Link Prediction via Community Detection in Bipartite Multi-Layer Graphs

Maksim Koptelov, Albrecht Zimmermann, Bruno Cremilleux

Link Prediction via Community Detection in Bipartite Multi-Layer Graphs

Maksim Koptelov[0000−0001−9065−2827], Albrecht Zimmermann[0000−0002−8319−7456], and Bruno Crémilleux[0000−0001−8294−9049]

Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France
{maksim.koptelov,albrecht.zimmermann,bruno.cremilleux}@unicaen.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.

Keywords: Link prediction · Community detection · Multi-layer graphs.

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, thus making the use of most existing link prediction methods in a straightforward way 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 [6, 26, 4].

To address these restrictions, we take inspiration from existing methods that use community detection to perform link prediction [17, 36, 29, 33]. While this decouples the problem into how to find communities in multilayer graphs, and how to exploit them for prediction, existing link prediction measures [5, 34, 12] 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 best measure and community detection approach combination. In addition, we demonstrate that the parameter settings of the spectral partitioning method can be effectively set via internal cross-validation.

The rest of the paper is organized as follows. Section 2 provides basic notations and definitions. Section 3 discusses related work on link prediction and community detection in multi-layer networks. 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 the future work.

2 Definitions

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 the vertices of which can be divided into two classes V_1 and V_2 such that there is no edge between vertexes 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 means no interaction between vertices, 1 confirmed interaction, and an intermediate value represents interaction probability. An *unweighted graph* is one where every edge is labeled by 1.

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 the vertices of which can be divided into two classes, and where exactly one of the layers is a bipartite graph. Note, according to the classification of Kivel *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, 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

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

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 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 based, random-walker based, 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 [11, 4]. 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 we extended the approach 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.

[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

The authors of [17, 36, 33] combine community detection with existing edge prediction methods to improve the 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 than elsewhere. Missing edges are predicted by node similarity using nearest-neighbor measures [36], Stochastic Block Models [17], or in-group and 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 to predict links [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, 34, 12]. 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 [22, 32, 10]. 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 approaches due to their ease of use and potentially low computational complexity.

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. To achieve this, we perform communities discovery using en exsting community detection approach, then exploit the discovered communities to solve the link prediction task.

4.1 Link prediction by community detection in 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 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 3), 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 the two proposed ways: by matching "community to community" or "node to community".

Community to community In this case, each drug community is paired with each target community, then an adapted measure 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*) formulation.

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. The approach based on this idea is called Bipartite Local Models [2]. 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 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*) formulation.

4.2 Existing link prediction measures adaptation

We divide all existing link prediction measures into two 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 separate group in our work.

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), a preferential attachment measure (PA), or SimRank (SR) [25]:

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

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

$$PA(d_i, t_j) = |\Gamma(d_i)| \cdot |\Gamma(t_j)|,\ |\Gamma(v)| = deg(v), \tag{3}$$

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

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:

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

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

$$PA_{CC}(d_i, t_j) = |\Gamma_C(d_i)| \cdot |\Gamma_C(t_j)|, \tag{7}$$

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

   where Γ_C represents the neighborhood of a *community*, the degree of which we defined in accordance with [8] as $|\Gamma_C(v)| = \sum_{d_l \in C(v), d_k \notin C(v)} w(d_l, d_k)$.

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:

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

   In the same manner, we can define *NC* versions for other measures from [25], the Jaccard coefficient (Eq. 10), preferential attachment (Eq. 11) and SimRank (Eq. 12), taking into account that $PA_{NC}(d_i) = \sum_{d_k} A(d_i, d_k) \cdot |\Gamma_C(t_j)|$ and $PA_{NC}(t_j) = \sum_{t_k} A(t_j, t_k) \cdot |\Gamma_C(d_i)|$:

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

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

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

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, which is in turn defined as the sum of weights of all edges inside community. The *NC* formulation of CCN takes a form:

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

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

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

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

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

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

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

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:

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

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

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

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

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, which the authors refer to as *Community relevance Jaccard coefficient* (CRJC), is defined as a 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:

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

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

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

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

5 Experimental evaluation

To evaluate the two community detection algorithms, the effect of their parameter settings, and the prediction measures defined in the preceding section, we performed experiments on a number of benchmark data sets for ligand-target activity prediction.

5.1 Experimental protocol

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 the methods' parameters, and report on their results. Finally, we show what happens if we optimized those parameters on the *test* data, giving us the arguably best results, and show that the internal cross-validation creates models of similar quality, i.e. using cross-validation is an appropriate method to fix parameters.

Community detection methods We test our approach with spectral partitioning [24] and the Louvain algorithm [3] as community detection approaches. The first finds the best cut to partition nodes based on eigenvalues of the Laplacian matrix and a threshold method [30, 13], while the second greedily optimizes modularity, generic measure to determine the quality of each partition produced by a community detection approach [28, 13]. We apply spectral partitioning to multi-layer graphs by flattening the graph, i.e. summing edge weights to derive the adjacency and degree matrices before performing partitioning. We apply the Louvain algorithm to multi-layer graphs by flattening the graph as well. Since the algorithm is not limited to adjacency and degree matrices, we create an instance of a single graph, multiple edges of which are aggregated with the sum function.

Parameters to optimize Spectral partitioning has two parameters: m value 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 thresholds 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, such that 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 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 different and the resolution parameter represents this change in time [23].

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 [6, 37, 26, 4], and can be considered benchmarks. The data consist of 3 networks: drug similarities, target similarities and drug-target interaction (the bipartite graph). The

interaction network is presented as binary relation lists, while the similarity data are presented by matrices. The data were transformed to adjacency lists, and for the sake of (computational) convenience we mapped ligand and drug names to consecutive integers, i.e. $l0, l1, l2, \ldots$, $t0, t1, t2, \ldots$, the required input for our implementation. The data sets' basic properties are presented in Table 1.

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

Evaluation protocol 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.

Quality measures We evaluate all the predictions by AUC (Area Under ROC Curve) and AUPR (Area Under Precision-Recall Curve), averaging the results. We also report standard deviation values when it is applicable.

Implementation We implemented spectral partitioning and all link prediction measures in Python[5], using the networkx library to model the multi-layer network, the python-louvain package as Louvain algorithm implementation, NumPy for all matrix computations and sklearn to calculate the curves.

5.2 Experimental results

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 of the community detection approaches: *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. 1, with *community to community* formulations on the left, *node to community* ones on the right of each plot. The best-performing

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

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.

Parameter selection for spectral partitioning via internal cross-validation 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 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.

Table 2 presents the results. 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, i.e. that there is not risk of overfitting when building the model.

Table 2. Spectral partitioning parameters optimization via Internal cross-validation

Data set	Measure	Internal validation				External validation			
		μ AUC	σ	μ AUPR	σ	μ AUC	σ	μ AUPR	σ
Enzyme	JC_{CC}	0.85	0.00	0.14	0.04	0.85	0.01	0.19	0.07
	JC_{NC}	0.91	0.00	0.19	0.02	0.92	0.01	0.26	0.06
GPCR	JC_{CC}	0.79	0.01	0.11	0.02	0.80	0.02	0.15	0.02
	JC_{NC}	0.84	0.01	0.19	0.01	0.85	0.01	0.25	0.02
IC	JC_{CC}	0.82	0.01	0.31	0.03	0.83	0.02	0.40	0.04
	JC_{NC}	0.87	0.00	0.32	0.06	0.88	0.01	0.41	0.05
NR	JC_{CC}	0.73	0.02	0.21	0.03	0.72	0.06	0.24	0.08
	JC_{NC}	0.75	0.03	0.19	0.02	0.77	0.06	0.23	0.13
Kinase	JC_{CC}	0.76	0.01	0.17	0.01	0.77	0.03	0.23	0.02
	JC_{NC}	0.85	0.00	0.26	0.02	0.86	0.01	0.35	0.02

Baseline comparison We have addressed this problem setting in prior work, using a random walk approach [21] which is basically an extension of PageRank for any number of layers, and thus can be considered as a baseline in this work.

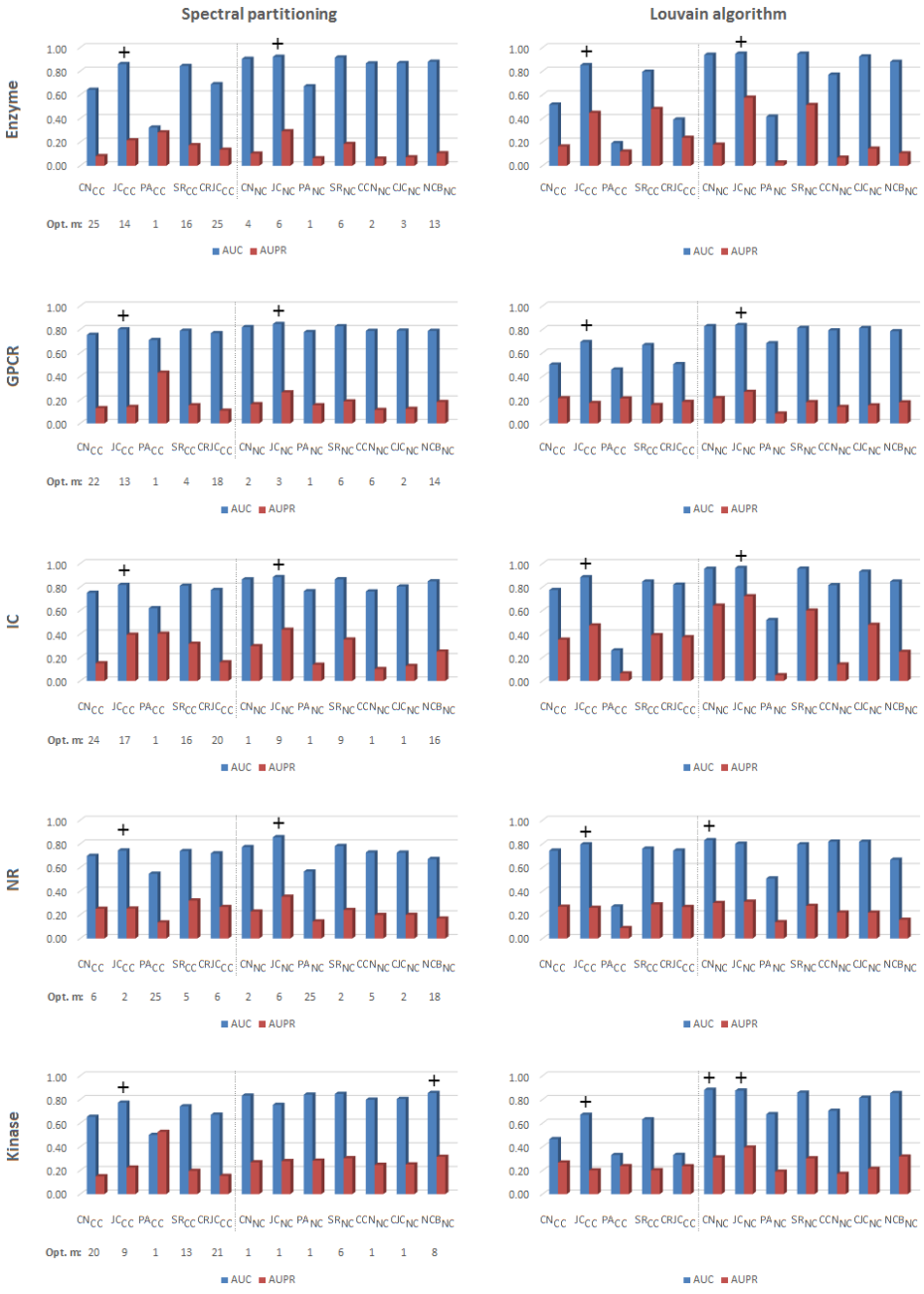

Fig. 1. Link prediction measures evaluation on the benchmark data sets. The symbol + denotes the best performing measure for each group of formulations in each data set.

JC_{CC} and JC_{NC} in combination with both spectral partitioning and the Louvain clearly outperform that baseline in terms of AUC (0.84, 0.8, 0.76, 0.63 and 0.61 for the Enzyme, GPCR, IC, NR, and Kinase data sets respectively) and AUPR (0.15, 0.22, 0.27, 0.26 and 0.13 for the Enzyme, GPCR, IC, NR, and Kinase data sets respectively).

Performance ceiling and state of the art comparison While we have shown above that using an internal cross-validation can be used to use our method's parameters, one could ask the question whether those results approach the best possible results on the data, and how they compare to the state of the art. To explore this issue, we performed parameter optimization on the *test data*, an unrealistic setting since we take the label information of unseen data into account. The performance in this setting can be interpreted as the ceiling of what is achievable.

Tables 3 and 4 list optimal parameter choices for the two community detection methods, as well as predictive performance. The results of Table 3 are very close to those of Table 2, showing that the internal cross validation gets close to the achievable ceiling.[6] We also report standard deviations, which show the model's performance is very stable.

Table 3. Spectral partitioning threshold and m value optimization

Data set	Measure	Optimal parameters		Performance			
		threshold	m value	AUC	σ	AUPR	σ
Enzyme	JC_{CC}	localized median	11	0.86	0.02	0.15	0.04
	JC_{NC}	individual mean	5	0.92	0.01	0.29	0.05
GPCR	JC_{CC}	localized median	17	0.80	0.03	0.17	0.03
	JC_{NC}	global mean	3	0.85	0.02	0.27	0.04
IC	JC_{CC}	individual median	13	0.84	0.03	0.47	0.09
	JC_{NC}	localized median	7	0.89	0.02	0.43	0.06
NR	JC_{CC}	global mean	4	0.77	0.08	0.29	0.11
	JC_{NC}	individual sum	2	0.78	0.05	0.25	0.09
Kinase	JC_{CC}	localized median	11	0.78	0.02	0.24	0.03
	JC_{NC}	localized mean	6	0.86	0.01	0.36	0.03

As one can see, the NC setting continues to outperform CC for all data sets. In addition, the Louvain algorithm provides not only better AUC than spectral partitioning, but significantly better AUPR, different from the results for the default parameter setting (Fig. 1).

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, 0.95, 0.98, 0.88 and 0.9 for the Enzyme, GPCR, IC, NR, and Kinase data sets respectively.

[6] We had to rerun internal parameter optimization for Louvain and the results were not yet available at the time of writing.

Table 4. Louvain algorithm resolution optimization

Data set	Measure	Optimal resolution	Performance			
			AUC	σ	AUPR	σ
Enzyme	JC_{CC}	0.4	0.95	0.01	0.68	0.03
	JC_{NC}	0.6	0.96	0.01	0.71	0.02
GPCR	JC_{CC}	0.7	0.83	0.03	0.35	0.05
	JC_{NC}	0.8	0.89	0.02	0.47	0.08
IC	JC_{CC}	0.4	0.95	0.01	0.68	0.03
	JC_{NC}	0.8	0.97	0	0.79	0.03
NR	JC_{CC}	0.8	0.83	0.06	0.31	0.09
	JC_{NC}	0.9	0.84	0.06	0.37	0.09
Kinase	JC_{CC}	0.7	0.73	0.01	0.19	0.01
	JC_{NC}	0.9	0.91	0.01	0.5	0.03

Interpretability Our approach offers a straightforward option for interpretation/explanainability 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.

Scalability In order to verify scalability of our approach we tested it on a bigger data set, IUPHAR, having 6 layers, and described in detail in [21]. The data set basic properties are presented in Table 1. We used spectral partitioning with default threshold and an optimal value of $m = 400$ optimized on the test data. We used JC, the best performing measure and CC as a matching technique for performance reasons. As for the smaller networks, the prediction quality (AUC=0.7365 $\pm$ 0.02 std, AUPR 0.01 $\pm$ 0.0 std) is much better than in our previous work [21] (AUC 0.5735) (which was derived from leave-out-out cross-validation). Running times for a single test fold, the majority of which is taken up by community matching, are shown in Table 1. We can see that the quotient of the number of IUPHAR edges to the number in smaller networks is lower than that of running times, indicating that the method scales when using spectral partitioning.

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 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 larger data sets that have shown themselves to be too computationally expensive for methods such as the one proposed in [4]. 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. Ahn, Y.Y., Bagrow, J.P., Lehmann, S.: Link communities reveal multiscale complexity in networks. nature **466**(7307), 761 (2010)
2. Bleakley, K., Yamanishi, Y.: Supervised prediction of drug–target interactions using bipartite local models. Bioinformatics **25**(18), 2397–2403 (2009)
3. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment **2008**(10), P10008 (2008)
4. Buza, K., Peska, L.: Aladin: A new approach for drug–target interaction prediction. In: ECML/PKDD. pp. 322–337. Springer (2017)
5. Cannistraci, C.V., Alanis-Lobato, G., Ravasi, T.: From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks. Scientific reports **3**, 1613 (2013)
6. Chen, X., Liu, M.X., Yan, G.Y.: Drug–target interaction prediction by random walk on the heterogeneous network. Molecular BioSystems **8**(7), 1970–1978 (2012)
7. Cheng, F., Zhou, Y., Li, W., Liu, G., Tang, Y.: Prediction of chemical-protein interactions network with weighted network-based inference method. PloS one **7**(7), e41064 (2012)
8. Clauset, A., Moore, C., Newman, M.E.: Hierarchical structure and the prediction of missing links in networks. Nature **453**(7191), 98 (2008)
9. Davis, M.I., Hunt, J.P., Herrgard, S., Ciceri, P., Wodicka, L.M., Pallares, G., Hocker, M., Treiber, D.K., Zarrinkar, P.P.: Comprehensive analysis of kinase inhibitor selectivity. Nature biotechnology **29**(11), 1046 (2011)
10. De Bacco, C., Power, E.A., Larremore, D.B., Moore, C.: Community detection, link prediction, and layer interdependence in multilayer networks. Phys. Review E **95**(4), 042317 (2017)
11. Ding, H., Takigawa, I., Mamitsuka, H., Zhu, S.: Similarity-based machine learning methods for predicting drug–target interactions: a brief review. Briefings in bioinformatics **15**(5), 734–747 (2013)
12. Ding, J., Jiao, L., Wu, J., Liu, F.: Prediction of missing links based on community relevance and ruler inference. Knowledge-Based Systems **98**, 200–215 (2016)
13. Fortunato, S.: Community detection in graphs. Phys. Reports **486**(3-5), 75–174 (2010)
14. Gallier, J.H.: Notes on elementary spectral graph theory. applications to graph clustering using normalized cuts. CoRR **abs/1311.2492** (2013)
15. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: A survey. Knowledge-Based Systems **151**, 78–94 (2018)
16. Guattery, S., Miller, G.L.: On the performance of spectral graph partitioning methods. In: SODA. vol. 95, pp. 233–242 (1995)
17. Guimerà, R., Sales-Pardo, M.: Missing and spurious interactions and the reconstruction of complex networks. Proc. of the Nat. Academy of Sciences **106**(52), 22073–22078 (2009)

18. Hristova, D., Noulas, A., Brown, C., Musolesi, M., Mascolo, C.: A multilayer approach to multiplexity and link prediction in online geo-social networks. EPJ Data Science **5**(1), 24 (2016)
19. Jalili, M., Orouskhani, Y., Asgari, M., Alipourfard, N., Perc, M.: Link prediction in multiplex online social networks. Royal Society open science **4**(2), 160863 (2017)
20. Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J.P., Moreno, Y., Porter, M.A.: Multilayer networks. J. Complex Networks **2**(3), 203–271 (2014)
21. Koptelov, M., Zimmermann, A., Crémilleux, B.: Link prediction in multi-layer networks and its application to drug design. In: IDA. pp. 175–187. Springer (2018)
22. Kuncheva, Z., Montana, G.: Community detection in multiplex networks using locally adaptive random walks. In: ASONAM. pp. 1308–1315. ACM (2015)
23. Lambiotte, R., Delvenne, J.C., Barahona, M.: Laplacian dynamics and multiscale modular structure in networks (2008)
24. Leskovec, J., Rajaraman, A., Ullman, J.D.: Mining of massive datasets. Cambridge university press (2014)
25. Liben-Nowell, D., Kleinberg, J.: The link-prediction problem for social networks. J. of the Am. society for information science and technology **58**(7), 1019–1031 (2007)
26. Lim, H., Gray, P., Xie, L., Poleksic, A.: Improved genome-scale multi-target virtual screening via a novel collaborative filtering approach to cold-start problem. Scientific reports **6**, 38860 (2016)
27. Liu, Z., He, J.L., Kapoor, K., Srivastava, J.: Correlations between community structure and link formation in complex networks. PloS one **8**(9), e72908 (2013)
28. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. Phys. Review E **69**(2), 026113 (2004)
29. Soundarajan, S., Hopcroft, J.: Using community information to improve the precision of link prediction methods. In: WWW. pp. 607–608. ACM (2012)
30. Spielman, D.A., Teng, S.H.: Spectral partitioning works: Planar graphs and finite element meshes. Linear Algebra and its Applications **421**(2-3), 284–305 (2007)
31. Sun, J., Qu, H., Chakrabarti, D., Faloutsos, C.: Neighborhood formation and anomaly detection in bipartite graphs. In: ICDM. pp. 8–pp. IEEE (2005)
32. Tagarelli, A., Amelio, A., Gullo, F.: Ensemble-based community detection in multilayer networks. Data Mining and Knowledge Discovery **31**(5), 1506–1543 (2017)
33. Valverde-Rebaza, J.C., de Andrade Lopes, A.: Link prediction in online social networks using group information. In: ICCSA. pp. 31–45. Springer (2014)
34. Xie, Z., Dong, E., Li, J., Kong, D., Wu, N.: Potential links by neighbor communities. Physica A: Statistical Mechanics and its Applications **406**, 244–252 (2014)
35. Yamanishi, Y., Araki, M., Gutteridge, A., Honda, W., Kanehisa, M.: Prediction of drug–target interaction networks from the integration of chemical and genomic spaces. Bioinformatics **24**(13), i232–i240 (2008)
36. Yan, B., Gregory, S.: Finding missing edges in networks based on their community structure. Phys. Review E **85**(5), 056112 (2012)
37. Zheng, X., Ding, H., Mamitsuka, H., Zhu, S.: Collaborative matrix factorization with multiple similarities for predicting drug-target interactions. In: KDD. pp. 1025–1033. ACM (2013)