Adaptive network automata modelling of complex networks

Alessandro Muscoloni[1,2], Umberto Michieli[2,3] and Carlo Vittorio Cannistraci[1,2,*]

[1]Center for Complex Network Intelligence (CCNI) at Tsinghua Laboratory of Brain and Intelligence (THBI), Department of Bioengineering, Tsinghua University, 160 Chengfu Rd., SanCaiTang Building, Haidian District, 100084, Beijing, China.
[2]Biomedical Cybernetics Group, Biotechnology Center (BIOTEC), Center for Molecular and Cellular Bioengineering (CMCB), Center for Systems Biology Dresden (CSBD), Cluster of Excellence Physics of Life (PoL), Department of Physics, Technische Universität Dresden, Tatzberg 47/49, 01307 Dresden, Germany.
[3]Department of Information Engineering, University of Padova, Padova, Italy.

*Corresponding author: Carlo Vittorio Cannistraci (kalokagathos.agon@gmail.com)

Keywords: complex networks, network models, link prediction, automata theory, network automata, Cannistraci-Hebb theory

Abstract

Many complex networks have a connectivity that might be only partially detected or that tends to grow over time, hence the prediction of non-observed links is a fundamental problem in network science. The aim of topological link prediction is to forecast these non-observed links by only exploiting features intrinsic to the network topology. It has a wide range of real applications, like suggesting friendships in social networks or predicting interactions in biological networks.
The Cannistraci-Hebb theory is a recent achievement in network science that includes a theoretical framework to understand local-based link prediction on paths of length n. In this study we introduce two innovations: theory of modelling (science) and theory of realization (engineering). For the theory of modelling we first recall a definition of network automata as a general framework for modelling the growth of connectivity in complex networks. We then show that several deterministic models previously developed fall within this framework and we introduce novel network automata following the Cannistraci-Hebb rule. For the theory of realization, we present how to build adaptive network automata for link prediction, which incorporate multiple deterministic models of self-organization and automatically choose the rule that better explains the patterns of connectivity in the network under investigation.
We compare Cannistraci-Hebb adaptive (CHA) network automaton against state-of-the-art link prediction methods such as structural perturbation method (SPM), stochastic block models (SBM) and artificial intelligence algorithms for graph embedding. CHA displays an overall higher link prediction performance across different evaluation frameworks on 1386 networks. Finally, we highlight that CHA offers the key advantage to explicitly explain the mechanistic rule of self-organization which leads to the link prediction performance, whereas SPM and graph embedding not. In comparison to CHA, SBM unfortunately shows irrelevant and unsatisfactory performance demonstrating that SBM modelling is not adequate for link prediction in real networks.

1. Introduction

Many complex networks have a connectivity that might be only partially detected or that tends to grow over time, hence the prediction of non-observed links is a fundamental problem in network science. The aim of topological link prediction is to forecast these non-observed links by only exploiting features intrinsic to the network topology. It has a wide range of real applications, like suggesting friendships in social networks or predicting interactions in biological networks [1]–[3]. A plethora of methods based on different methodological principles have been developed in recent years, and in this study we will consider for reference the state-of-the-art algorithms. The first is Structural Perturbation Method (SPM), a model-free global approach that relies on a theory derived from the first-order perturbation in quantum mechanics [4]. The second represents a class of generative models named Stochastic Block Models (SBM), whose general idea is that the nodes are partitioned into groups and the probability that two nodes are connected depends on the groups to which they belong [5]. The third class of methods is model-free and includes machine learning algorithms for graph embedding (GE). Such methods convert the graph data into a low dimensional space in which certain graph structural information and properties are preserved. Different graph embedding variants have been developed aiming to preserve different information in the embedded space, some examples are: HOPE [6], node2vec [7], NetSMF [8], ProNE [9].
While the aforementioned approaches have been shown to be competitive link predictors, they do not offer a clear interpretability of the mechanisms behind the network growth, except SBM that is model-based. This property, instead, can be fulfilled for example by mechanistic models based on the Cannistraci-Hebb theory [3], [10]–[14], since each of them is based on an explicit deterministic mathematical formulation. Each model in the Cannistraci-Hebb theory represents a specific rule of self-organization which is associated to explicit principles that drive the growth's dynamics of the underlying complex networked physical system.
The Cannistraci-Hebb theory is a recent achievement in network science [14], [15] that includes a theoretical framework to understand local-based link prediction on paths of length n. It includes any type of classical local link predictor based on paths of length two such as common neighbors (CN) [16], resource allocation (RA) [17], Jaccard [18] and preferential attachment (PA) [16]; and even the recently proposed link predictor on paths of length three of Kovács et al. [19], which triggered a fundamental discovery on the organization of protein interaction networks (PPI). Following the discovery of a previous article of Daminelli et al. [10] that stressed the importance of paths of length three for link prediction in bipartite

networks [10], on the same line Kovács et al. suggested that proteins interact according to an underlying bipartite scheme. Indeed, proteins interact not if they are similar to each other, but if one of them is similar to the other's partners. This principle, such as the one proposed by Daminelli et al. [10], mathematically relies on network paths of length three (L3) [19], whereas most of the deterministic local based models previously developed were based on paths of length two (L2) [3]. These findings lead to a change of perspective in the field, highlighting the existence of different classes of networks whose patterns of interactions are organized either as L2 or L3. However, a conceptual limitation of the studies of Daminelli et al. [10] and Kovács et al. is that the L3-based link predictors developed [19] were not properly connected to already known principles of modelling, which prompted us to formulate and introduce a generalized theory.

In this study we seek to bring innovations on two sides: theory of modelling (science) and theory of realization (engineering). For the theory of modelling (section 2.1), we first recall a definition of network automata as a general framework for modelling the growth of connectivity in complex networks. We then show that several deterministic models previously developed fall within this framework and we introduce novel network automata following the Cannistraci-Hebb rule. For the theory of realization (section 2.2), we present how to build adaptive network automata for link prediction, which incorporate multiple deterministic models of self-organization and automatically choose the rule that better explains the patterns of connectivity in the network under investigation. Finally (section 2.3), we compare our proposed Cannistraci-Hebb adaptive (CHA) network automaton with SPM, stochastic block models and artificial intelligence algorithms for embedding of graph data, in order to understand what type of link predictor is overall best performing across several evaluation frameworks.

2. Results

2.1. Science of physical modelling

2.1.1. Network automata

The concept of network automata has been originally introduced by Wolfram [20] and later formally defined by Smith et al. [21] as a general framework of network growth. A network automaton represents a network whose links update their state over time and the ruleset governing the network evolution is only dependent on quantities that can be computed from the current topology. In other words, the evolution of the network from time step t to the next step t+1 can be expressed in terms of some operation F acting upon the adjacency matrix X:

$$X(t+1) = F(X(t))$$

The ruleset could be related to any property of the nodes or links and might be deterministic or stochastic. In contrast to cellular automata on a network [20], [22], in which the states of nodes evolve and whose neighborhoods are defined by the network, in the network automata the states of links evolve, therefore the topology itself changes over time.

Smith et al. [21] provide an example in which the state update of a link $X_{u,v}(t+1)$ is determined by a simple topological property such as the sum of the node degrees $f_{u,v}(t) = d_u(t) + d_v(t)$. If the link is existing $X_{u,v}(t) = 1$ and the property is larger than a survival threshold $f_{u,v}(t) > x_S$, then the link survives, otherwise not. If the link is not existing $X_{u,v}(t) = 0$ and the property is larger than a birth threshold $f_{u,v}(t) > x_B$, then the link is born, otherwise not. In this example, the computation of the topological property and the link update based on the survival and birth thresholds constitute the F operation. This basic ruleset exhaustively describes the evolution of a network automata.

We let notice that, if we focus on the topological property $f_{u,v}(t)$, we could replace the sum of the node degrees with any of the several mathematical models already developed for link prediction, such as common neighbors (CN) [16], resource allocation (RA) [17], Jaccard [18] and preferential attachment (PA) [16]. Indeed, while they are often indicated as heuristics, the previous definition and the example provided clearly highlight that such local and deterministic models actually represent network automata.

As final remark, in this link prediction study we specifically use the algorithms only to predict the non-observed links that are more likely to be born at the next step of the network evolution, therefore we will not further consider and discuss the survival and birth thresholds, but only focus on the topological property $f_{u,v}(t)$. We will also omit the time variable t for simplicity of notation.

2.1.2. Network automata on paths of length n

After having recalled the framework of network automata defined by Smith et al. [21], here we introduce a particular subclass named network automata on paths of length *n*. These automata evaluate the topological property between two nodes based on the topological information contained along the paths of length *n* between them. In mathematical terms, we can express the topological property in the form:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 December 2020 doi:10.20944/preprints202012.0808.v1

$$f(u,v) = \sum_{z_1 \ldots z_{n-1} \in Ln} f'(z_1 \ldots z_{n-1})$$

where *u* and *v* are the two seed nodes of the candidate interaction; the summation is executed over all the paths of length *n*; $z_1 \ldots z_{n-1}$ are the intermediate nodes on each path of length *n*; and $f'(z_1 \ldots z_{n-1})$ is some function dependent on the intermediate nodes.

A simple example is represented by the resource allocation (RA) model developed by Zhou et al. [17], which is a network automaton on paths of length two (L2), using as function $f'(z)$ the inverse of the degree of the intermediate node (common neighbour in the L2 case). The mathematical formula is as follows:

$$RA\_L2(u,v) = \sum_{z \in L2} \frac{1}{d_z}$$

where the summation is executed over all the paths of length two; *z* is the intermediate node on each path of length two; and d_z is the respective node degree.

In order to generalize to paths of length *n* > 2, we need an operator that merges the individual topological contributions of the intermediate nodes on a path of length *n*. If, without lack of generality, we use as merging operator the geometrical mean, we derive the following generalized formula for RA on paths of length *n*:

$$RA\_Ln(u,v) = \sum_{z_1 \ldots z_{n-1} \in Ln} \frac{1}{\left(d_{z_1} * \ldots * d_{z_{n-1}}\right)^{\frac{1}{n-1}}}$$

where the summation is executed over all the paths of length *n*; $z_1 \ldots z_{n-1}$ are the intermediate nodes on each path of length *n*; $d_{z_1} \ldots d_{z_{n-1}}$ are the respective node degrees.

We let notice that for paths of length three (L3), the formula given above becomes equal to the one proposed by Kovács et al. [19], which indeed extends the resource allocation principle to paths of length three, although this was not properly clarified in their study, but was subsequently explained by us in a preliminary study [14] supporting the present one. From here forward we will refer to it with the name of RA-L3.

2.1.3. Cannistraci-Hebb network automata on paths of length n

In this section we are going to introduce a new rule of self-organization that can be modelled using different network automata on paths of length *n*, but let's first recall some theory.

In 1949, Donald Olding Hebb advanced a local learning rule in neuronal networks that can be summarized as follows: neurons that fire together wire together [23]. However, the concept of wiring together was not further specified, and could be interpreted in two different ways. The first interpretation is that the connectivity already present, between neurons that fire together, is reinforced; the second interpretation is the formation of new connectivity between neurons not yet interacting but already integrated in an interacting cohort. In 2013, Cannistraci et al. [3] named the second interpretation of the Hebbian learning as *epitopological learning*, and noticed that such learning could be formalized as a mere problem of topological link prediction in complex networks. The rationale is that, in a network with local-community organization, cohort of neurons tend to be co-activated (fire together) and to learn by forming new connections between them (wire together) because they are topologically isolated in the same local community.

Cannistraci et al. [3] postulated that the identification of epitopological learning in neuronal networks was only a special case, hence they proposed it as a general rule of local learning valid for topological link prediction in any complex network with *local-community-paradigm (LCP)* architecture [3]. Based on this idea, they devised a new class of link predictors that demonstrated, also in following studies of other authors, to outperform state of the art link predictors both in monopartite [3], [24]–[30] and bipartite topologies [10], [11], not only on brain connectomes but also on other complex network classes (such as social, biological, economical, etc.). In addition, a recent study of Narula et al. [31] shows that local-community-paradigm and epitopological learning can enhance our understanding of how local brain connectivity is able to process, learn and memorize chronic pain [31].

The previous conceptual and mathematical formalizations of the LCP theory were immature and put more emphasis on the fact that the information related with the common neighbour nodes should be complemented with the topological information emerging from the interactions between them (*internal local-community-links*, *iLCL*, in Fig. 1). This represents a current limitation of the LCP theory that we want to overcome in this study. Indeed, as recently shown by Cannistraci [12], the local isolation of the common neighbour nodes in every local community is equally important to carve the LCP architecture, and this is guaranteed by the fact that the common neighbours minimize their interactions external to the local community (*external local-community-links*, *eLCL*, in Fig. 1) [12]. This minimization forms a sort of topological energy barrier, which limits the information processing to remain internal to the local community [12].

Here, we introduce the concept of *Cannistraci-Hebb (CH)* rule, which revises the mathematical formalization of the LCP theory and represents any rule that explicitly considers the *minimization of eLCL*. In particular, we name Cannistraci-Hebb network automata on paths of length *n* all the network automata models whose function $f'(z_1 \ldots z_{n-1})$ follows the CH rule.

The first CH model we present is a model already introduced by Cannistraci et al. [3] and previously named Cannistraci-Resource-Allocation (CRA) model, which here we rename as CH1. Its mathematical formula in L2 is:

$$CH1\_L2(u,v) = \sum_{z \in L2} \frac{di_z}{d_z}$$

where di_z is the internal node degree (number of iLCL) of the intermediate node. Since the degree of the intermediate node can be decomposed as $d_z = 2 + iLCL + eLCL$, there is an explicit minimization of eLCL, which makes the CRA a CH model. However, the mathematical formulation of CH1 is not so clean, because the minimization of eLCL is conditioned by the presence of iLCL ($di_z > 0$), indeed if $di_z = 0$ then the intermediate node does not give any contribution to the summation. For this reason, we introduced the CH2 model, whose mathematical formula in L2 is:

$$CH2\_L2(u,v) = \sum_{z \in L2} \left(\frac{di_z^*}{de_z^*} = \frac{1 + di_z}{1 + de_z}\right)$$

where di_z and de_z are the internal and external node degrees of the intermediate node (respectively the number of iLCL and eLCL). Note that a unitary term is added to the numerator and denominator to avoid the saturation of the value in case of iLCL or eLCL equal to zero. Finally, we propose a third model (CH3) that exclusively implements the CH rule, solely based on the minimization of eLCL. The mathematical formula in L2 is:

$$CH3\_L2(u,v) = \sum_{z \in L2} \left(\frac{1}{de_z^*} = \frac{1}{1 + de_z}\right)$$

We let notice that the RA model is also a CH network automaton, whereas for example the CN model is not. In the rest of the study we will focus on these four CH network automata: RA, CH1, CH2 and CH3. We will not consider non-CH network automata, such as the several ones already considered by Kovács et al. [32], since they were already shown to be less performing. Analogously to RA, also the other CH network automata can be generalized to paths of *Ln*. The mathematical formula of the four CH models on paths of L2, L3 and Ln are summarized in Fig. 1. In the generalized case *Ln*, the local community is the set of all intermediate nodes involved in any path of length *n* between the two seed nodes. The iLCL are the links between two nodes in the local community, while the eLCL are the links between a node in the local community and a node external to the local community (excluding the seed nodes).

2.2. Engineering the adaptive network automata machine

It has been shown that different complex networks can be organized according to different patterns of connectivity, such as L2 or L3, and therefore there is not a unique network automaton model that would be able to effectively describe all of them. For example, while performing link prediction on a social network one might decide to adopt a L2-based method, whereas on a PPI network it would likely be better to choose a L3-based method. However, here we want to make a step forward from the engineering point of view and design a computational machine that is adaptive to the network under investigation and would automatically select the model that would likely provide the best prediction.

In order to do this, we exploit a particular property of the network automata models discussed in the previous section. Such deterministic rules for link prediction can assign both to observed and non-observed links a score that is comparable, meaning that the scores of observed links are not biased to be higher or lower than the scores of non-observed links. This is because the mathematical equation to compute the score of the connection between two nodes is independent from the existence of that link, whether the link is observed or not in the current topology does not affect the score.

Given a model, we can assign likelihood scores to both observed and non-observed links and compute the AUPR to evaluate how well the model can discriminate them. The assumption is that, if the model tends to score observed links higher than non-observed links, then it would be more effective in predicting missing or future links. Therefore, the adaptive network automaton works as follows: given a network and a set of candidate models, for each model we compute the AUPR, then we automatically select as link prediction result the scores of the non-observed links from the model that obtained the highest AUPR. In the next section we will discuss the computational experiments that highlighted what is the best set of candidate network automata models to adopt for building the adaptive version.

2.3. Computational results on prediction of connectivity in real networks

In order to perform a wide investigation, we collected an ATLAS of 1371 real networks of size up to 20.000 nodes and we categorized them into 14 classes. As first analysis, we performed link prediction using a 10% link removal evaluation (see Methods section 4.2.1. for details) and applying as methods the CH models (RA, CH1, CH2, CH3) on path lengths L2 and L3. In Fig. 2A we report for each network class the win rate of L2 versus L3 models, highlighting that the prediction of connectivity in most classes is dominated by either the L2 or the L3 rule.

For completeness, we also verified that considering longer paths (L4, L5 and L6) does not improve the link prediction performance with respect to simply using L2 or L3 (see Suppl. Table 1). In addition, computing longer paths increases the time complexity of the algorithm, therefore, at least for the networks here considered, it seems not worth to develop CH adaptive (CHA) network automata based on paths longer than L3.

The following analysis is instead focused to understand what is the best combination of CH models to include in the adaptive variant. By testing all the possible CH model combinations, the results of link prediction over the whole ATLAS suggest that using the combination of CH2-CH3 models is the best choice (see Suppl. Table 2). Such models are also the best performing when tested individually for each network class (see Suppl. Table 3).

The CHA network automaton that we propose in this study is therefore based on the CH models CH2-CH3 and on the path lengths L2-L3. Fig. 2B shows the comparison of link prediction win rate over the whole ATLAS for the CHA method versus the single CH models incorporated in CHA. The results clearly highlight the advantage of using an adaptive strategy, with a win rate of almost 0.80 for CHA versus a win rate of around 0.50 for CH2-L2 as best single model.

In addition, the next relevant achievement is that the link prediction win rate of CHA is considerably higher than state-of-the-art methods such as SPM and HOPE (Fig. 2C and Suppl. Table 4), as well as than other embedding-based methods (Suppl. Table 5). We have also compared CHA against several SBM variants on 900 networks of lower size (up to 100 nodes, due to the high computational time requirements of SBM), confirming the outperformance of the adaptive automaton (Fig. 2D and Suppl. Table 6).

As final analysis, we performed temporal link prediction (see Methods section 4.2.2. for details) using CHA, SPM and the embedding-based methods on 15 networks with temporal information. The results presented in Suppl. Table 7 highlight the higher performance of CHA also in this different evaluation framework, with 0.80 win rate versus 0.60 of SPM as second best method.

3. Discussion

In this study, after recalling a definition of network automata as a general framework for modelling the growth of connectivity in complex networks, we have introduced a particular subclass named network automata on paths of length *n*. Within this subclass, we have defined a set of network automata following the Cannistraci-Hebb (CH) rule, which are deterministic models explicitly considering the minimization of links external to a local community for prediction of connectivity. As engineering contribution, we presented how to build adaptive network automata for link prediction, which incorporate multiple deterministic models of self-organization and automatically choose the rule that better explains the patterns of connectivity in the network under investigation. In particular, we have shown that the combination of models CH2-CH3 and of path lengths L2-L3 represents the optimal choice for a Cannistraci-Hebb adaptive (CHA) network automaton on the tested networks. Then, we compared our proposed CHA against state-of-the-art link prediction methods such as SPM, SBM and artificial intelligence algorithms for embedding of graph data, highlighting an overall higher link prediction performance across different evaluation frameworks. Finally, we stress that CHA offers the key advantage to explicitly explain the mechanistic rule of self-organization which leads to the link prediction performance, whereas SPM and graph embedding not. In comparison to CHA, SBM unfortunately shows irrelevant and unsatisfactory performance demonstrating that SBM modelling is not adequate for link prediction in real networks.

4. Methods

4.1. Link prediction methods

4.1.1. Structural Perturbation Method (SPM)

The structural perturbation method (SPM) relies on a theory similar to the first-order perturbation in quantum mechanics [4]. A high-level description of the procedure is the following: (1) randomly remove 10% of the links from the network adjacency matrix X, obtaining a reduced network X’ = X - R, where R is the set of removed links; (2) compute the eigenvalues and eigenvectors of X’; (3) considering the set of links R as a perturbation of X’, construct the perturbed matrix X^P via a first-order approximation that allows the eigenvalues to change while keeping fixed the eigenvectors; (4) repeat steps 1-3 for 10 independent iterations and take the average of the perturbed matrices X^P. The link prediction result is given by the values of the average perturbed matrix, which represent the scores for the non-observed links. The higher the score the greater the likelihood that the interaction exists.

The idea behind the method is that a missing part of the network is predictable if it does not significantly change the structural features of the observable part, represented by the eigenvectors of the matrix. If this is the case, the perturbed matrices should be good approximations of the original network [4].

The MATLAB implementation has been provided by the authors of the study [4].

4.1.2. Stochastic Block Model (SBM)

The general idea of stochastic block model (SBM) is that the nodes are partitioned into B blocks and a B x B matrix specifies the probabilities of links existing between nodes of each block. SBM provides a general framework for statistical analysis and inference in networks, in particular for community detection and link prediction [33]. The concept of degree-corrected (DC) SBM has been introduced for community detection tasks in [34] and for prediction of spurious and missing links in [35], in order to keep into account the variations in node degree typically observed in real networks. The nested (N) version of SBM has been introduced [5] in order to overcome two major limitations: the inability to separate true structures from noise and to detect smaller but well-defined clusters as network size becomes large.

All the four variants tested (SBM, SBM-DC, SBM-N, SBM-DC-N) require to find a proper partitioning of the network in order to make inference. We considered the implementation available in Graph-tool [36], that adopts an optimized Monte Carlo Markov Chain (MCMC) to sample the space of the possible partitions [33]. Graph-tool [36] is a Python module that can be downloaded on the website: http://graph-tool.skewed.de/. As suggested in [37], in general the predictive performance is higher when averaging over collections of partitions than when considering only the single most plausible partition, since this can lead to overfitting. Therefore for a given network we sampled P partitions, for each partition we obtained the likelihood scores related to the non-observed links, and then considered the average likelihood scores as the link prediction result. We set $P = 100$ for the ATLAS networks with $N \leq 100$, and $P = 50$ for the connectomes with $N > 100$.

4.1.3. HOPE

High-Order Proximity preserved Embedding (HOPE) is a graph embedding algorithm aiming to preserve the high-order proximities of graphs and capturing the asymmetric transitivity [6]. Asymmetric transitivity depicts the correlation among directed edges, therefore HOPE is particularly useful for embedding of directed networks, however can be analogously adopted for undirected networks. Many high-order proximity measurements can reflect the asymmetric transitivity in graphs, among which the Katz index [38], and many of them share a general formulation. Instead of computing the proximity matrix and performing singular value decomposition (SVD) on that, HOPE exploits such general formulation to transform the original SVD problem into a generalized SVD problem that can learn directly the embedding vectors, avoiding the calculation of the proximity matrix [6]. However, for the link prediction purpose, an approximation of the proximity matrix can be reconstructed from the embedding. In summary, in our scenario, HOPE provides a scalable solution for an approximation of the Katz index through graph embedding. The entries of the approximated Katz proximity matrix represent the link prediction result. The higher the proximity the greater the likelihood that the interaction exists.

The MATLAB implementation is available at: https://github.com/ZW-ZHANG/HOPE. We set as dimensions of embedding the minimum between 128 and N (number of nodes), and used the default values for the other parameters.

4.1.4. node2vec

node2vec is a graph embedding algorithm that maps nodes to a low-dimensional feature space maximizing the likelihood of preserving network neighborhoods of nodes [7]. The maximization is performed on a custom graph-based objective function using stochastic gradient descent, motivated by prior work on natural language processing and related to the Skip-gram model [7]. Considering a flexible definition of a neighborhood, the algorithm exploits a 2nd-order random walk approach to sample network neighborhoods for nodes. The random walk is dependent on two parameters p (return parameter) and q (in-out parameter) that bias the walk towards different network exploration strategies [7].

After embedding the nodes, we obtain for each node pair a feature vector using the Hadamard (element-wise) product of the feature vectors of the two nodes, as suggested in the node2vec study [7]. Then, the node pairs features are used to train a logistic regression classifier in order to obtain likelihood scores for the non-observed links of the network (see Methods section 4.1.7. for details).

The implementation of the node2vec embedding method is available on GitHub: https://github.com/snap-stanford/snap/. We set as dimensions of embedding the minimum between 128 and N-1 and we discarded node features having the same value for all the nodes. We tested the parameters p and q using three configurations (p = 0.5, q = 2; p = 1, q = 1; p = 2, q = 0.5) and we chose the best one using cross-validation (see Methods section 4.1.7. for details). We used the default values for the other parameters.

4.1.5. ProNE and ProNE-SMF

ProNE has been proposed as a fast and scalable graph embedding algorithm that maps nodes to a low-dimensional feature space using a two-steps procedure [9]. The first step consists in initializing the network embedding using sparse matrix factorization to efficiently obtain an initial node representation, which is achieved by randomized truncated singular value decomposition. The second step is inspired by the higher-order Cheeger's inequality and consists in performing spectral propagation in order to enhance the initial embedding [9]. In our analysis we considered both the embeddings obtained after the first step (ProNE-SMF) and after the second step (ProNE).

After embedding the nodes, we obtain for each node pair a feature vector using the Hadamard (element-wise) product of the feature vectors of the two nodes. Then, the node pairs features are used to train a logistic regression classifier in order to obtain likelihood scores for the non-observed links of the network (see Methods section 4.1.7. for details).

The implementation of the ProNE and ProNE-SMF embedding methods is available on GitHub: https://github.com/THUDM/ProNE/. We set as dimensions of embedding the minimum between 128 and N-1 and we discarded node features having the same value for all the nodes. We used the default values for the parameters.

4.1.6. NetSMF

NetSMF is a graph embedding algorithm that maps nodes to a low-dimensional feature space [8]. It is based on previous results according to which several network embedding algorithms implicitly factorize a certain closed-form matrix, and that explicitly factorizing the matrix leads to higher performance. However, such matrix is dense and therefore computationally expensive to handle for large networks. NetSMF proposes a scalable solution that firstly leverages spectral graph sparsification techniques to build a sparse matrix spectrally close to the original dense one, and then performs randomized singular value decomposition to efficiently factorize the sparse matrix, obtaining the embedding [8].

After embedding the nodes, we obtain for each node pair a feature vector using the Hadamard (element-wise) product of the feature vectors of the two nodes. Then, the node pairs features are used to train a logistic regression classifier in order to obtain likelihood scores for the non-observed links of the network (see Methods section 4.1.7. for details).

The implementation of the ProNE and ProNE-SMF embedding methods is available on GitHub: https://github.com/xptree/NetSMF/. We set as dimensions of embedding the minimum between 128 and N-1 and we discarded node features having the same value for all the nodes. We set rounds = 10000 and used the default values for the other parameters.

4.1.7. Logistic regression classifier

After obtaining feature vectors for each node pair from the network embedding of node2vec, ProNE, ProNE-SMF and NetSMF, we trained a logistic regression classifier in order to obtain likelihood scores for the non-observed links of the network.

In particular, we repeated 10 times a 5-fold cross-validation. For each repetition $i \in [1, 10]$, we performed the following steps: (1) create a learning set with all the observed links and an equal amount of non-observed links (if available, otherwise all of them); (2) split the learning set for a 5-fold cross-validation; (3) for each cross-validation iteration $j \in [1, 5]$: (3.1) train a logistic regression classifier using 4 folds and get coefficient estimates $B_{i,j}$; (3.2) validation: using the coefficients $B_{i,j}$, get from the classifier likelihood scores for the remaining fold and evaluate the prediction computing $\text{AUPR}_{i,j}$.

After the 10 repetitions, we compute the mean coefficients estimates $\bar{B}$ over the 10 repetitions and the 5 cross-validation iterations. Using the coefficients $\bar{B}$, we get from the classifier likelihood scores for the non-observed links of the network, which represent the link prediction result.

doi:10.20944/preprints202012.0808.v1

In the case of node2vec, for which multiple parameter configurations are tested, we compute the mean $\overline{\text{AUPR}}$ of the validation phase over the 10 repetitions and 5 cross-validation iterations, and we chose as link prediction result the one related to the parameter configuration that obtains the highest $\overline{\text{AUPR}}$. In the case of ProNE, ProNE-SMF and NetSMF, for which only one parameter configuration is tested, the validation step (3.2) is not required. We used the MATLAB implementation of the logistic regression classifier (functions *mnrfit* and *mnrval*).

4.2. Link prediction evaluation

4.2.1. 10% link removal evaluation

The 10% link removal evaluation framework is adopted when there is no information available about missing links or links that will appear in the future with respect to the time point of the network under consideration.

Given a network X, 10% of links are randomly removed, obtaining a reduced network X' = X - R, where R is the set of removed links. For a certain algorithm to be evaluated, the reduced network X' is given in input, obtaining in output likelihood scores for the non-observed links in X'. The non-observed links are ranked by decreasing likelihood scores and the area under precision-recall curve (AUPR) is computed, considering as positive samples the set R of links previously removed. Due to the randomness of the link removal, the evaluation is repeated 10 times and the mean AUPR indicates the performance of the algorithm on the network X.

4.2.2. Temporal evaluation

The temporal evaluation framework is adopted when there is information available about links that will appear in the future with respect to the time point of the network under consideration. For a given network, a certain number T of snapshots are available, corresponding to different time points of the network. Each snapshot at times $i \in [1, T\text{-}1]$ is given in input to a certain algorithm to be evaluated, obtaining in output likelihood scores for the non-observed links at time *i*. For each pair of time points (i, j), with $i \in [1, T\text{-}1]$ and $j \in [i\text{+}1, T]$, the non-observed links at time *i* are ranked by decreasing likelihood scores and the area under precision-recall curve (AUPR) is computed, considering as positive samples the non-observed links at time *i* that appear at time *j*. Non-observed links at time *i* involving nodes that disappear at time *j* are removed from the ranking. The mean AUPR over all the pairs of time points (i, j) indicates the performance of the algorithm on the network.

4.3. Datasets

4.3.1. ATLAS

We have collected a dataset of 1371 real networks, either downloaded from publicly available online sources or provided by authors of previous scientific studies. The networks have been categorized into 14 classes (respective number of networks in brackets): collaboration (18), contact (32), covert (86), friendship (16), PPI (19), connectome (529), foodweb (71), trade (200), transcription (8), coauthorship (24), flightmap (36), internet (215), socialnetwork (108), software (9).

For a complete list of the networks, basic properties (number of nodes and edges), references, sources and descriptions, please refer to Supplementary Information File 2.

4.3.2. Temporal networks

We have collected a dataset of 15 real networks with temporal information, downloaded from publicly available online sources. For each network, a certain number of snapshots are available, corresponding to different time points.

For a complete list of the networks, basic properties (number of nodes and edges), references, sources and descriptions, please refer to Supplementary Information File 2.

Hardware and software

The sources of the software for the algorithms used is indicated in the respective Methods sections. Where not indicated, MATLAB code has been implemented. The simulations have been carried out partly on a workstation under Windows 8.1 Pro with 512 GB of RAM and 2 Intel(R) Xenon(R) CPU E5-2687W v3 processors with 3.10 GHz, and partly on the ZIH-Cluster Taurus of the TU Dresden.

Competing interests

The authors declare no competing financial interests.

References

[1] L. Lü and T. Zhou, "Link prediction in complex networks: A survey," *Phys. A Stat. Mech. its Appl.*, vol. 390, no. 6, pp. 1150–1170, 2011, doi: 10.1016/j.physa.2010.11.027.

[2] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *J. Am. Soc. Inf. Sci. Technol.*, vol. 58, no. 7, pp. 1019–1031, 2007, doi: 10.1002/asi.20591.

[3] C. V. Cannistraci, G. Alanis-Lobato, and T. Ravasi, "From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks," *Sci. Rep.*, vol. 3, no. 1613, pp. 1–13, 2013, doi: 10.1038/srep01613.

[4] L. Lü, L. Pan, T. Zhou, Y.-C. Zhang, and H. E. Stanley, "Toward link predictability of complex networks," *Proc. Natl. Acad. Sci.*, vol. 112, no. 8, pp. 2325–2330, 2015, doi: 10.1073/pnas.1424644112.

[5] T. P. Peixoto, "Hierarchical block structures and high-resolution model selection in large networks," *Phys. Rev. X*, 2014, doi: 10.1103/PhysRevX.4.011047.

[6] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," 2016, doi: 10.1145/2939672.2939751.

[7] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," 2016, doi: 10.1145/2939672.2939754.

[8] J. Qiu *et al.*, "NetSMF: Large-scale network embedding as sparse matrix factorization," 2019, doi: 10.1145/3308558.3313446.

[9] J. Zhang, Y. Dong, Y. Wang, J. Tang, and M. Ding, "Prone: Fast and scalable network representation learning," 2019, doi: 10.24963/ijcai.2019/594.

[10] S. Daminelli, J. M. Thomas, C. Durán, and C. V. Cannistraci, "Common neighbours and the local-community-paradigm for topological link prediction in bipartite networks," *New J. Phys.*, vol. 17, no. 11, p. 113037, 2015, doi: 10.1088/1367-2630/17/11/113037.

[11] C. Durán, S. Daminelli, J. M. Thomas, V. J. Haupt, M. Schroeder, and C. V. Cannistraci, "Pioneering topological methods for network-based drug–target prediction by exploiting a brain-network self-organization theory," *Brief. Bioinform.*, vol. 8, no. W1, pp. 3–62, 2017, doi: 10.1093/bib/bbx041.

[12] C. V. Cannistraci, "Modelling Self-Organization in Complex Networks Via a Brain-Inspired Network Automata Theory Improves Link Reliability in Protein Interactomes," *Sci. Rep.*, vol. 8, no. 1, p. 15760, 2018, doi: 10.1038/s41598-018-33576-8.

[13] A. Muscoloni and C. V. Cannistraci, "Local-ring network automata and the impact of hyperbolic geometry in complex network link-prediction," *arXiv:1707.09496 [physics.soc-ph]*, 2017.

[14] A. Muscoloni, I. Abdelhamid, and C. V. Cannistraci, "Local-community network automata modelling based on length-three-paths for prediction of complex network structures in protein interactomes, food webs and more," *bioRxiv*, 2018.

[15] T. Zhou, Y.-L. Lee, and G. Wang, "Experimental analyses on 2-hop-based and 3-hop-based link prediction algorithms," *Phys. A Stat. Mech. its Appl.*, vol. 564, p. 125532, 2021, doi: https://doi.org/10.1016/j.physa.2020.125532.

[16] M. E. J. Newman, "Clustering and preferential attachment in growing networks," vol. 64, pp. 1–4, 2001, doi: 10.1103/PhysRevE.64.025102.

[17] T. Zhou, L. Lu, and Y. C. Zhang, "Predicting missing links via local information," *Eur. Phys. J. B*, vol. 71, no. 4, pp. 623–630, 2009, doi: 10.1140/epjb/e2009-00335-8.

[18] P. Jaccard, "Distribution comparée de la flore alpine dans quelques régions des Alpes occidentales et orientales," *Bull. la Murithienne*, 1902.

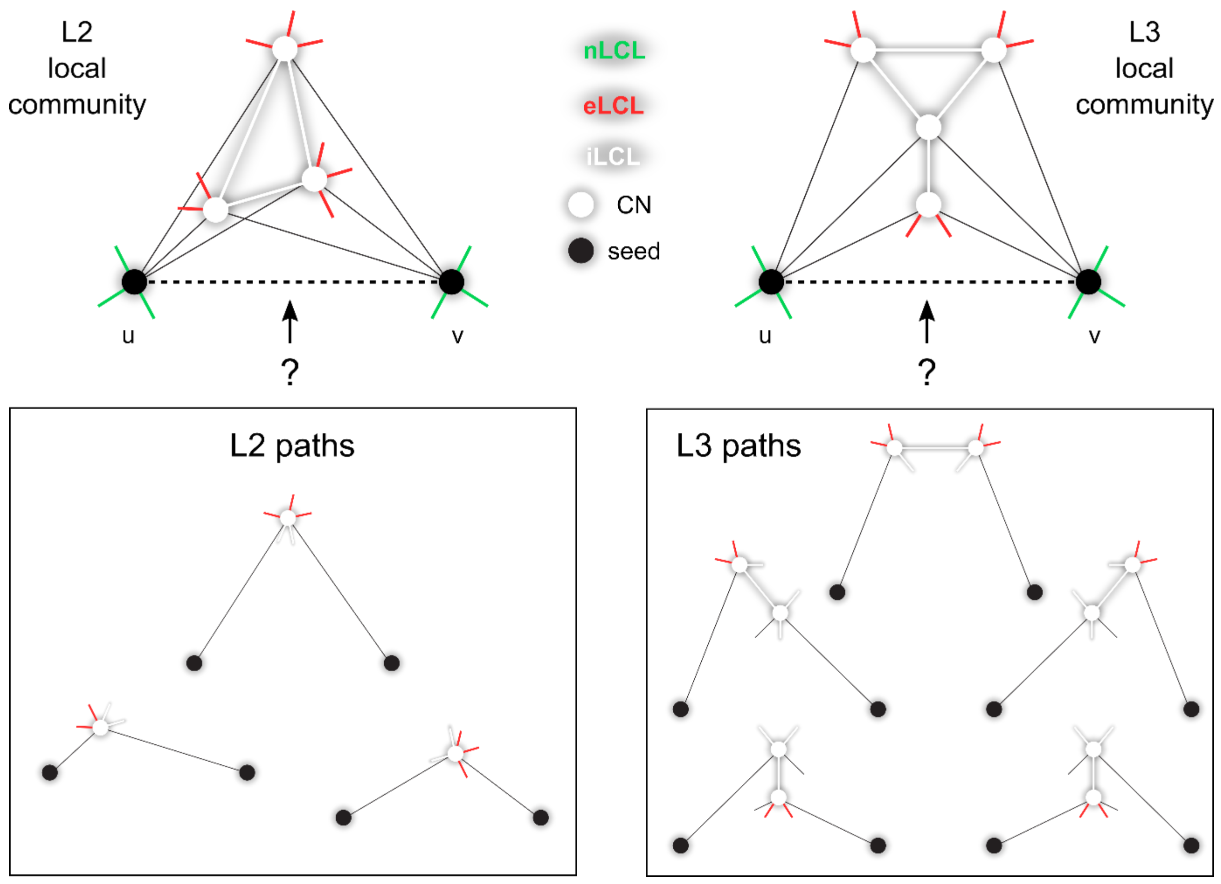
[19] I. A. Kovács *et al.*, "Network-based prediction of protein interactions," *Nat. Commun.*, 2019, doi: 10.1038/s41467-019-09177-y.

[20] S. Wolfram, *A New Kind of Science*. Wolfram Media, 2002.

[21] D. M. D. Smith, J.-P. Onnela, C. F. A. N. Lee, M. D. Fricker, and N. F. Johnson, "Network Automata: coupling structure and function in dynamic networks," *Adv. Complex Syst.*, vol. 14, no. 03, pp. 317–339, 2011, doi: 10.1142/S0219525911003050.

[22] C. Marr and M. T. Hütt, "Topology regulates pattern formation capacity of binary cellular automata on graphs," *Phys. A Stat. Mech. its Appl.*, 2005, doi: 10.1016/j.physa.2005.02.019.

[23] D. O. Hebb, *The Organization of Behavior*, vol. 911, no. 1. 1949.
[24] Z. Liu, J. L. He, K. Kapoor, and J. Srivastava, "Correlations between Community Structure and Link Formation in Complex Networks," *PLoS One*, vol. 8, no. 9, 2013, doi: 10.1371/journal.pone.0072908.
[25] L. Pan, T. Zhou, L. Lü, and C.-K. Hu, "Predicting missing links and identifying spurious links via likelihood analysis," *Sci. Rep.*, vol. 6, pp. 1–10, 2016, doi: 10.1038/srep22955.
[26] F. Tan, Y. Xia, and B. Zhu, "Link prediction in complex networks: A mutual information perspective," *PLoS One*, vol. 9, no. 9, 2014, doi: 10.1371/journal.pone.0107056.
[27] W. Wang, F. Cai, P. Jiao, and L. Pan, "A perturbation-based framework for link prediction via non-negative matrix factorization," *Sci. Rep.*, vol. 6, no. December, p. 38938, 2016, doi: 10.1038/srep38938.
[28] T. Wang, H. Wang, and X. Wang, "CD-Based indices for link prediction in complex network," *PLoS One*, vol. 11, no. 1, pp. 5–7, 2016, doi: 10.1371/journal.pone.0146727.
[29] R. Pech, D. Hao, L. Pan, H. Cheng, and T. Zhou, "Link Prediction via Matrix Completion," *EPL*, no. 117, p. 38002, 2017, doi: 10.1209/0295-5075/117/38002.
[30] H. Shakibian and N. M. Charkari, "Mutual information model for link prediction in heterogeneous complex networks," *Sci. Rep.*, vol. 7, no. January, p. 44981, 2017, doi: 10.1038/srep44981.
[31] V. Narula, A. G. Zippo, A. Muscoloni, G. E. M. Biella, and C. V. Cannistraci, "Can local-community-paradigm and epitopological learning enhance our understanding of how local brain connectivity is able to process, learn and memorize chronic pain?," *Appl. Netw. Sci.*, vol. 2, no. 1, p. 28, Aug. 2017, doi: 10.1007/s41109-017-0048-x.
[32] I. A. Kovács *et al.*, "Network-based prediction of protein interactions," *bioRxiv*, Jan. 2018, [Online]. Available: http://biorxiv.org/content/early/2018/03/02/275529.abstract.
[33] T. P. Peixoto, "Efficient Monte Carlo and greedy heuristic for the inference of stochastic block models," *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, vol. 89, no. 1, 2014, doi: 10.1103/PhysRevE.89.012804.
[34] B. Karrer and M. E. J. Newman, "Stochastic blockmodels and community structure in networks," *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, vol. 83, no. 1, 2011, doi: 10.1103/PhysRevE.83.016107.
[35] X. Zhang, X. Wang, C. Zhao, D. Yi, and Z. Xie, "Degree-corrected stochastic block models and reliability in networks," *Phys. A Stat. Mech. its Appl.*, vol. 393, pp. 553–559, 2014, doi: 10.1016/j.physa.2013.08.061.
[36] T. P. Peixoto, "The Graph-tool Python Library," *Figshare*, 2014, doi: 10.6084/m9.figshare.1164194.
[37] T. Vallès-Català, T. P. Peixoto, M. Sales-Pardo, and R. Guimerà, "Consistencies and inconsistencies between model selection and link prediction in networks," *Phys. Rev. E*, 2018, doi: 10.1103/PhysRevE.97.062316.
[38] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, 1953, doi: 10.1007/BF02289026.



L2	L3	Ln
$RA\_L2(u,v) = \sum_{z \in L2} \frac{1}{d_z}$	$RA\_L3(u,v) = \sum_{z_1,z_2 \in L3} \frac{1}{\sqrt{d_{z_1} * d_{z_2}}}$	$RA\_Ln(u,v) = \sum_{z_1 \ldots z_{n-1} \in Ln} \frac{1}{\left(d_{z_1} * \ldots * d_{z_{n-1}}\right)^{\frac{1}{n-1}}}$
$CH1\_L2(u,v) = \sum_{z \in L2} \frac{di_z}{d_z}$	$CH1\_L3(u,v) = \sum_{z_1,z_2 \in L3} \frac{\sqrt{di_{z_1} * di_{z_2}}}{\sqrt{d_{z_1} * d_{z_2}}}$	$CH1\_Ln(u,v) = \sum_{z_1 \ldots z_{n-1} \in Ln} \frac{\left(di_{z_1} * \ldots * di_{z_{n-1}}\right)^{\frac{1}{n-1}}}{\left(d_{z_1} * \ldots * d_{z_{n-1}}\right)^{\frac{1}{n-1}}}$
$CH2\_L2(u,v) = \sum_{z \in L2} \frac{di_z^*}{de_z^*}$	$CH2\_L3(u,v) = \sum_{z_1,z_2 \in L3} \frac{\sqrt{di_{z_1}^* * di_{z_2}^*}}{\sqrt{de_{z_1}^* * de_{z_2}^*}}$	$CH2\_Ln(u,v) = \sum_{z_1 \ldots z_{n-1} \in Ln} \frac{\left(di_{z_1}^* * \ldots * di_{z_{n-1}}^*\right)^{\frac{1}{n-1}}}{\left(de_{z_1}^* * \ldots * de_{z_{n-1}}^*\right)^{\frac{1}{n-1}}}$
$CH3\_L2(u,v) = \sum_{z \in L2} \frac{1}{de_z^*}$	$CH3\_L3(u,v) = \sum_{z_1,z_2 \in L3} \frac{1}{\sqrt{de_{z_1}^* * de_{z_2}^*}}$	$CH3\_Ln(u,v) = \sum_{z_1 \ldots z_{n-1} \in Ln} \frac{\left(di_{z_1}^* * \ldots * di_{z_{n-1}}^*\right)^{\frac{1}{n-1}}}{\left(de_{z_1}^* * \ldots * de_{z_{n-1}}^*\right)^{\frac{1}{n-1}}}$

Figure 1. Cannistraci-Hebb epitopological rationale.
The figure shows an explanatory example for the topological link prediction performed using the L2 or L3 Cannistraci-Hebb epitopological rationale. The two black nodes represent the seed nodes whose non-observed interaction should be scored with a likelihood. The white nodes are the L2 or L3 common-neighbours (CNs) of the seed nodes, further neighbours are not shown for simplicity. The cohort of common-neighbours and the iLCL form the local community. The different types of links are reported with different colours: non-LCL (green), external-LCL (red), internal-LCL (white). The set of L2 and L3 paths related to the given examples of local communities are shown. At the bottom, the mathematical description of the L2, L3 and Ln methods considered in this study are reported. Notation: u, v are the seed nodes; z is the intermediate node (CN) in the L2 path; d_z is the degree of z; di_z is the internal degree (number of iLCL) of z; de_z is the external degree (number of eLCL) of z. For any degree it is valid the following: $d^* = 1 + d$. For L3 and Ln paths the definitions are analogous.

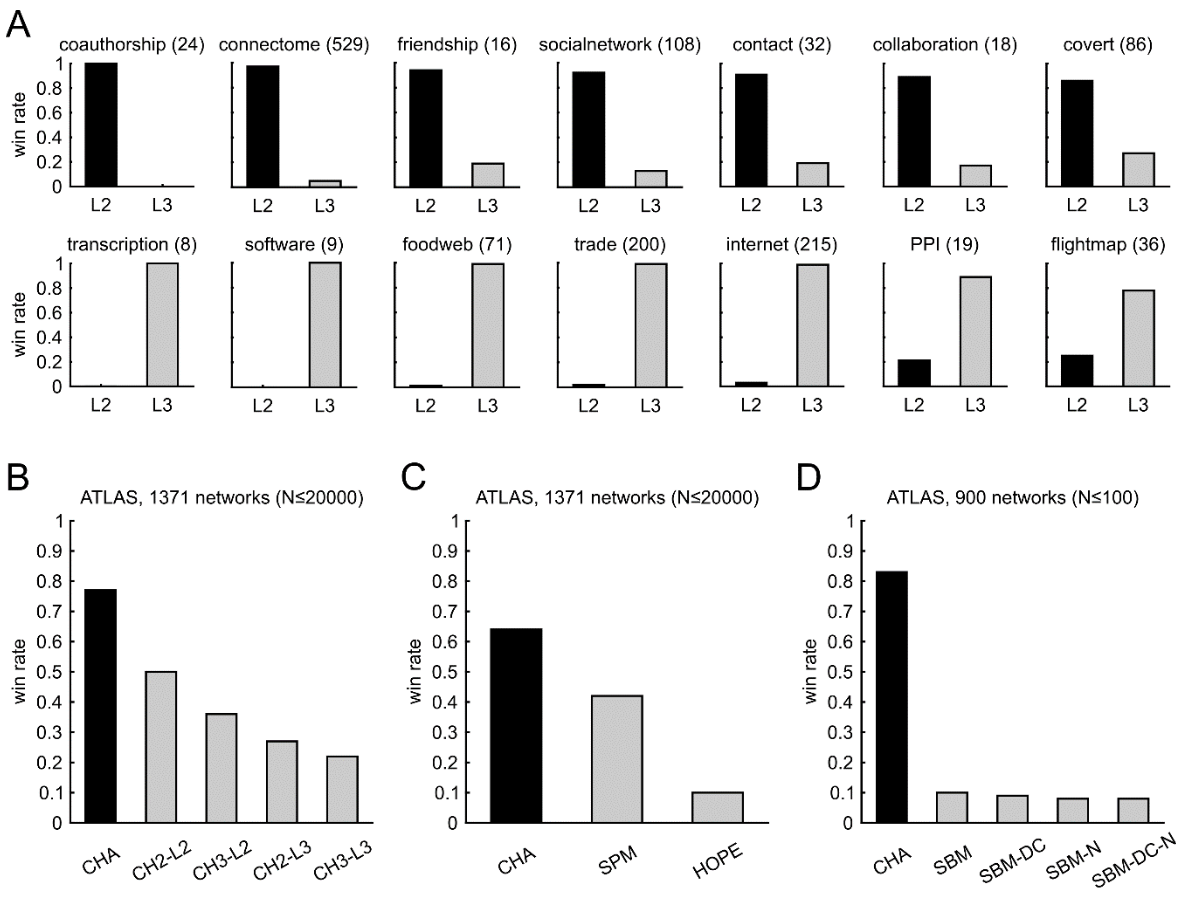


Figure 2. L2-L3 network classes and results on ATLAS.
(A) For each network of the ATLAS (considering N≤20000, 1371 networks) and for each CH model (RA, CH1, CH2, CH3) of path lengths L2-L3, we applied the 10% link removal evaluation, obtaining the AUPR as measure of performance (see Methods section 4.2.1. for details). For each path length L2-L3, we assigned as performance on a network the maximum AUPR over the CH models. The barplots report for each network class and for each path length the win rate over the networks of that class. For each class, the number of networks is shown in brackets. **(B)** For each network of the ATLAS (considering N≤20000, 1371 networks) and for the algorithms CHA, CH2-L2, CH3-L2, CH2-L3 and CH2-L3, we applied the 10% link removal evaluation, obtaining the AUPR as measure of performance. The barplot reports the mean win rate over the network classes. **(C)** The barplot is analogous to (B), for each network of the ATLAS (considering N≤20000, 1371 networks) and for the algorithms CHA, SPM and HOPE. **(D)** The barplot is analogous to (B), for each network of the ATLAS (considering N≤100, 900 networks) and for the algorithms CHA and SBM variants.

	L2	L3	L4	L5	L6
collaboration (17)	0.88 (0.32)	0.18 (0.26)	0.00 (0.19)	0.00 (0.17)	0.00 (0.15)
contact (23)	0.96 (0.32)	0.13 (0.28)	0.04 (0.25)	0.04 (0.23)	0.04 (0.22)
covert (84)	0.86 (0.28)	0.25 (0.20)	0.00 (0.13)	0.00 (0.09)	0.01 (0.07)
friendship (16)	0.94 (0.25)	0.19 (0.21)	0.06 (0.17)	0.06 (0.16)	0.06 (0.15)
PPI (14)	0.07 (0.03)	1.00 (0.09)	0.00 (0.01)	0.00 (0.05)	
connectome (522)	0.98 (0.42)	0.04 (0.27)	0.01 (0.21)	0.01 (0.19)	
foodweb (70)	0.01 (0.06)	0.97 (0.33)	0.00 (0.08)	0.03 (0.19)	
trade (196)	0.01 (0.02)	0.80 (0.14)	0.03 (0.02)	0.29 (0.11)	
transcription (8)	0.00 (0.01)	1.00 (0.11)	0.00 (0.00)	0.00 (0.02)	
coauthorship (18)	1.00 (0.74)	0.00 (0.46)	0.00 (0.26)		
flightmap (35)	0.26 (0.13)	0.74 (0.21)	0.03 (0.10)		
internet (170)	0.04 (0.12)	0.98 (0.22)	0.02 (0.14)		
socialnetwork (19)	0.79 (0.24)	0.26 (0.20)	0.00 (0.12)		
software (7)	0.00 (0.02)	1.00 (0.10)	0.00 (0.02)		

Suppl. Table 1. Results of CH for increasing path lengths on ATLAS.
For each class, we have selected a set of ATLAS networks and a maximum path length that made the computation feasible in a reasonable amount of time. For each class, the number of networks is shown in brackets and the path lengths not evaluated are left empty in the table. As algorithms, we have considered the CH models (RA, CH1, CH2, CH3) of each path length. For each network and for each algorithm, we applied the 10% link removal evaluation, obtaining the AUPR as measure of performance (see Methods section 4.2.1. for details). For each path length, we assigned as performance on a network the maximum AUPR over the CH models. The table reports for each class and for each path length the win rate and the mean AUPR (in brackets) over the networks of that class. For each class, the best win rate is highlighted in red.

	win rate	AUPR
upper bound	1	0.28
CH2-CH3	0.75	0.27
RA-CH2-CH3	0.75	0.27
CH3	0.74	0.27
RA-CH3	0.73	0.27
CH1-CH2-CH3	0.69	0.27
CH1-CH3	0.68	0.27
RA-CH1-CH2-CH3	0.68	0.27
RA-CH1-CH3	0.67	0.27
RA-CH2	0.38	0.27
CH2	0.37	0.26
CH1-CH2	0.34	0.26
RA-CH1-CH2	0.33	0.26
RA	0.25	0.26
RA-CH1	0.22	0.26
CH1	0.16	0.24

Suppl. Table 2. Results of CHA variants on ATLAS.
We considered as algorithms several CHA variants on path lengths L2-L3 using all the possible combinations of CH models. For each network of the ATLAS (considering $N \leq 20000$, 1371 networks) and for each algorithm, we applied the 10% link removal evaluation, obtaining the AUPR as measure of performance (see Methods section 4.2.1. for details). For each network class, we computed the win rate and the mean AUPR over the networks of that class. The table reports for each algorithm the mean win rate and mean AUPR over the classes. The best win rate and AUPR are highlighted in red. The algorithms are sorted by decreasing win rate, their names represent the combinations of CH models used in the adaptive variant. The upper bound indicates the performance that would be reached if the best CH model and path length would be selected for each network.

	CH3-L3	CH3-L2	CH2-L3	CH2-L2	RA-L2	RA-L3	CH1-L2	CH1-L3
collaboration (18)	0.00 (0.26)	0.56 (0.32)	0.06 (0.24)	0.33 (0.32)	0.22 (0.30)	0.11 (0.24)	0.28 (0.30)	0.00 (0.24)
contact (32)	0.13 (0.29)	0.56 (0.34)	0.09 (0.29)	0.25 (0.34)	0.38 (0.34)	0.16 (0.30)	0.25 (0.33)	0.06 (0.29)
covert (86)	0.17 (0.21)	0.37 (0.28)	0.12 (0.20)	0.30 (0.27)	0.42 (0.28)	0.10 (0.21)	0.15 (0.25)	0.09 (0.19)
friendship (16)	0.06 (0.20)	0.56 (0.24)	0.06 (0.19)	0.63 (0.24)	0.50 (0.24)	0.13 (0.20)	0.38 (0.24)	0.06 (0.19)
PPI (19)	0.84 (0.10)	0.00 (0.03)	0.53 (0.09)	0.16 (0.04)	0.00 (0.02)	0.21 (0.09)	0.11 (0.05)	0.05 (0.08)
connectome (529)	0.04 (0.27)	0.94 (0.42)	0.04 (0.26)	0.05 (0.39)	0.05 (0.39)	0.02 (0.26)	0.02 (0.37)	0.02 (0.26)
foodweb (71)	0.80 (0.33)	0.01 (0.04)	0.62 (0.32)	0.00 (0.06)	0.01 (0.04)	0.08 (0.27)	0.00 (0.06)	0.06 (0.28)
trade (200)	0.73 (0.15)	0.02 (0.03)	0.50 (0.14)	0.01 (0.02)	0.01 (0.02)	0.57 (0.14)	0.01 (0.02)	0.42 (0.14)
transcription (8)	1.00 (0.11)	0.00 (0.01)	0.13 (0.10)	0.00 (0.01)	0.00 (0.01)	0.00 (0.09)	0.00 (0.00)	0.13 (0.08)
coauthorship (24)	0.00 (0.44)	0.92 (0.72)	0.00 (0.39)	0.21 (0.69)	0.29 (0.71)	0.00 (0.40)	0.04 (0.62)	0.00 (0.34)
flightmap (36)	0.47 (0.22)	0.08 (0.13)	0.25 (0.21)	0.08 (0.13)	0.08 (0.13)	0.33 (0.21)	0.08 (0.10)	0.25 (0.20)
internet (215)	0.98 (0.20)	0.01 (0.09)	0.28 (0.19)	0.03 (0.11)	0.01 (0.08)	0.09 (0.18)	0.01 (0.10)	0.00 (0.18)
socialnetwork (108)	0.13 (0.10)	0.50 (0.17)	0.05 (0.10)	0.59 (0.18)	0.05 (0.16)	0.03 (0.08)	0.16 (0.17)	0.02 (0.09)
software (9)	1.00 (0.20)	0.00 (0.02)	0.44 (0.20)	0.00 (0.02)	0.00 (0.02)	0.11 (0.18)	0.00 (0.02)	0.11 (0.18)
mean win rate (AUPR)	0.45 (0.22)	0.32 (0.20)	0.23 (0.21)	0.19 (0.20)	0.14 (0.20)	0.14 (0.20)	0.11 (0.19)	0.09 (0.20)

Suppl. Table 3. Results of CH models on ATLAS.
For each network of the ATLAS (considering $N \leq 20000$, 1371 networks) and for each algorithm, we applied the 10% link removal evaluation, obtaining the AUPR as measure of performance (see Methods section 4.2.1. for details). The table reports for each network class and for each algorithm the win rate and the mean AUPR (in brackets) over the networks of that class. For each class, the number of networks is shown in brackets. As bottom row, the table reports the mean win rate and AUPR over the classes. For each class, the best win rate is highlighted in red. The algorithms are sorted left to right according to the best mean win rate.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 December 2020 doi:10.20944/preprints202012.0808.v1

	CHA	SPM	HOPE
collaboration (18)	0.78 (0.33)	0.33 (0.31)	0.11 (0.25)
contact (32)	0.66 (0.35)	0.41 (0.34)	0.28 (0.30)
covert (86)	0.69 (0.27)	0.20 (0.23)	0.31 (0.22)
friendship (16)	0.56 (0.24)	0.56 (0.24)	0.25 (0.22)
PPI (19)	0.79 (0.10)	0.42 (0.09)	0.21 (0.08)
connectome (529)	0.03 (0.42)	0.98 (0.47)	0.02 (0.29)
foodweb (71)	0.45 (0.33)	0.59 (0.43)	0.00 (0.07)
trade (200)	0.75 (0.14)	0.31 (0.11)	0.06 (0.04)
transcription (8)	1.00 (0.11)	0.13 (0.06)	0.00 (0.02)
coauthorship (24)	1.00 (0.72)	0.04 (0.57)	0.00 (0.40)
flightmap (36)	0.72 (0.21)	0.22 (0.16)	0.06 (0.13)
internet (215)	0.54 (0.20)	0.62 (0.22)	0.01 (0.17)
socialnetwork (108)	0.13 (0.19)	0.90 (0.24)	0.03 (0.19)
software (9)	0.89 (0.20)	0.11 (0.16)	0.00 (0.10)
mean win rate (AUPR)	0.64 (0.27)	0.42 (0.26)	0.10 (0.18)

Suppl. Table 4. Results of CHA, SPM and HOPE on ATLAS.
For each network of the ATLAS (considering *N*≤20000, 1371 networks) and for each algorithm, we applied the 10% link removal evaluation, obtaining the AUPR as measure of performance (see Methods section 4.2.1. for details). The table reports for each network class and for each algorithm the win rate and the mean AUPR (in brackets) over the networks of that class. For each class, the number of networks is shown in brackets. As bottom row, the table reports the mean win rate and AUPR over the classes. For each class, the best win rate is highlighted in red. The algorithms are sorted left to right according to the best mean win rate.

	CHA	HOPE	node2vec	ProNE	ProNE-SMF	NetSMF
collaboration (18)	1.00 (0.33)	0.11 (0.25)	0.00 (0.16)	0.00 (0.12)	0.00 (0.12)	0.00 (0.06)
contact (32)	0.88 (0.35)	0.28 (0.30)	0.00 (0.20)	0.00 (0.16)	0.00 (0.16)	0.03 (0.13)
covert (86)	0.74 (0.27)	0.35 (0.22)	0.03 (0.11)	0.01 (0.11)	0.01 (0.08)	0.00 (0.07)
friendship (16)	0.81 (0.24)	0.25 (0.22)	0.00 (0.15)	0.06 (0.13)	0.00 (0.08)	0.00 (0.11)
PPI (10)	1.00 (0.09)	0.20 (0.05)	0.00 (0.00)	0.00 (0.01)	0.00 (0.01)	0.00 (0.00)
connectome (529)	0.98 (0.42)	0.03 (0.29)	0.01 (0.21)	0.00 (0.19)	0.01 (0.19)	0.00 (0.06)
foodweb (71)	1.00 (0.33)	0.00 (0.07)	0.00 (0.09)	0.00 (0.05)	0.01 (0.08)	0.00 (0.04)
trade (200)	0.86 (0.14)	0.04 (0.04)	0.07 (0.05)	0.01 (0.02)	0.13 (0.04)	0.01 (0.01)
transcription (8)	1.00 (0.11)	0.00 (0.02)	0.00 (0.01)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
coauthorship (18)	1.00 (0.74)	0.00 (0.42)	0.00 (0.15)	0.00 (0.18)	0.00 (0.15)	0.00 (0.13)
flightmap (36)	0.86 (0.21)	0.14 (0.13)	0.00 (0.06)	0.00 (0.05)	0.00 (0.06)	0.00 (0.01)
internet (108)	1.00 (0.20)	0.01 (0.15)	0.01 (0.01)	0.01 (0.03)	0.01 (0.07)	0.01 (0.00)
socialnetwork (40)	0.33 (0.21)	0.68 (0.23)	0.00 (0.06)	0.00 (0.12)	0.00 (0.12)	0.00 (0.04)
software (6)	1.00 (0.10)	0.00 (0.05)	0.00 (0.00)	0.00 (0.01)	0.00 (0.01)	0.00 (0.01)
mean win rate (AUPR)	0.89 (0.27)	0.15 (0.17)	0.01 (0.09)	0.01 (0.08)	0.01 (0.08)	0.00 (0.05)

Suppl. Table 5. Results of CHA and embedding methods on ATLAS.
For each network of the ATLAS (considering $N \leq 5000$, 1178 networks) and for each algorithm, we applied the 10% link removal evaluation, obtaining the AUPR as measure of performance (see Methods section 4.2.1. for details). The table reports for each network class and for each algorithm the win rate and the mean AUPR (in brackets) over the networks of that class. For each class, the number of networks is shown in brackets. As bottom row, the table reports the mean win rate and AUPR over the classes. For each class, the best win rate is highlighted in red. The algorithms are sorted left to right according to the best mean win rate.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 December 2020 doi:10.20944/preprints202012.0808.v1

	CHA	SBM	SBM-DC	SBM-N	SBM-DC-N
collaboration (14)	1.00 (0.34)	0.07 (0.26)	0.00 (0.21)	0.00 (0.26)	0.07 (0.21)
contact (24)	1.00 (0.34)	0.04 (0.25)	0.04 (0.19)	0.00 (0.25)	0.04 (0.19)
covert (79)	0.80 (0.27)	0.13 (0.18)	0.09 (0.16)	0.09 (0.18)	0.09 (0.17)
friendship (13)	0.92 (0.26)	0.15 (0.17)	0.08 (0.13)	0.08 (0.17)	0.08 (0.13)
connectome (507)	0.99 (0.43)	0.01 (0.27)	0.01 (0.28)	0.01 (0.27)	0.01 (0.28)
foodweb (39)	0.77 (0.29)	0.15 (0.26)	0.05 (0.21)	0.15 (0.28)	0.00 (0.21)
trade (193)	0.58 (0.13)	0.10 (0.08)	0.34 (0.11)	0.12 (0.09)	0.34 (0.11)
coauthorship (4)	1.00 (0.83)	0.00 (0.45)	0.00 (0.38)	0.00 (0.55)	0.00 (0.47)
flightmap (26)	0.27 (0.17)	0.38 (0.21)	0.27 (0.19)	0.38 (0.21)	0.19 (0.18)
socialnetwork (1)	1.00 (0.20)	0.00 (0.15)	0.00 (0.11)	0.00 (0.14)	0.00 (0.09)
mean win rate (AUPR)	0.83 (0.33)	0.10 (0.23)	0.09 (0.20)	0.08 (0.24)	0.08 (0.20)

Suppl. Table 6. Results of CHA and SBM variants on ATLAS.
For each network of the ATLAS (considering $N \leq 100$, 900 networks) and for each algorithm, we applied the 10% link removal evaluation, obtaining the AUPR as measure of performance (see Methods section 4.2.1. for details). The table reports for each network class and for each algorithm the win rate and the mean AUPR (in brackets) over the networks of that class. For each class, the number of networks is shown in brackets. As bottom row, the table reports the mean win rate and AUPR over the classes. For each class, the best win rate is highlighted in red. The algorithms are sorted left to right according to the best mean win rate.

	CHA	SPM	HOPE	ProNE-SMF	node2vec	ProNE	NetSMF
contact-SFHH	0.08	0.08	0.08	0.07	0.07	0.07	0.06
contact-officefrance2013	0.16	0.16	0.15	0.11	0.11	0.09	0.06
socialnetwork-enronemployees	0.16	0.18	0.15	0.11	0.10	0.12	0.10
socialnetwork-facebookforum	0.02	0.01	0.01	0.01	0.01	0.01	0.01
contact-ACM2009	0.23	0.22	0.23	0.18	0.20	0.18	0.15
email-manufacturing	0.19	0.21	0.20	0.18	0.17	0.14	0.06
contact-primaryschool	0.23	0.23	0.20	0.16	0.18	0.14	0.17
contact-wirelesshaggle	0.40	0.37	0.39	0.23	0.25	0.12	0.04
email-EUcore	0.15	0.16	0.13	0.10	0.05	0.07	0.04
email-DNC	0.10	0.10	0.09	0.02	0.02	0.01	0.00
socialnetwork-UCImessages	0.03	0.02	0.01	0.02	0.01	0.01	0.01
socialnetwork-retweets	0.03	0.02	0.01	0.00	0.00	0.00	0.00
socialnetwork-mathoverflow	0.04	0.04	0.03	0.02	0.01	0.02	0.00
ARK201012	0.06	0.03	0.04	0.00	0.00	0.00	0.00
facebook	0.02	0.02	0.02	0.01	0.00	0.01	0.01
mean AUPR	0.13	0.12	0.12	0.08	0.08	0.07	0.05
win rate	0.80	0.60	0.20	0.00	0.00	0.00	0.00

Suppl. Table 7. Results of CHA, SPM and embedding methods on temporal networks.
For each temporal network and for each algorithm, we applied the temporal evaluation, obtaining the AUPR as measure of performance (see Methods section 4.2.2. for details), which is reported in the table. As bottom rows, the table reports the mean AUPR and win rate over the networks. For each network, the best AUPR is highlighted in red. The algorithms are sorted left to right according to the best win rate.