinct types of nodes, e.g. drugs and targets in our experiments (see Fig. 1). To achieve this, we perform community discovery using an existing community detection approach, then exploit the discovered communities to solve the link prediction task.

4.1 Link prediction by community detection in a bipartite setting

To be able to use existing link-prediction-by-community-detection measures, we have to adapt them to the bipartite setting. Due to the construction of the networks we use and the community detection methods we evaluate, resulting communities can be *mixed*, i.e. containing both types of

[1] All nodes are shared between all layers
[2] Each node is present only in a single layer
[3] Inter-layer edges, that cross layers, are only between nodes and their counterparts
[4] Diagonal couplings for which all possible inter-layer edges are present

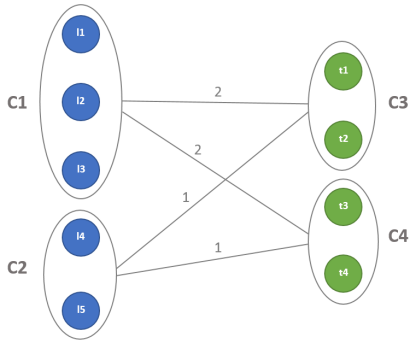

Figure 2: An example of *community to community* matching (label on edges represent number of existing edges between vertices of matched communities).

nodes, drugs and targets, as well as *pure*, of either type, drugs or targets only. Also, the community detection methods we use produce *non-overlapping* communities only. Mixed communities can be exploited directly with existing measures for link prediction via community detection (see Section 2), but pure communities cannot, ignoring a large number of drug-target pairs. To overcome this, we treat all communities as non-mixed and split mixed communities into pure ones. Notably, this split does not have to be done explicitly, but a mixed community can be treated as two pure ones with links between them. We exploit discovered communities in one of two proposed ways: by matching "community to community" or "node to community".

4.1.1 Community to community. In this case, each drug community is paired with each target community, then an adapted measure is used to perform link prediction between paired communities. Each non-interacting drug-target pair between paired communities is assigned the same link probability score. At the end of the matching, each non-interacting drug-target pair from the network will have been assigned a single score, which can be used to rank predictions. We refer to this approach as *community to community* (or *CC*).

In the example shown in Figure (Fig. 2), for instance, community $C1$ is twice connected to community $C3$ and twice to community $C4$. $C2$, on the other hand, is connected once to $C3$ and once to $C4$. This matching therefore implicitly assumes that all ligands in $C1$ have a connection of strength two to all targets in $C3$ etc. The big advantage of this matching is that even vertices that have no bipartite connection at all can receive a positive score.

4.1.2 Node to community. Another way of exploiting communities is to pair each node of one type with communities of the other type. The advantage of that method is that for a selected drug d_i and target t_j, the prediction can be made twice: once analyzing connections of a drug with target communities and second analyzing target connections with drug communities, providing a more reliable estimate. This approach is also referred to as Bipartite Local Models [2].

Figure 3 illustrates an NC matching that is equivalent to the CC matching in Figure 2: $l1$ is connected twice to $C3$ and once to $C4$, and $l2$ once to $C3$, while $t1$ is connected twice to $C1$ and once to $C2$, and $t2$ once to $C1$, for instance. For the rest of the figure: l3 is connected once to C4, l4 once to C3 and once to C4, l5 neither to C3 nor C4. t3 is connected once to C1, once to C2, t4 is connected once C1, once to C2.

Vertices belonging to the same community will therefore not necessarily receive the same score but a vertex such as $l5$ will be strongly punished because it is not connected with a bipartite edge. The link probability score between d_i and t_j is computed by aggregating the two results [4]. We report results using *mean* as an aggregation function. Our experiments showed that the difference between *max* and *mean* is negligible, and we use *mean* to get a more reliable result. We do not consider *min* as aggregator, because in the case of no evidence for existence of the link in one of the independent predictions the combined probability is also 0. We refer to this approach as *node to community* (or *NC*).

4.2 Adapting existing link prediction measures

We divide all existing link prediction measures into two broad categories: neighborhood measures and others, which we refer as community-based. The first group of measures are based on the notion of neighborhood, i.e. the set of vertices directly connected to the examined vertices. The semantic similarity between neighborhood and community, i.e. sets of vertices in both cases, allows us to use neighborhood measures in our setting. The other measures are not based on a notion of neighborhood, but on other metrics, and thus are grouped into a separate group in our work.

4.2.1 Neighborhood measures. Many existing link prediction measures exploit the neighborhoods of vertices e.g. in the form of common neighbors (CN), the Jaccard coefficient (JC), preferential attachment (PA), or SimRank (SR) [25]:

$$CN(d_i, t_j) = |\{v \mid (d_i, v) \in E\} \cap \{u \mid (t_j, u) \in E\}|, \quad (1)$$

$$JC(d_i, t_j) = \frac{CN(d_i, t_j)}{|\{v \mid (d_i, v) \in E\} \cup \{u \mid (t_j, u) \in E\}|}, \quad (2)$$

$$PA(d_i, t_j) = \Gamma(d_i) \cdot \Gamma(t_j), \text{with } \Gamma(d_i) = \sum_{k=1}^{|V|} A_{ik}, \Gamma(t_j) = \sum_{k=1}^{|V|} A_{jk}, \quad (3)$$

$$SR(d_i, t_j) = \frac{CN(d_i, t_j)}{PA(d_i, t_j)}. \quad (4)$$

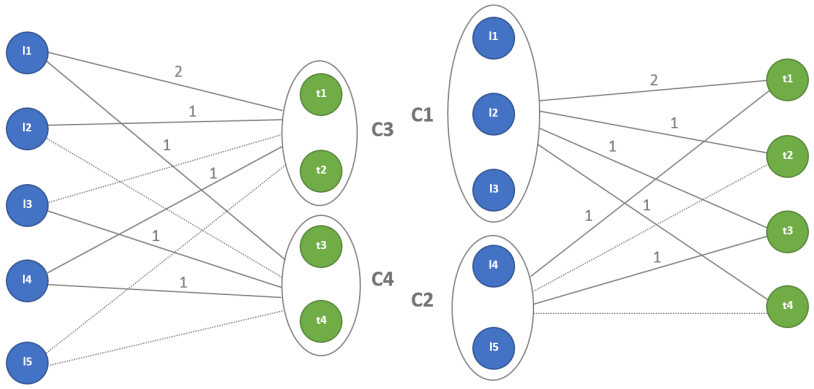

Figure 3: An example of *node to community* matching (labels on edges represent number of existing edges between vertices and matched communities, dotted edges represent non-interacting pairing)

Γ denotes the weight of a neighborhood of a vertex, which for the individual vertices is equivalent to their degree.

Due to the nature of communities we obtain, there is little overlap between vertices' neighborhoods, preventing the direct use of neighborhood-based measures. To overcome this, we adapt neighborhood measures for use with our communities, treating them like neighborhoods: the CN measure turns the number of common neighbors of communities d_i and t_j into the number of connections, JC represents the fraction of all possible connections of d_i and t_j that are connected to both, PA is defined by a product of degrees of communities d_i and t_j, finally SR is equal to the number of connections of communities d_i and t_j normalized by the product of their degrees. Using the *CC* and *NC* formulations, our bipartite adaptations take the form:

(1) Instead of the measures from Eq. 1-4 we define CN_{CC}, JC_{CC}, PA_{CC} and SR_{CC} versions corresponding to *CC* matching:

$$CN_{CC}(d_i, t_j) = |\{(d, t) \in E \mid d \in C(d_i), t \in C(t_j)\}|, \quad (5)$$

$$JC_{CC}(d_i, t_j) = \frac{CN_{CC}(d_i, t_j)}{|C(d_i)| \cdot |C(t_j)|}, \quad (6)$$

$$PA_{CC}(d_i, t_j) = \Gamma(C(d_i)) \cdot \Gamma(C(t_j)), \quad (7)$$

$$SR_{CC}(d_i, t_j) = \frac{CN_{CC}(d_i, t_j)}{PA_{CC}(d_i, t_j)}, \quad (8)$$

where $C(d_i)$, $C(t_j)$ represent communities of a drug d_i and a target t_j respectively. We overload Γ for communities as $\Gamma(C(v)) = \sum_{v \in C(v), u \notin C(v)} w(v, u)$.

(2) The *NC* version of CN is defined as the average of the two independent predictions, $CN_{NC}(d_i) = |\{t \mid (d_i, t) \in E, t \in C(t_j)\}|$, $CN_{NC}(t_j) = |\{d \mid (t_j, d) \in E, d \in C(d_i)\}|$, for d_i, and t_j respectively:

$$CN_{NC}(d_i, t_j) = \frac{1}{2}\left(CN_{NC}(d_i) + CN_{NC}(t_j)\right). \quad (9)$$

In the same manner, we can define *NC* versions for other measures from [25], the JC (Eq. 10), PA (Eq. 11) and SR (Eq. 12), taking into account that $PA_{NC}(d_i) = \Gamma(d_i) \cdot \Gamma(C(t_j))$ and $PA_{NC}(t_j) = \Gamma(t_j) \cdot \Gamma(C(d_i))$:

$$JC_{NC}(d_i, t_j) = \frac{1}{2}\left(\frac{CN_{NC}(d_i)}{|C(t_j)|} + \frac{CN_{NC}(t_j)}{|C(d_i)|}\right), \quad (10)$$

$$PA_{NC}(d_i, t_j) = \frac{1}{2}(PA_{NC}(d_i) + PA_{NC}(t_j)), \quad (11)$$

$$SR_{NC}(d_i, t_j) = \frac{1}{2}\left(\frac{CN_{NC}(d_i)}{PA_{NC}(d_i)} + \frac{CN_{NC}(t_j)}{PA_{NC}(t_j)}\right). \quad (12)$$

4.2.2 Community-based measures. Other measures proposed in the literature are based on one or several of the following assumptions: all vertices have the same semantic, all edges have the same semantic, edges are unweighted, or vertices whose link is to be predicted find themselves in the same community. We therefore cannot use most of the measures proposed in the literature but we can adapt some to our bipartite setting.

(1) Cannistraci *et al.* [5] in their *CAR-based* measures propose to exploit the density of communities to reward (or penalize) densely (sparsely) connected neighbors of the vertices whose link is to be predicted. Our adapted *CAR-based common neighbors* (CCN) will be defined as CN regularized by community local degree, in turn defined as the sum of weights of all edges inside community. The *NC* formulation of CCN takes the form:

$$CCN_{NC}(d_i, t_j) = \frac{1}{2}\left(CCN_{NC}(d_i) + CCN_{NC}(t_j)\right), \quad (13)$$

with $CCN_{NC}(d_i)$ and $CCN_{NC}(t_j)$ in turn defined as:

$$CCN_{NC}(d_i) = |\{t \mid (d_i, t) \in E, t \in C(t_j)\}| \cdot \sum_{t_l, t_k \in C(t_j)} A(t_l, t_k) \text{ and}$$

$$CCN_{NC}(t_j) = |\{t \mid (t_j, d) \in E, d \in C(d_i)\}| \cdot \sum_{d_l, d_k \in C(d_i)} A(d_l, d_k).$$

In the same manner, the *CAR-based Jaccard coefficient* (CJC) is redefined as CCN normalized by the size of the community:

$$CJC_{NC}(d_i, t_j) = \frac{1}{2}\left(CJC_{NC}(d_i) + CJC_{NC}(t_j)\right), \quad (14)$$

$$\text{with } CJC_{NC}(d_i) = \frac{CCN_{NC}(d_i)}{|C(t_j)|} \text{ and}$$

$$CJC_{NC}(t_j) = \frac{CCN_{NC}(t_j)}{|C(d_i)|}.$$

(2) Xie *et al.* [34] propose to exploit the connection of vertices to communities, summing over all communities. Our adaptation of their measure, which we refer to as *Neighboring community-based* (NCB) is defined as the normalized sum of all CN regularized by the size of the respective community. The *NC* formulation of this measure will take the form:

$$NCB_{NC}(d_i, t_j) = \frac{1}{2}\left(NCB_{NC}(d_i) + NCB_{NC}(t_j)\right), \quad (15)$$

with $NCB_{NC}(d_i)$ and $NCB_{NC}(t_j)$ in turn:

$$NCB_{NC}(d_i) = \sum_{k=1}^{c_t} \frac{|\{t \mid (d_i, t) \in E, t \in C_k\}|}{|C_k|} \cdot \frac{|\{t \mid t \in C_k\}|}{|T|} \text{ and}$$

$$NCB_{NC}(t_j) = \sum_{k=1}^{c_d} \frac{|\{d \mid (t_j, d) \in E, d \in C_k\}|}{|C_k|} \cdot \frac{|\{d \mid d \in C_k\}|}{|D|}.$$

Moreover, assuming communities are pure, i.e. consisting only of either drugs or targets, these equations can be simplified to the sum of all CN normalized by the number of vertices of one type:

$$NCB_{NC}(d_i) = \frac{1}{|T|} \sum_{k=1}^{c_t} |\{t \mid (d_i, t) \in E, t \in C_k\}|,$$

$$NCB_{NC}(t_j) = \frac{1}{|D|} \sum_{k=1}^{c_d} |\{d \mid (t_j, d) \in E, d \in C_k\}|.$$

(3) Ding *et al.* [12], finally, propose to exploit the neighborhoods of *communities*. Our adaptation of their measure, called *Community relevance Jaccard coefficient* (CRJC), is defined as the number of common nodes of examined communities and nodes of the opposite type connected to those communities normalized by the total number of nodes in this selection. The adapted measure is better suited to a *CC* formulation:

$$CRJC_{CC}(d_i, t_j) = \frac{|CRJC_{CC}(d_i) \cap CRJC_{CC}(t_j)|}{|CRJC_{CC}(d_i) \cup CRJC_{CC}(t_j)|}, \quad (16)$$

with $CRJC_{CC}(d_i)$ and $CRJC_{CC}(t_j)$ in turn:

$$CRJC_{CC}(d_i) = \{t \mid (d, t) \in E, d \in C(d_i)\} \cup \{d \mid d \in C(d_i)\} \text{ and}$$

$$CRJC_{CC}(t_j) = \{d \mid (t, d) \in E, t \in C(t_j)\} \cup \{t \mid t \in C(t_j)\}.$$

5 EXPERIMENTAL EVALUATION

To evaluate the two community detection methods, effects of their parameter settings, and prediction measures defined in the preceding section, we performed experiments on several benchmark data sets for drug-target activity prediction.

5.1 Experimental setup

We begin by evaluating the different measures described in Section 4 with two common community detection approaches, keeping most of the parameters fixed, and select the best performing measure. Following this, we show how an internal cross-validation can be used to fixed a methods' parameters, and report on their results, which we compare to the state-of-the-art.

We also perform experiments on the larger and more challenging IUPHAR data set and give some scalability results. The results of those experiments can be found in our supplementary material[5].

5.1.1 Community detection methods. We test our approach with spectral partitioning [24] and the Louvain algorithm [3] as community detection methods. The first finds the best cut to partition nodes based on eigenvalues of the Laplacian matrix and a threshold method [13, 30], the second greedily optimizes modularity, a generic measure to determine the quality of any partition produced by a community detection method [13, 28]. We apply spectral partitioning to multilayer graphs by "flattening" the graph, i.e. summing edge weights to derive the adjacency and degree matrices before performing partitioning. Since Louvain does not employ matrices, we translate the graph into a single-layer graph by summing up the weights of all edges between two vertices.

5.1.2 Parameters to optimize. Spectral partitioning has two parameters: the value of m and the thresholding method. The m parameter represents the number of eigenvectors corresponding to the m smallest non-zero eigenvalues used to partition the graph into at most 2^m groups. As thresholding methods we can use *sign cut* (or *default*), which partitions entries based on whether they are greater or less than zero, *bisection cut* (or *median*), using the median value of entries in an eigenvector as a threshold, producing two components of approximately equal size [16]. We also evaluate *mean*, which uses the average, and *sum* that exploits the fact that there are approximately equally as many positive and negative values in eigenvectors of Laplacian matrix, i.e. in practice they sum to a value close to zero. Moreover, the same threshold can be applied to all eigenvectors, or each individual eigenvector can have its own threshold. We call the former approach *global*, and the latter *individualized*. Additionally, the global

[5] https://zimmermanna.users.greyc.fr/supplementary-material.html

threshold can be computed by applying the aggregating function (mean, median or sum) to all eigenvectors or only to the m actually used. We refer to this latter type as *localized*. To sum up, we evaluate 9 different thresholding methods: global, localized, individualized and their combinations with mean, median and sum. The combination *global sum* is a special case since taking the first eigenvector, whose entries all have the same, positive value, into account violates the "close to zero" property sum thresholding exploits. We therefore do not evaluate that thresholding method, but add *default* thresholding to the mix for the experimental evaluation.

The Louvain algorithm has only one parameter – *resolution limit* – which defines a modularity scale. Practically speaking, at different moments of time t, the difference between optimal partitioning and partitioning produced by the Louvain is various and the resolution parameter represents this change in time [23].

5.1.3 Data sets. We perform our experiments on the data sets introduced in [35]: Enzyme, G-protein coupled receptors (GPCR), Ion Channels (IC) and Nuclear Receptors (NR). In addition, we use the Kinase set [9]. These data sets have been used in prior work on drug-target interaction prediction [4, 6, 26, 37], and can be considered benchmarks. The data consist of 3 networks: drug similarities, target similarities and drug-target interaction (the bipartite graph). The data sets' basic properties are presented in Table 1.

5.1.4 Evaluation protocol and quality measures. To perform evaluation of our experiments, we performed a 5×5-fold cross-validation, with each fold containing 20% of all drug-target interactions, acting as test set for link prediction once, while community detection is performed on the other 80%. The process is repeated 5 times, the results are averaged among all runs. We evaluate all the predictions by AUC (Area Under ROC Curve) and AUPR (Area Under Precision-Recall Curve), averaging the results.

5.1.5 Implementation. We implemented spectral partitioning and link prediction measures in Python[6], and we used python-louvain package as implementation of Louvain.

5.2 Experimental results

5.2.1 Link prediction measures evaluation. We first test the different link prediction measures from Section 4 on communities produced by either spectral partitioning or the Louvain algorithm. To reduce computational complexity, we use default parameters for community detection: *default* as threshold for spectral partitioning and 1.0 as resolution limit for the Louvain algorithm. Note that we optimize m for spectral partitioning on the test data in this experiment, because there is no a priori number of eigenvalues that will fit all data. The results are presented in Fig. 4, with *community to community* formulations on the left, *node to community* ones on the right of each plot. The best-performing measure for each group is indicated by a + sign over the corresponding bar. We also report on the figure the optimal m value for every measure in spectral partitioning.

The results show that a number of measures, e.g. SR_{CC}, $CN_{NC}, SR_{NC}, CCN_{NC}, CJC_{NC}, NCB_{NC}$, have acceptable performance in terms of AUC on most of data sets while the Jaccard coefficient usually performs best for both the *CC* and *NC* versions. These two, JC_{CC} and JC_{NC}, are therefore the measures we will use going forward. Another result is that for these parameter settings spectral partitioning and the Louvain algorithm give approximately the same AUC, but the latter improves on AUPR. Finally, using *node to community* predictions requires a lower m, i.e. less fine-grained partitions, for spectral partitioning.

5.2.2 Parameter selection via internal cross-validation and state of the art comparison. In link prediction, as in any other predictive task, the main issue is choosing parameter values, in the case of spectral partition m and the thresholding method, in the case of Louvain the resolution limit. In the absence of other knowledge, one would use *cross-validation* as a systematic method to optimize parameters. Several-fold cross-validation is also the method of choice to evaluate the performance of a classifier, however, so that we use a double cross-validation in this section: splitting off an *external* test fold (containing 20% of present edges) to evaluate the model, and using an *internal* five-fold cross-validation to fix the model's parameters.

During the internal cross-validation, we performed grid search over the different parameter settings, varying m in the interval $[1, 25]$, and testing this value with all nine options for thresholding. We report averaged results using mean.

Table 2 presents the results for spectral partitioning. It shows both the results of internal evaluation, i.e. on the validation set used to fix parameter values, and of the external evaluation, i.e. on the unseen training data. The main conclusion to draw is that those values align very closely w.r.t. AUC, i.e. that there is no risk of overfitting when building the model. Concerning AUPR, interestingly enough, the results on the testing folds are in fact *higher* than for the validation data used in the internal cross-validation. Notably, using *NC* matching always gives better results than *CC* matching, often by a large margin.

The only parameter to optimize for Louvain is the resolution parameter, which we vary in the interval $[0.1, 1]$ using steps of 0.1. As in the case of spectral partitioning, the performance of Louvain is rather close in terms of AUC but the differences in AUPR are even more pronounced (Table 3).

[6] https://zimmermanna.users.greyc.fr/supplementary-material.html

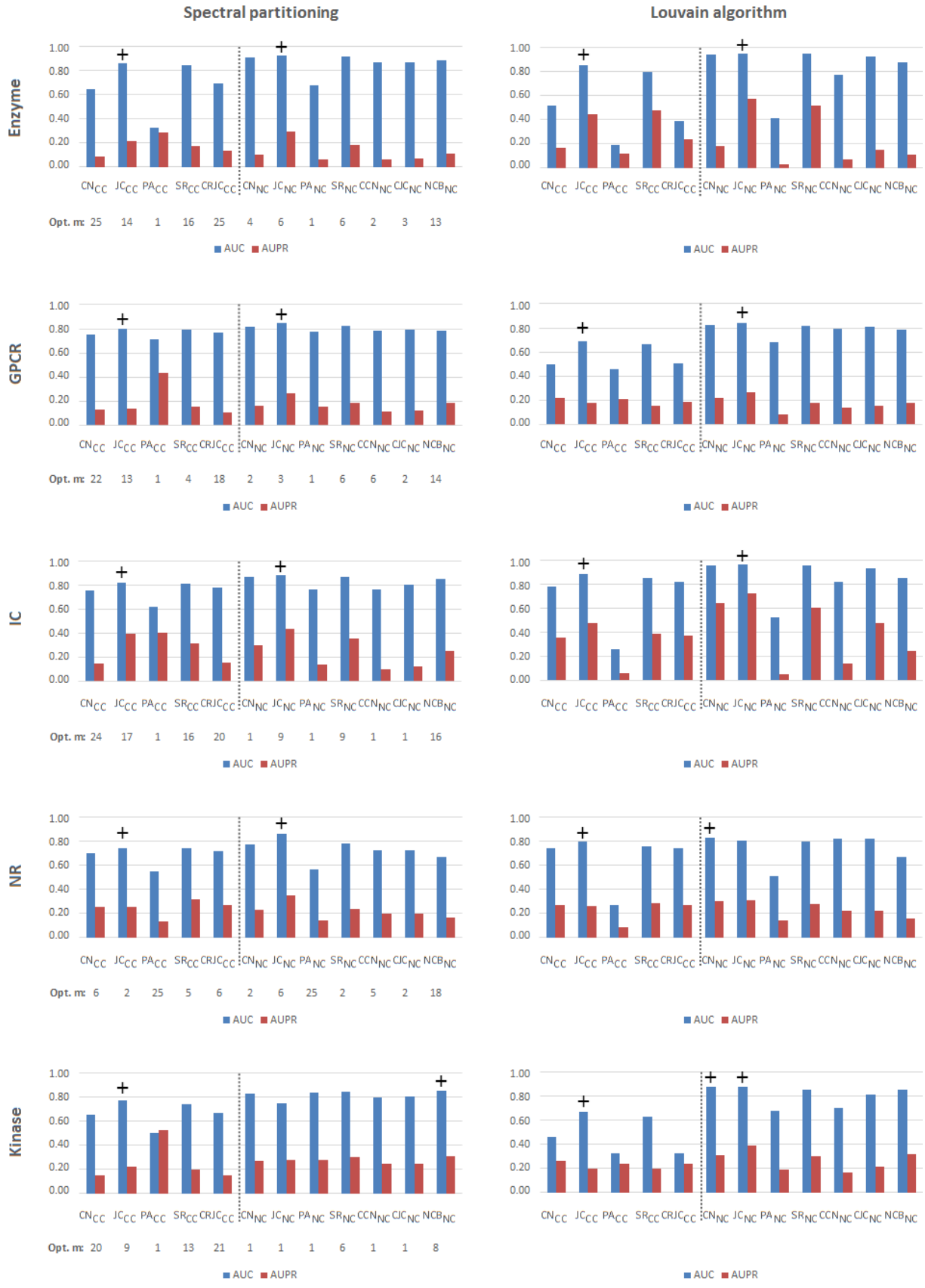

Figure 4: Link prediction measures evaluation on the benchmark data sets. The symbol + denotes the best performing measure for each group of formulations in each data set.

Table 1: Basic properties of Benchmark and IUPHAR data sets and running times (* – for 1 fold in average with Spectral partitioning and JC_{CC} measure)

Data set	Drugs	Targets	Interactions	Layers	$\|V\|$	$\|E\|$	Sparsity	CC	Running time*, s
Enzyme	445	664	2926	3	1109	321832	0.524	1	58.67
GPCR	223	95	635	3	318	29853	0.592	1	3.99
IC	210	204	1476	3	414	44127	0.516	1	7.06
NR	54	26	90	3	80	1846	0.584	1	0.15
Kinase	68	442	1527	3	510	101266	0.780	1	9.94
IUPHAR	8137	2502	12456	6	10639	26706838	0.472	1	3477.8

Table 2: Spectral partitioning parameter optimization via internal cross-validation

Data set	Measure	Internal		External	
		AUC	AUPR	AUC	AUPR
Enzyme	JC_{CC}	0.85	0.14	0.85	0.19
	JC_{NC}	0.91	0.19	0.92	0.26
GPCR	JC_{CC}	0.79	0.11	0.80	0.15
	JC_{NC}	0.84	0.19	0.85	0.25
IC	JC_{CC}	0.82	0.31	0.83	0.40
	JC_{NC}	0.87	0.32	0.88	0.41
NR	JC_{CC}	0.73	0.21	0.72	0.24
	JC_{NC}	0.75	0.19	0.77	0.23
Kinase	JC_{CC}	0.76	0.17	0.77	0.23
	JC_{NC}	0.85	0.26	0.86	0.35

Table 3: Louvain algorithm resolution optimization via internal cross-validation

Data set	Measure	Internal		External	
		AUC	AUPR	AUC	AUPR
Enzyme	JC_{CC}	0.94	0.52	0.96	0.69
	JC_{NC}	0.95	0.56	0.96	0.71
GPCR	JC_{CC}	0.73	0.18	0.80	0.31
	JC_{NC}	0.85	0.28	0.89	0.49
IC	JC_{CC}	0.94	0.50	0.95	0.65
	JC_{NC}	0.96	0.61	0.97	0.78
NR	JC_{CC}	0.79	0.27	0.82	0.45
	JC_{NC}	0.81	0.30	0.80	0.42
Kinase	JC_{CC}	0.72	0.15	0.73	0.19
	JC_{NC}	0.90	0.38	0.91	0.50

GPCR is a bit of an outlier for this experiment in that the differences between internal quality estimation and test fold results are larger than for the other data sets. The superior performance of *NC* matching holds, however.

Concerning comparison with the state-of-the-art, the Louvain using JC_{NC} comes close to the performance of the state-of-the-art methods reported in [4] in terms of AUC: 0.97 compared to 0.96 for our method with Louvain for Enzyme, 0.95 compared to 0.89 for GPCR, 0.98 compared to 0.97 for IC, 0.88 compared to 0.82 for NR, and 0.9 compared to 0.91 for Kinase.

5.2.3 Baseline comparison. Our problem setting was addressed in [21], using a random walk approach which is basically an extension of PageRank for any number of layers, and thus can be considered as a baseline in this work. JC_{CC} and JC_{NC} in combination with both spectral partitioning and the Louvain clearly outperform that baseline in terms of AUC: 0.84 *vs* 0.96 (Louvain) and 0.92 (spectral partitioning) for Enzyme, 0.8 *vs* 0.89 and 0.85 for GPCR, 0.76 *vs* 0.97 and 0.88 for IC, 0.63 *vs* 0.82 and 0.77 for NR, and 0.61 *vs* 0.91 and 0.86 for Kinase, respectively.

5.2.4 Interpretability. Our approach offers a straightforward option for interpretation/explainability of a link prediction: for each of the two vertices, we can show the communities they belong to, the weights of intra-community edges, the number and layout of inter-community edges, and their numerical translation by the measure and matching technique. This is a possibility that is not available for recent, well-performing techniques based on graph embeddings.

6 CONCLUSION AND PERSPECTIVES

We have presented a framework for link prediction in bipartite multi-layer graphs using graph community structure and link prediction measures adapted from those proposed in the literature. We have found empirically that combining the well-known and relatively straightforward Jaccard coefficient, particularly in a BLM formulation, with the Louvain algorithm for community detection allows us to achieve results that are competitive with the state-of-the-art. We have also demonstrated that it is possible to set the parameter values of the community detection techniques via internal cross-validation and that they transfer well to unseen data.

In our supplementary material[7], we show the evaluation of our approach on the much larger and sparser IUPHAR data

[7] https://zimmermanna.users.greyc.fr/supplementary-material.html

set. Using those data, we have verified predicted interactions in terms of their biological semantics, shown that external validation aligns well with validation using known labels in test data, and assessed scalability.

We have limited ourselves to two easy-to-use community detection methods in this work, and will evaluate the use of other methods in the future. We also intend to perform experiments on non-biological data to test generic side of our approach. As we mentioned in Section 3, so far we have only looked at predicting interaction as a binary setting but the more challenging setting would be to predict the *strength* of the interaction. Finally, we intend to add layers derived from other information sources to the networks and use our approach to identify possible redundancies among them.

REFERENCES

[1] Y. Ahn, J. P. Bagrow, and S. Lehmann. 2010. Link communities reveal multiscale complexity in networks. *Nature* 466, 7307 (2010), 761.

[2] K. Bleakley and Y. Yamanishi. 2009. Supervised prediction of drug–target interactions using bipartite local models. *Bioinf.* 25, 18 (2009), 2397–2403.

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

[4] K. Buza and L. Peska. 2017. ALADIN: A New Approach for Drug–Target Interaction Prediction. In *ECML/PKDD*. Springer, 322–337.

[5] C. V. Cannistraci, G. Alanis-Lobato, and T. Ravasi. 2013. From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks. *Scientific reports* 3 (2013), 1613.

[6] X. Chen, M. X. Liu, and G. Y. Yan. 2012. Drug–target interaction prediction by random walk on the heterogeneous network. *Molecular BioSystems* 8, 7 (2012), 1970–1978.

[7] F. Cheng, Y. Zhou, W. Li, G. Liu, and Y. Tang. 2012. Prediction of chemical-protein interactions network with weighted network-based inference method. *PloS one* 7, 7 (2012), e41064.

[8] A. Clauset, C. Moore, and M. Newman. 2008. Hierarchical structure and the prediction of missing links in networks. *Nature* 453, 7191 (2008), 98.

[9] M. I. Davis, J. P. Hunt, S. Herrgard, P. Ciceri, L. M. Wodicka, G. Pallares, M. Hocker, D. K. Treiber, and P. P. Zarrinkar. 2011. Comprehensive analysis of kinase inhibitor selectivity. *Nature Biotech.* 29, 11 (2011), 1046.

[10] C. De Bacco, E. A. Power, D. B. Larremore, and C. Moore. 2017. Community detection, link prediction, and layer interdependence in multilayer networks. *Phys. Review E* 95, 4 (2017), 042317.

[11] H. Ding, I. Takigawa, H. Mamitsuka, and S. Zhu. 2013. Similarity-based machine learning methods for predicting drug–target interactions: a brief review. *Briefings in bioinformatics* 15, 5 (2013), 734–747.

[12] J. Ding, L. Jiao, J. Wu, and F. Liu. 2016. Prediction of missing links based on community relevance and ruler inference. *Knowledge-Based Systems* 98 (2016), 200–215.

[13] S. Fortunato. 2010. Community detection in graphs. *Phys. Reports* 486, 3-5 (2010), 75–174.

[14] J. H. Gallier. 2013. Notes on Elementary Spectral Graph Theory. Applications to Graph Clustering Using Normalized Cuts. *CoRR* abs/1311.2492 (2013). arXiv:1311.2492

[15] P. Goyal and E. Ferrara. 2018. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems* 151 (2018), 78–94.

[16] S. Guattery and G. L. Miller. 1995. On the performance of spectral graph partitioning methods. In *SODA*, Vol. 95. 233–242.

[17] R. Guimerà and M. Sales-Pardo. 2009. Missing and spurious interactions and the reconstruction of complex networks. *Proc. of the Nat. Academy of Sciences* 106, 52 (2009), 22073–22078.

[18] D. Hristova, A. Noulas, C. Brown, M. Musolesi, and C. Mascolo. 2016. A multilayer approach to multiplexity and link prediction in online geo-social networks. *EPJ Data Science* 5, 1 (2016), 24.

[19] M. Jalili, Y. Orouskhani, M. Asgari, N. Alipourfard, and M. Perc. [n. d.]. Link prediction in multiplex online social networks. ([n. d.]).

[20] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter. 2014. Multilayer networks. *J. Complex Networks* 2, 3 (2014), 203–271.

[21] M. Koptelov, A. Zimmermann, and B. Crémilleux. 2018. Link Prediction in Multi-layer Networks and Its Application to Drug Design. In *IDA*. Springer, 175–187.

[22] Z. Kuncheva and G. Montana. 2015. Community detection in multiplex networks using locally adaptive random walks. In *ASONAM*. ACM, 1308–1315.

[23] R. Lambiotte, J. C. Delvenne, and M. Barahona. 2008. Laplacian dynamics and multiscale modular structure in networks. (2008).

[24] J. Leskovec, A. Rajaraman, and J. D. Ullman. 2014. *Mining of massive datasets*. Cambridge university press.

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

[26] H. Lim, P. Gray, L. Xie, and A. Poleksic. 2016. Improved genome-scale multi-target virtual screening via a novel collaborative filtering approach to cold-start problem. *Scientific reports* 6 (2016), 38860.

[27] Z. Liu, J. L. He, K. Kapoor, and J. Srivastava. 2013. Correlations between community structure and link formation in complex networks. *PloS one* 8, 9 (2013), e72908.

[28] M. E. J. Newman and M. Girvan. 2004. Finding and evaluating community structure in networks. *Phys. Review E* 69, 2 (2004), 026113.

[29] S. Soundarajan and J. Hopcroft. 2012. Using community information to improve the precision of link prediction methods. In *WWW*. ACM, 607–608.

[30] D. A. Spielman and S. H. Teng. 2007. Spectral partitioning works: Planar graphs and finite element meshes. *Linear Algebra Appl.* 421, 2-3 (2007), 284–305.

[31] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos. 2005. Neighborhood formation and anomaly detection in bipartite graphs. In *ICDM*. IEEE, 418–425.

[32] A. Tagarelli, A. Amelio, and F. Gullo. 2017. Ensemble-based community detection in multilayer networks. *DMKD* 31, 5 (2017), 1506–1543.

[33] J. C. Valverde-Rebaza and A. de Andrade Lopes. 2014. Link prediction in online social networks using group information. In *ICCSA*. Springer, 31–45.

[34] Z. Xie, E. Dong, J. Li, D. Kong, and N. Wu. 2014. Potential links by neighbor communities. *Physica A: Statistical Mechanics and its Applications* 406 (2014), 244–252.

[35] Y. Yamanishi, M. Araki, A. Gutteridge, W. Honda, and M. Kanehisa. 2008. Prediction of drug–target interaction networks from the integration of chemical and genomic spaces. *Bioinfo.* 24, 13 (2008), i232–i240.

[36] B. Yan and S. Gregory. 2012. Finding missing edges in networks based on their community structure. *Phys. Review E* 85, 5 (2012), 056112.

[37] X. Zheng, H. Ding, H. Mamitsuka, and S. Zhu. 2013. Collaborative matrix factorization with multiple similarities for predicting drug-target interactions. In *KDD*. ACM, 1025–1033.