Robledo *et al. Applied Network Science* (2022) 7:35
https://doi.org/10.1007/s41109-022-00471-1

Applied Network Science

RESEARCH Open Access

Influence of clustering coefficient on network embedding in link prediction

Omar F. Robledo[1], Xiu-Xiu Zhan[1,2], Alan Hanjalic[1] and Huijuan Wang[1*]

*Correspondence: H.Wang@tudelft.nl

[1] Faculty of Electrical Engineering, Mathematics, and Computer Science, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands
Full list of author information is available at the end of the article

Abstract

Multiple network embedding algorithms have been proposed to perform the prediction of missing or future links in complex networks. However, we lack the understanding of how network topology affects their performance, or which algorithms are more likely to perform better given the topological properties of the network. In this paper, we investigate how the clustering coefficient of a network, i.e., the probability that the neighbours of a node are also connected, affects network embedding algorithms' performance in link prediction, in terms of the AUC (area under the ROC curve). We evaluate classic embedding algorithms, i.e., Matrix Factorisation, Laplacian Eigenmaps and node2vec, in both synthetic networks and (rewired) real-world networks with variable clustering coefficient. Specifically, a rewiring algorithm is applied to each real-world network to change the clustering coefficient while keeping key network properties. We find that a higher clustering coefficient tends to lead to a higher AUC in link prediction, except for Matrix Factorisation, which is not sensitive to the change of clustering coefficient. To understand such influence of the clustering coefficient, we (1) explore the relation between the link rating (probability that a node pair is the missing link) derived from the aforementioned algorithms and the number of common neighbours of the node pair, and (2) evaluate these embedding algorithms' ability to reconstruct the original training (sub)network. All the network embedding algorithms that we tested tend to assign higher likelihood of connection to node pairs that share an intermediate or high number of common neighbours, independently of the clustering coefficient of the training network. Then, the predicted networks will have more triangles, thus a higher clustering coefficient. As the clustering coefficient increases, all the algorithms but Matrix Factorisation could also better reconstruct the training network. These two observations may partially explain why increasing the clustering coefficient improves the prediction performance.

Keywords: Network embedding, Link prediction, Clustering coefficient

Introduction

Networks (Newman 2003, 2010; Albert and Barabási 2002) or, equivalently, graphs are commonly used to represent complex systems, the elements of which are represented by network *nodes* and their connections or relations by the *links*. Networks have been deployed to model phenomena in many fields (da Fontoura Costa 2011), such as biology (Girvan and Newman 2002; Winterbach et al. 2013), telecommunications or social

sciences (Bruch and Newman 2018; Liu et al. 2018). One of such phenomena is the emerging of the connections or relations between two network nodes, which is typically addressed by the approaches to *link prediction*. Link prediction aims to predict future or missing links based on, but not only, the observed network topology. The link prediction problem is ubiquitous, as the link to be predicted can be the interaction between two proteins (Kovács et al. 2019), a movie or book recommendation for a specific user in a multimedia service, or friend suggestions in a social network (Liben-Nowell and Kleinberg 2003).

The link prediction problem has been studied extensively over the past years (Lü and Zhou 2011). Initially, similarity-based methods were proposed (e.g., Liao et al. 2015), which predict links between the node pairs that are more similar. The similarity of a node pair is evaluated via the number of paths of a certain length[1] l between the two nodes, where usually $l = 2, 3, 4$. In those methods, node pairs that are connected by many short paths are considered to be likely connected. Recently, network embedding has attracted attention as a way of solving the link prediction problem (e.g., Torres et al. 2020; Zhang et al. 2018; Epasto and Perozzi 2019), and multiple methods have been proposed along this line to further improve the prediction quality, the computational efficiency, or both. These algorithms embed each node in a low-dimensional vector space based on the network topology (Cui et al. 2019). The closer the learnt representations of two nodes are in this vector space, the more likely it is that these two nodes are connected or related to each other. Matrix-based embedding methods, like Laplacian Eigenmaps (Belkin and Niyogi 2001), which embeds nodes via eigenvectors of graph matrices, and random-walk based embedding algorithms, like DeepWalk (Perozzi et al. 2014) and node2vec (Grover and Leskovec 2016), are examples of network embedding algorithms.

It has been observed that embedding-based link prediction methods may perform differently in different real-world networks (Khosla et al. 2021). This is because the performance of an embedding algorithm in link prediction depends on the properties of the underlying network, i.e., inherent patterns that enable link prediction, and how the embedding algorithm samples or preserves properties of the network. However, we still lack foundational understanding about how a network property influences the performance of embedding methods. This is challenging since real-world networks, different in many properties, especially network size, are difficult to be compared. Without such understanding, we are unable to explain the performance of network embedding algorithms or to select the most effective method, given a network whose links need to be predicted.

Hence, we aim to study the effect of network properties, specifically the *clustering coefficient* of a network, on the performance of network embedding algorithms in link prediction. The clustering coefficient of a network measures, on average, how densely the neighbours of each node are connected. Various metrics have been proposed to capture diverse topological properties of a network (Newman 2010). The focus of this work is on the effect of the clustering coefficient, a basic local or microscopic network property, and it is motivated by three perspectives, also in relation to previous work.

[1] The length of a path is the number of links that constitute the path.

First, preliminary evidence in literature indicates that the clustering coefficient of a network may influence the performance of link prediction algorithms. For example, similarity-based link prediction methods tend to perform better in networks with a high clustering coefficient. Feng et al. (2012) have showed that by applying six similarity-based measures to multiple synthetic and real networks with different clustering coefficients. Cao et al. (2019) have showed that similarity-based measures can outperform network embedding algorithms based on random walks in networks with a small average path length. A comparative study performed by Khosla et al. (2021) has shown that the difference in accuracy between different network embedding algorithms may depend on, among other properties, the clustering coefficient.

Second, the aforementioned preliminary evidences have been obtained by comparing the performance of prediction methods in networks with not only different clustering coefficient (or average path length), but also different local topological properties; e.g., link density and degree distribution. Thus, the difference in performance could be attributed to the influence of other structural properties. We still lack a systematic method to identify the influence of the clustering coefficient without the influence of other local properties. To address this challenge, we design a methodology that allows to compare the performance in networks, both real and synthetic, with different clustering coefficients, but the same local properties.

Third, it is essential to understand the influence of a microscopic property, such as the clustering coefficient, before we explore further the influence of a complex macroscopic network property, or of multiple network properties, due to the correlations between them Li et al. (2011). Several works have revealed the relation between the clustering coefficient (or transitivity) and other macroscopic network properties (Wharrie et al. 2019; Asikainen et al. 2020; Peixoto 2022; Foster et al. 2011; Orman et al. 2013). For example, Wharrie et al. (2019) have shown that communities can be formed by processes of motif generation, such as triadic closure. Asikainen et al. (2020) have shown the emergence of core-periphery structure introduced partially by triadic closure. Such nontrivial relation between the microscopic clustering coefficient and macroscopic network properties illustrates the difficulty to explore the effect of macroscopic or multiple network properties on the performance of link prediction algorithms. Network models that generate networks in multiple scales (Peixoto 2022) could be an interesting approach to control micro- and macroscopic properties at the same time. Moreover, macroscopic network properties, such as the average path length and community structure, have a higher computational complexity. Hence, it is more practical to use the clustering coefficient to pre-estimate the performance of prediction methods in large networks.

We focus on the question of how the clustering coefficient influences the performance of network embedding algorithms. Classic network embedding methods including Laplacian Eigenmaps and Matrix Factorisation, and node2vec are considered. We evaluate their performance in both network models and real-world networks with a variable clustering coefficient. For each selected real-world network, we modify its clustering coefficient through the rewiring of links, which keeps basic network properties, like the degree of each node, unchanged. We find that network embedding methods tend to perform better in link prediction in networks with a higher clustering coefficient, except for Matrix Factorisation, which performs well and stably overall. In order to understand

how and why these embedding algorithms differ in performance, we explore further the distance of two nodes in the embedding space in relation to their number of common neighbours. We find that these embedding algorithms tend to embed node pairs with a large or intermediate number of common neighbours closely, independently of the clustering coefficient of the network. Hence, they tend to predict such node pairs as the missing links, leading to a large number of triangles, or high clustering coefficient, in the predicted network. This may partially explain why the prediction quality is better as the clustering coefficient increases.

Moreover, an embedding algorithm is not expected to predict the missing links well if the embedding cannot retain the given training network topology. Hence, we evaluate further the influence of the clustering coefficient on the embedding algorithms in the network reconstruction task, where each embedding method derives the embedding of nodes based on the given network topology, and the closest node pairs in the embedding space are reconstructed as links. Consistently, we find that embedding methods better reconstruct the given network topology if the network has a higher clustering coefficient. The better performance of the embedding algorithms as the clustering coefficient increases, and their possible outperforming of $l = 2$ similarity-based (also called the Common Neighbors) method imply that embedding algorithms, besides the number of common neighbours, should be considered for link prediction tasks.

This paper is organised as follows. In "Methods" section, we describe our approach, detail the model used for generating synthetic networks, along with the characteristics of the real networks used, and briefly review the network embedding techniques compared. In "Results" section, we analyse the performance of the embedding algorithms in link prediction as the clustering coefficient varies in the network models and real-world networks. To understand such influence, we explore the distance of a node pair in the embedding space in relation to the number of common neighbours of the node pair, and analyse how the clustering coefficient affects the ability of the algorithms to reconstruct the given training network. Finally, we present our conclusions in "Conclusions" section.

Methods

We aim to understand the influence of network topological properties on the performance of network embedding algorithms. Specifically, we focus on how the clustering coefficient, i.e., the link density of the neighbours of a node, affects the performance of network embedding algorithms in link prediction. First, we introduce our evaluation method for the algorithms in the link prediction task. Afterwards, we describe how to construct networks with various clustering coefficients via network models, and via rewiring a real-world network, respectively. Finally, several classic embedding algorithms that we have chosen are briefly reviewed.

Evaluation of network embedding algorithms

A network $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, with its set $\mathcal{N}$ of $N = |\mathcal{N}|$ nodes, and set $\mathcal{L}$ of $L = |\mathcal{L}|$ links, can be represented by an *adjacency matrix* A, an $N \times N$ matrix in which $a_{i,j} = 1$ if node n_i and n_j are connected, and $a_{i,j} = 0$ otherwise. The objective of network embedding algorithms is to learn a low-dimensional representation for each node in the network such that they can be used for an inference task while keeping the structure of the network.

We will evaluate their performance in two tasks, namely link prediction and network reconstruction.

Link prediction is the task of predicting the links that are not observed in a network based on the observed network topology. We divide the links $\mathcal{L}$ of a given network in two groups: 75% of the links are chosen uniformly at random as the training link set $\mathcal{L}^{train}$, or the observed network to learn the embedding, and the remaining 25% links are considered as the test set $\mathcal{L}^{test}$, i.e., the links to predict. The embedding vector for each node learnt from the training set is used further to compute the distance score between each node pair, which will be defined in the next subsection. The distance score of a node pair is used as an estimation of the likelihood that the node pair is connected. Node pairs that have the highest likelihood are predicted as links, denoted as set $\mathcal{L}^{predict}$.

We evaluated link prediction performance via the *Area Under the ROC Curve* (*AUC*) (Herlocker et al. 2004; Hanley and McNeil 1982). The *Area Under the Precision-Recall Curve* (*AUPRC*) (Raghavan et al. 1989) has also been used to address class-imbalanced classification problems (Saito and Rehmsmeier 2015), as is the case of link prediction in sparse networks. We focus on AUC as an example, since we aim to compare the performance of embedding algorithms in networks with the same link density, but different clustering coefficients. A negative link is a node pair that is not connected in network $\mathcal{G}$, or a link that does not exist in $\mathcal{G}$. To evaluate link prediction, we consider the set $\mathcal{L}^{test}$ of positive links, and a set $\mathcal{L}^{-}$ of $|\mathcal{L}^{-}| = |\mathcal{L}^{test}|$ negative links that are randomly selected from all possible negative links. The measure *AUC* can be interpreted as the probability that a randomly chosen missing link will have a higher score than a negative link randomly chosen from $\mathcal{L}^{-}$.

In the network reconstruction task, the original network is used to obtain the embedding vectors for all the nodes. The $\eta \times L$ node pairs with the highest (or lowest) scores are predicted as links in the reconstructed network, denoted by $\mathcal{L}^{predict}(\eta)$, where $\eta \in (0, 1]$. The quality of network reconstruction of an embedding algorithm can be measured by $Precision@\eta = \frac{|\mathcal{L}^{predict}(\eta) \cap \mathcal{L}|}{\eta \times L}$, which is also referred to as *Precision@k* (Wang et al. 2016). The precision of a random estimator, or reconstruction, is equal to the link density of the network, which is both independent of η and small, since real-world networks are mostly sparse.

Network construction

In this section, we introduce how to construct networks that have the same key network property, like the degree distribution, but different clustering coefficients. First, the clustering coefficient is defined. Then, we briefly describe Ostroumova's Polynomial Model, which we use to generate synthetic networks with a given degree distribution but a tuneable clustering coefficient. Finally, we describe the empirical network datasets that will be used and the rewiring method to tune the clustering coefficient of a given empirical network without changing the degree of each node.

We consider undirected networks without any self-loop nor duplicated links. The *degree* k_i of a node n_i is the number of connections that said node has; i.e., $k_i = \sum_{j=0}^{N-1} a_{i,j}$. The *clustering coefficient* c_i of a node n_i measures how densely its neighbours are connected:

Table 1 Clustering coefficient for networks generated by different combination of γ and β. For each (γ, β), the clustering coefficient is the average over 100 generated networks

	$\gamma = 2.25$	$\gamma = 2.50$	$\gamma = 3.00$
$\beta = 0.00$	0.23 ± 0.03	0.10 ± 0.02	0.04 ± 0.01
$\beta = 0.20$	0.41 ± 0.03	0.25 ± 0.02	0.18 ± 0.01
$\beta = 0.40$	0.58 ± 0.02	–	–
$\beta = 0.60$	–	0.54 ± 0.02	0.46 ± 0.01

$$c_i = \begin{cases} \dfrac{\sum_{j=0}^{N-1} a_{i,j} \sum_{l=0}^{N-1} a_{i,l} \cdot a_{j,l}}{k_i \cdot (k_i - 1)}, & \text{if } k_i > 1 \\ 0 & \text{if } k_i = 1 \end{cases} \tag{1}$$

The clustering coefficient of the network is the average over all nodes. That is,

$$C = \frac{1}{N} \sum_{i=0}^{N-1} c_i. \tag{2}$$

We use Ostroumova's Polynomial Model (Ostroumova et al. 2013) to generate synthetic networks with a given target degree distribution, and a tuneable clustering coefficient. We consider the power-law degree distribution, i.e., $p(k) \sim k^{-\gamma}$, where $\gamma \in (2, 3)$, as observed in many real-world networks (Barabási and Albert 1999). Ostroumova's Polynomial Model is a growing network model based on Preferential Attachment. Its input parameters are γ, m and β. γ is the slope of the power-law distribution for the degree of the nodes, m is the number of links that we add when creating a new node, and β is related to the probability of new links forming triangles in the network, so it controls the clustering coefficient. γ needs to be in the range [2, 3], and β in the range [0, 1]; however, with the polynomial that we used, β has a restriction depending on the value of γ that has been chosen, such that $\beta_{max} = 2 - \dfrac{2}{\gamma - 1}$; i.e., β can only reach its maximum value (that is, 1) if $\gamma = 3.00$.

The possible values for γ and β that we have considered are given in Table 1, together with the resultant clustering coefficient. We have fixed $m = 2$, so that the generated network is sparse, and $N = 1000$ nodes. According to the network model, the maximum possible value for β is 0.40 when $\gamma = 2.25$. For each combination of γ and β, the reported clustering coefficient is the average over 100 realisations. The choice of β is motivated by the objective of generating networks with three different clustering coefficients per each degree distribution, while keeping the maximum clustering coefficient comparable.

We consider further two empirical networks, i.e., *Hamsterster* and *Maayan-vidal* (Kunegis 2013; Rual et al. 2005). A link in *Hamsterster* represents friendship relation between two nodes. *Maayan-vidal* is a protein network with links indicating protein-protein interactions. Basic properties of the two networks are reported in Table 2.

We apply Alstott's rewiring algorithm (Alstott et al. 2018) to change the clustering coefficient of a given real-world network without changing the degree of any node. The rewiring process iterates the following procedure until the target clustering coefficient is achieved. First, we choose the two non-existing links with the highest number of

Table 2 Properties of the empirical networks. *N* is the number of nodes, *L* is the number of links, *C* is the clustering coefficient of the network, and f_{CC} is the fraction of nodes that belong to the largest connected component in the network

	N	L	C	f_{CC}
Hamsterster	1858	12534	0.1414	0.9623
Maayan-Vidal	3023	6149	0.0658	0.9206

common neighbours that meet the following requirements: they involve four different nodes, and there are exactly two links between them. Then, we rewire them by exchanging one end node from each link such that the degree remains the same, but the number of common neighbours between the connected nodes is the highest possible.Rewiring links in a network not only modifies the clustering coefficient, but also introduces randomness to the network topology. For *Hamsterster*, whose original clustering coefficient is 0.1414, we create two rewired networks with a high clustering coefficient $C = 0.20$, and $C = 0.30$ respectively; for *Maayan-Vidal* with its original clustering coefficient 0.0658, we also generate two rewired networks with $C = 0.10$, and $C = 0.20$ respectively.

Embedding algorithms

We consider three classic embedding algorithms: matrix-based embedding algorithm *Laplacian Eigenmaps* (Belkin and Niyogi 2001), *Matrix Factorisation*, which has also been applied in recommender systems (Koren et al. 2009; Koren 2008; Kotu and Deshpande 2019), and the random-walk-based network embedding algorithm *node2vec* (Grover and Leskovec 2016). In the link prediction task, we also compare these embedding algorithms with a structural similarity-based link prediction method, *Common Neighbours*, which estimates the likelihood that two nodes are connected via the number of their common neighbours.

We denote the representation of a node n_i as $\overrightarrow{y_i}$ for every algorithm in this section. We choose the dimension of the representations to be 10 for Laplacian Eigenmaps and Matrix Factorisation. For node2vec, we keep the default values for all the parameters (including the embedding dimension, fixed at 128), except for the two parameters of the random walk, which we optimise through a parameter sweep. In each embedding algorithm, a score that measures the distance between two nodes in the embedding space is defined and used as an estimation of the likelihood that the two nodes are connected.

Laplacian Eigenmaps Given an undirected network $\mathcal{G}$, the objective is to obtain the representation for each node with dimension d, such that the nodes that are connected in the network are closer in the representation domain. Specifically, the objective function is

$$\underset{Y^T DY=I}{\operatorname{argmin}} tr(Y^T QY), \tag{3}$$

where the solution Y is an $N \times d$-dimensional matrix whose ith row y_i is the d-dimensional representation of node n_i, $Q = D - A$ is the *Laplacian* matrix of the network, and D is the diagonal matrix that contains the degree of each node. Moreover,

$$tr(Y^T QY) = \sum_{i,j} a_{i,j} \| \overrightarrow{y_i} - \overrightarrow{y_j} \|^2, \tag{4}$$

The obtained embedding vector of a node contains actually this node's eigenvector components corresponding to the d smallest, but non-zero, eigenvalues of the problem $Q\overrightarrow{y} = \lambda D\overrightarrow{y}$.

The distance score of a node pair is then defined as

$$s_{LE}(n_i, n_j) = || \overrightarrow{y_i} - \overrightarrow{y_j} ||_2^2, \tag{5}$$

A higher score of a node pair suggests a lower likelihood of being connected.

Matrix factorisation This algorithm has been used in *recommender systems*. It calculates a representation for every *user* and *item*, based on the ratings given by the users to the items. In this work, we adapt the algorithm for unweighted networks, where a link between two nodes is interpreted as a rating of 1. Matrix Factorisation aims to obtain a representation for each node such that connected nodes are close in embedding, i.e.,

$$\underset{Y,b}{\operatorname{argmin}} \frac{1}{L} \sum_{(n_i,n_j)\in\mathcal{L}} \left[(1 - b_i - b_j - \overrightarrow{y_i} \cdot \overrightarrow{y_j})^2 + \lambda \cdot (|| \overrightarrow{y_i} ||_2^2 + || \overrightarrow{y_j} ||_2^2 + b_i^2 + b_j^2) \right], \tag{6}$$

where b_i is the bias of node n_i, and λ is the regularisation term.

The distance score between two nodes is measured as

$$s_{MF}(n_i, n_j) = \overrightarrow{y_i} \cdot \overrightarrow{y_j} + b_i + b_j. \tag{7}$$

Nodes with a higher score are more likely to be connected.

node2vec Inspired by *word2vec*, node2vec performs biased random walks on the given network to sample trajectory paths via random walks. Node pairs that are within a given distance on a trajectory path are used as the input of *Skip-gram*, a representative language model that embeds nodes, which are regarded as words, into vectors. The score for each node pair is

$$s_{n2v}(n_i, n_j) = \overrightarrow{y_i} \cdot \overrightarrow{y_j}. \tag{8}$$

A higher score between two nodes suggests a higher likelihood that the two nodes are connected.

Results

In this section, we evaluate the performance of all the algorithms discussed in Sect. 2 in the link prediction task for the datasets described previously. Then, we perform network reconstruction to study the relation between the capacity of an embedding to keep the original structure of the network and its performance on link prediction via *AUC* and *precision*. Finally, we compare the number of common neighbours between two nodes and the score given by each algorithm to further explain the results obtained.

Table 3 AUC for all the algorithms applied to the networks generated by the Polynomial model

Parameters		Algorithms			
		Common Neighbours	**Matrix Factorisation**	**Laplacian Eigenmaps**	**node2vec**
$\gamma = 2.25$	$\beta = 0.00$	0.501 ± 0.018	$\mathbf{0.841 \pm 0.020}$	0.472 ± 0.025	0.429 ± 0.016
	$\beta = 0.20$	0.561 ± 0.021	$\mathbf{0.835 \pm 0.013}$	0.511 ± 0.022	0.462 ± 0.015
	$\beta = 0.40$	0.628 ± 0.018	$\mathbf{0.829 \pm 0.014}$	0.565 ± 0.023	0.511 ± 0.018
$\gamma = 2.50$	$\beta = 0.00$	0.511 ± 0.009	$\mathbf{0.728 \pm 0.018}$	0.487 ± 0.018	0.452 ± 0.018
	$\beta = 0.20$	0.578 ± 0.012	$\mathbf{0.719 \pm 0.016}$	0.527 ± 0.020	0.484 ± 0.018
	$\beta = 0.60$	0.699 ± 0.013	$\mathbf{0.719 \pm 0.032}$	0.659 ± 0.027	0.615 ± 0.022
$\gamma = 3.00$	$\beta = 0.00$	0.502 ± 0.005	$\mathbf{0.615 \pm 0.018}$	0.481 ± 0.014	0.459 ± 0.017
	$\beta = 0.20$	0.576 ± 0.009	$\mathbf{0.626 \pm 0.016}$	0.528 ± 0.017	0.517 ± 0.018
	$\beta = 0.60$	$\mathbf{0.701 \pm 0.012}$	0.615 ± 0.021	0.660 ± 0.022	0.645 ± 0.020

In each class of networks, the AUC of the algorithm that performs the best is in bold

Table 4 AUC for all the algorithms applied to real networks Hamsterster and Maayan-Vidal and their rewired versions

Datasets		Algorithms			
		Common Neighbours	**Matrix Factorisation**	**Laplacian Eigenmaps**	**node2vec**
Hamsterster	Original	0.749 ± 0.003	$\mathbf{0.807 \pm 0.005}$	0.738 ± 0.035	0.780 ± 0.005
	$C_L = 0.20$	0.788 ± 0.004	$\mathbf{0.806 \pm 0.004}$	0.747 ± 0.044	0.786 ± 0.005
	$C_L = 0.30$	$\mathbf{0.863 \pm 0.005}$	0.800 ± 0.006	0.777 ± 0.040	0.823 ± 0.007
Maayan-Vidal	Original	0.590 ± 0.003	$\mathbf{0.746 \pm 0.008}$	0.624 ± 0.016	0.607 ± 0.008
	$C_L = 0.10$	0.643 ± 0.005	$\mathbf{0.742 \pm 0.007}$	0.636 ± 0.016	0.617 ± 0.008
	$C_L = 0.20$	$\mathbf{0.744 \pm 0.006}$	0.721 ± 0.008	0.695 ± 0.023	0.681 ± 0.009

In each network, the AUC of the algorithm that performs the best is in bold

Link prediction

Performance evaluation

First, we evaluate the performance of all the aforementioned algorithms in the task of link prediction in both network models and real-world networks with a varying clustering coefficient. The objective is to understand how clustering coefficient influences the performance of the algorithms.

We start with the performance measure AUC, the most widely studied link prediction measure. The algorithms are compared in polynomial model and real-world networks, with the results shown in Tables 3 and 4 respectively. For each class of networks, i.e. Polynomial Model with a parameter set (γ, β), or networks that are rewired from a given real-world network with a target clustering coefficient, the AUC performance shown for a given algorithm is the average over 10 network realisations, and 10 choices of the training and test datasets for each network realisation.

For both *Matrix Factorisation* and *node2vec*, the representation is calculated 10 times for each network and each training data set due to the random nature of these

algorithms in, e.g., initialisation and random walk process. The link prediction quality is the average over all realisations of the embedding, training and test data sets.

In all the networks that we considered, all algorithms perform better as the clustering coefficient increases, except for *Matrix Factorisation*, whose performance remains relatively the same. When we rewire a given real-world network towards a higher clustering coefficient, the rewired network is also more random, thus more difficult to predict than the original real-world. Therefore, our observation that the AUC of *Matrix Factorisation* does not improve with an increasing clustering coefficient in (rewired) real-world networks does not exclude the possibility that a higher clustering coefficient could facilitate the performance of *Matrix Factorisation*. All the other algorithms perform better in the rewired real-world networks with a higher clustering coefficient, even though the randomness of the rewired networks could lower their performance. This illustrates evidently that a high clustering coefficient could lead to a high link prediction quality. It is found that network embedding algorithms based on random walks sometimes perform worse in link prediction than Common Neighbors (Cao et al. 2019). The common neighbours method is expected to perform better as the clustering coefficient increases. Our finding that a high clustering coefficient leads to a better performance of network embedding algorithms suggests that, for the link prediction task in networks with a high clustering, not only the common neighbours but embedding algorithms could be considered as well. In all network classes considered, the common neighbours method, which performs well, can be in many cases outperformed in terms of AUC by one of these embedding algorithms. Among the three embedding algorithms, *Matrix Factorisation* performs mostly the best in AUC in each class of networks.

Link score/rating analysis

In order to further understand why and how the three embedding methods perform differently, we explore the scores/ratings of the test and negative links with a given number of common neighbours respectively derived by each embedding algorithm.

As examples, we consider the generated networks ($\gamma = 2.50$, $\beta = 0.00, 0.60$); and the real-world networks Hamsterster and Maayan-vidal, as well as their rewired networks with clustering coefficient $C = 0.30$ and $C = 0.20$ respectively. However, the same conclusions can be drawn from all the networks discussed in Sect. 2.2. Firstly, we explore the probability distribution of the number of common neighbours of a test link and a negative link, in each network class with a given parameter set. As shown in the subfigure (a), of Figures 1, 2, 3, 4, 5 and 6, most links, either test or negative links have a small number of common neighbours where as few have a large number of common neighbours. Moreover, the maximal possible number of common neighbours decreases in polynomial networks as the clustering coefficient increases. The contrary has been observed in (re-wired) real-world networks. The higher clustering coefficient of polynomial networks is obtained via the formation of more triangles. The rewired real-world network with a higher clustering coefficient is obtained via rewiring, where links are rewired to disconnected node pairs with a large number of common neighbours.

Figures 1, 2, 3, 4, 5 and 6 show further the average score together with its standard deviation for test and negative links with a given number of common neighbours, for each embedding algorithm. For both test and negative links, the average score increases

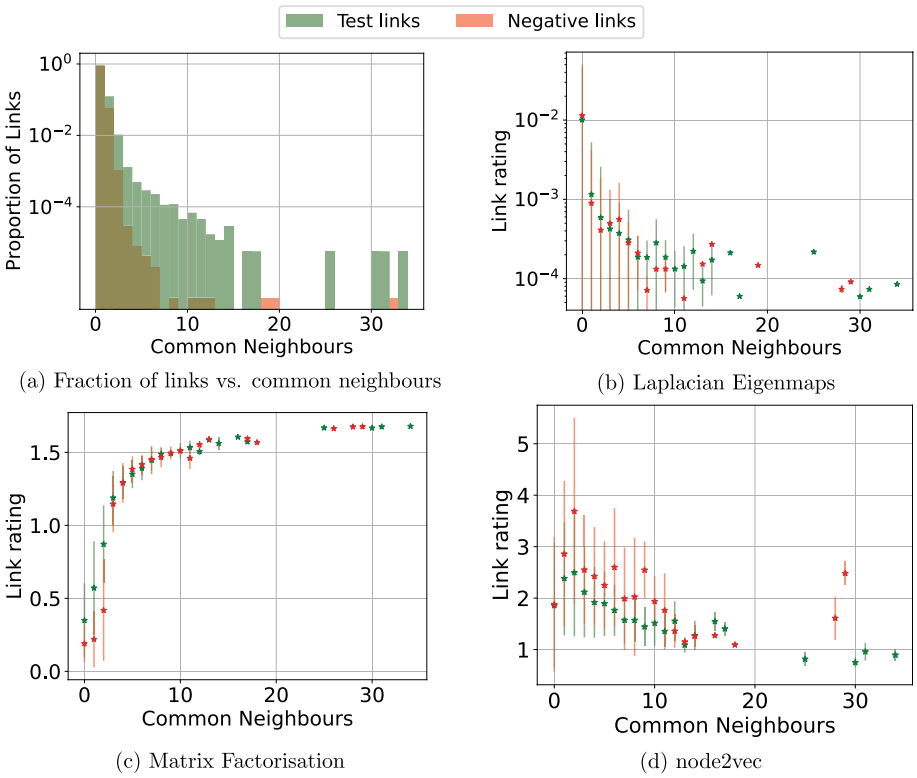


Fig. 1 **a** Fraction of links of the generated networks ($\gamma = 2.50, \beta = 0.00$) that share a given number of neighbours. Average rating of negative and test links with a given number of common neighbours derived by **b** Laplacian Eigenmaps, **c** Matrix Factorisation, and **d** node2vec

monotonically with the number of common neighbours in *Matrix Factorisation* in every data set but the Hamsterster one. The average score decreases with the number of common neighbours in *Laplacian Eigenmaps*, whereas, for node2vec, it increases first and decrease afterwards as the number of common neighbours increases. These observations suggest that node pairs with a large number of common neighbours tend to be predicted as the missing links by *Matrix Factorisation* and *Laplacian Eigenmaps*, while *node2vec* tends to predict node pairs with an intermediate number of common neighbours as missing links. While most node pairs have a small number of common neighbours, *Matrix Factorisation* and *Laplacian Eigenmaps* (*node2vec*) assign a higher score to node pairs with a high (intermediate) number of neighbours respectively, thus tend to better identify missing links with high (intermediate) number of neighbours.

Network reconstruction

We explore further whether an embedding algorithm could better reconstruct a network if the network has a higher clustering coefficient. We aim to understand whether these embedding algorithms' better performance in link prediction as the clustering coefficient increases is associated with their capability to better reconstruct the network.

Given an embedding algorithm and a network, either generated from the Polynomial Model or a (rewired) real-world network, we obtain the embedding vectors for all

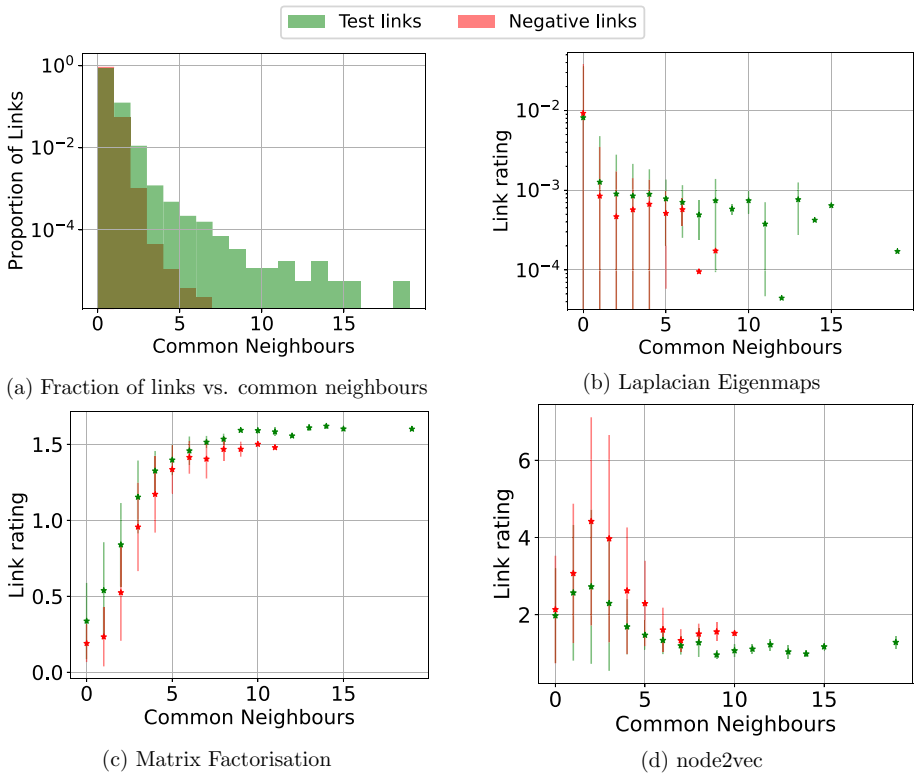
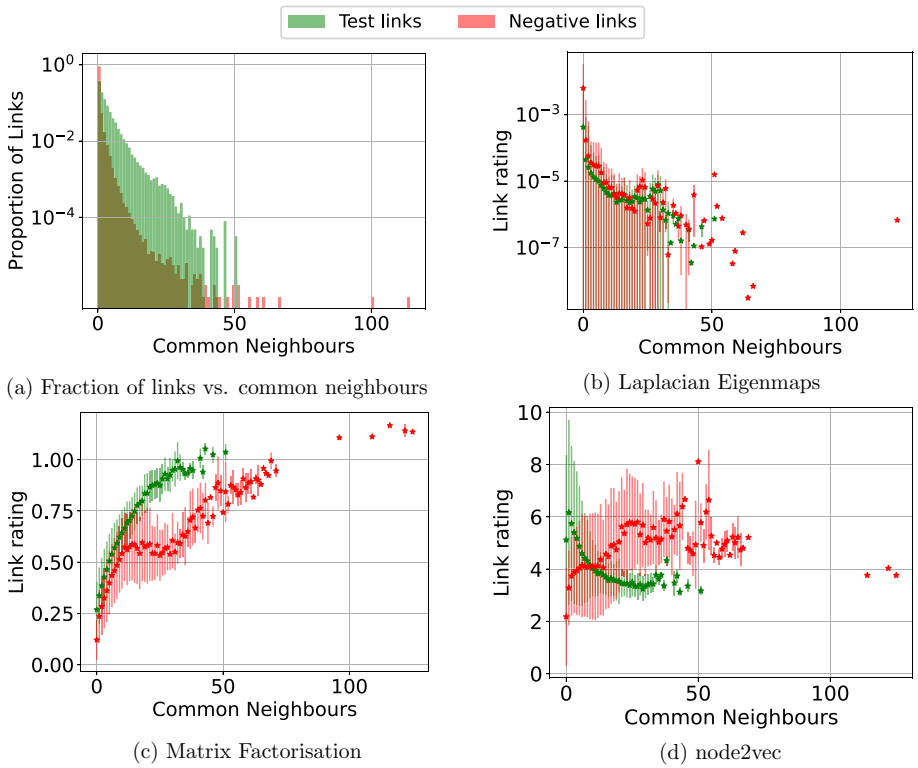


Fig. 2 **a** Fraction of links of the generated networks ($\gamma = 2.50, \beta = 0.60$) that share a given number of neighbours. Average rating of negative and test links with a given number of common neighbours derived by **b** Laplacian Eigenmaps, **c** Matrix Factorisation, and **d** node2vec

the nodes. The $\eta \times L$ node pairs with the highest (or lowest) scores are considered to be connected; i.e. the reconstructed (partial) network. We plot the reconstruction quality *Precision@*η as a function of $\eta \in (0, 1]$ from Figure 7, 8, 9, 10 and 11. As in the link prediction task, the reconstruction quality is the average of 10 network realisations per model parameter set (γ, β), or of 10 rewired networks per targeting clustering coefficient. For both *Matrix Factorisation* and *node2vec*, the reconstruction quality for a network realisation is the average over 10 realisations of the embedding. In *node2vec*, the two parameters that governs the random walk are set as the ones that lead to the best reconstruction quality.

When $\eta = 1$, i.e., when we aim to reconstruct the same number of links, L, as in the given network, *Precision@*$(\eta = 1)$ measures the reconstruction quality in terms of the overlap of the reconstructed network and the given network. Hence, it captures to what extent an embedding algorithm could keep the given network topology. When $\eta < 1$, *Precision@*η quantifies how the ηL node pairs with the highest rating overlap with the given network. When we reconstruct the network randomly, the precision *Precision@*$\eta = \frac{2L}{N(N-1)}$ equals the link density, which is small in real-world sparse networks.

Figures 7, 8, 9, 10 and 11 show the trend that the higher the clustering coefficient of the network is, the higher the precision tends to be; i.e., the bigger the part of the



(a) Fraction of links vs. common neighbours

(b) Laplacian Eigenmaps

(c) Matrix Factorisation

(d) node2vec

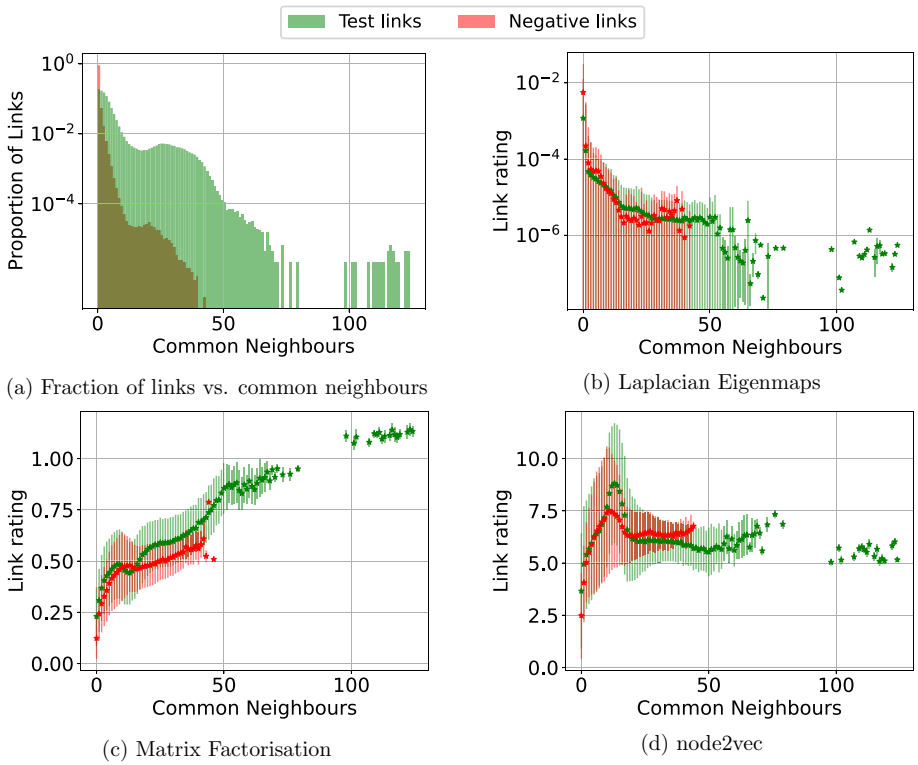
Fig. 3 **a** Fraction of links of the Hamsterster network (original) that share a given number of neighbours. Average rating of negative and test links with a given number of common neighbours derived by **b** Laplacian Eigenmaps, **c** Matrix Factorisation, and **d** node2vec

structure that the representations tend to be able to keep ($Precision@(\eta = 1)$). This is particularly apparent in the real networks (Figs. 10 and 11). The Matrix Factorisation algorithm, however, is equally able to reconstruct a given network regardless of its clustering coefficient.

Our finding that the network reconstruction quality increases as the clustering coefficient increases is in line with our previous observation that the embedding methods perform better in link prediction tasks when the clustering coefficient is larger. An embedding algorithm is not expected to predict the missing links well if it cannot keep or reconstruct the given network.

Conclusions

In this paper, we study how the clustering coefficient of the network affects the performance of embedding algorithms in link prediction tasks. We employ the Polynomial Model and Alstott's rewiring method to generate both synthetic and real networks with a variable clustering coefficient to evaluate embedding algorithms' performance in link prediction. We find that Matrix Factorisation is the only method that is not evidently benefited from higher clustering in the network on the link prediction task. Both node2vec and Laplacian Eigenmaps perform better in terms of AUC when the clustering coefficient increases. Besides, Matrix Factorisation has the best



(a) Fraction of links vs. common neighbours

(b) Laplacian Eigenmaps

(c) Matrix Factorisation

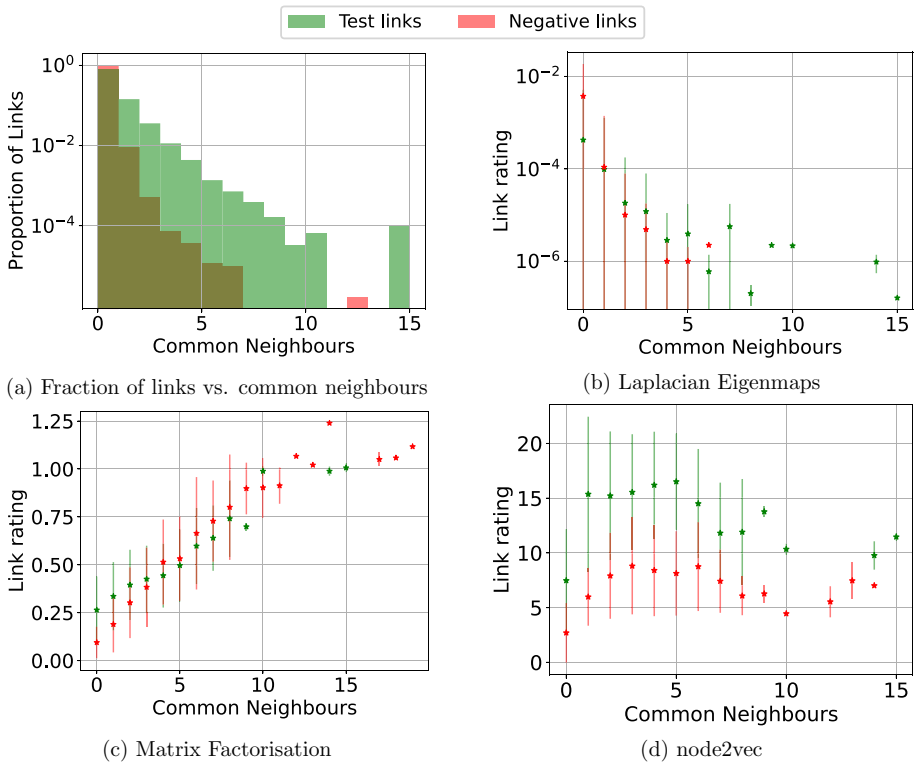
(d) node2vec

Fig. 4 **a** Fraction of links of the Hamsterster network (rewired $C = 0.30$) that share a given number of neighbours. Average rating of negative and test links with a given number of common neighbours derived by **b** Laplacian Eigenmaps, **c** Matrix Factorisation, and **d** node2vec

performance in most of the networks that we considered, except for the few with the highest clustering coefficient, where the Common Neighbor metric performs the best. Hence, in a network with a high clustering coefficient, embedding algorithms are supposed to perform well and the Common Neighbor metric could also be considered.

We explore further how these algorithms differ in assigning ratings/scores to node pairs with a given number of common neighbours. While Matrix Factorisation and Laplacian Eigenmaps give a higher probability (higher score for Matrix Factorisation, and lower for Laplacian Eigenmaps) of being missing links to node pairs with a high number of common neighbours, node2vec does that to node pairs with an intermediate number of common neighbours. Predicting node pairs with a large or intermediate number of common neighbours as the missing links tends to lead to more triangles in the predicted network, thus a high clustering coefficient, in contrast to predicting node pairs with a small number of common neighbours. This may partially explain why the prediction is better when the clustering coefficient of the given network is high.

Finally, we explore whether the increasing performance of an embedding algorithm as the clustering coefficient increases is associated with the algorithm's capability to better reconstruct the training sub-network. We find that, in (rewired) real-world networks, all the embedding algorithms can indeed better reconstruct the training network as the clustering coefficient increases. This trend is less evident in the network models, which are more random than real-world networks.



(a) Fraction of links vs. common neighbours

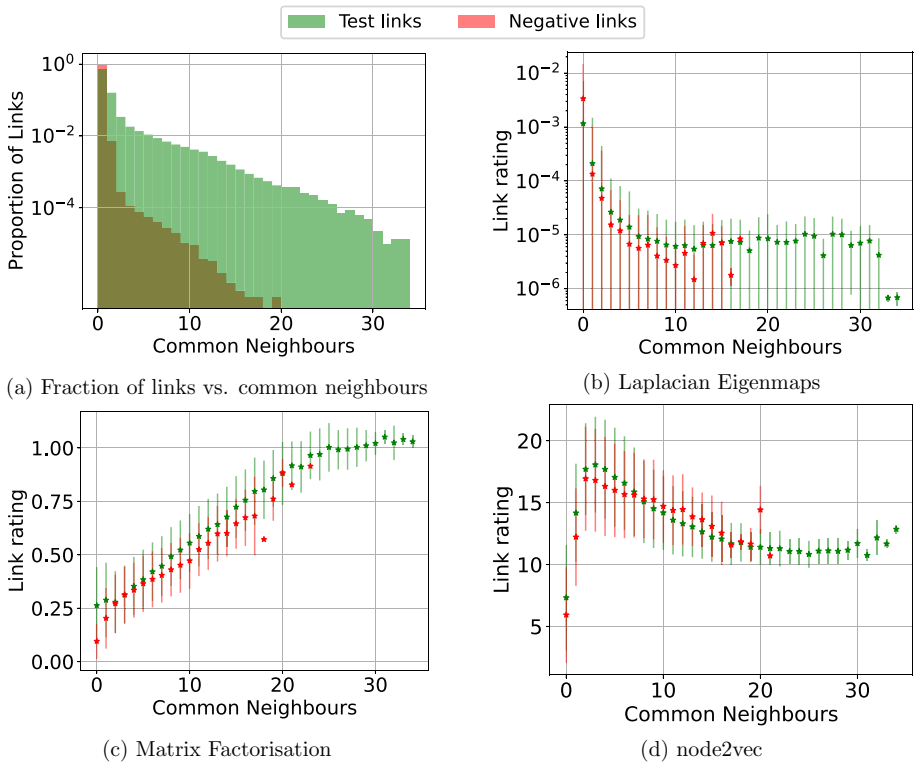
(b) Laplacian Eigenmaps

(c) Matrix Factorisation

(d) node2vec

Fig. 5 **a** Fraction of links of the Maayan-Vidal network (original) that share a given number of neighbours. Average rating of negative and test links with a given number of common neighbours derived by **b** Laplacian Eigenmaps, **c** Matrix Factorisation, and **d** node2vec

In this work, we illustrate the method to investigate how network or data properties influence the embedding algorithms, although our methodology is not limited to network embedding algorithms. We have confined ourselves to the clustering coefficient, the link prediction task, and limited choices of algorithms and networks. Generalisations can be made along with the following perspectives. The effect of other or multiple network properties, whose challenges have been discussed in Sect. 1, can be explored further. For example, the effect of degree-degree correlation and community structure can be investigated, since positive (negative) degree-degree correlation and community structure have been observed in social (man-made) networks (Van Mieghem et al. 2010; Fortunato 2010). It is interesting to study which algorithm may perform the best for a given network with specific properties, in view of not only AUC but also other evaluation methods like AUPRC and the complexity. Advanced embedding algorithms, e.g., for temporal networks, have been developed recently (Zhan et al. 2020; Torricelli et al. 2020; Tandon et al. 2021). Our method can also be further applied to explore the influence of network properties on other network-based algorithms (Wang et al. 2022), or dynamic processes deployed on networks (Pastor-Satorras et al. 2015). More in-depth investigation of the mechanisms of embedding algorithms in relation to the network properties they could retain may further inspire the design of embedding algorithms.



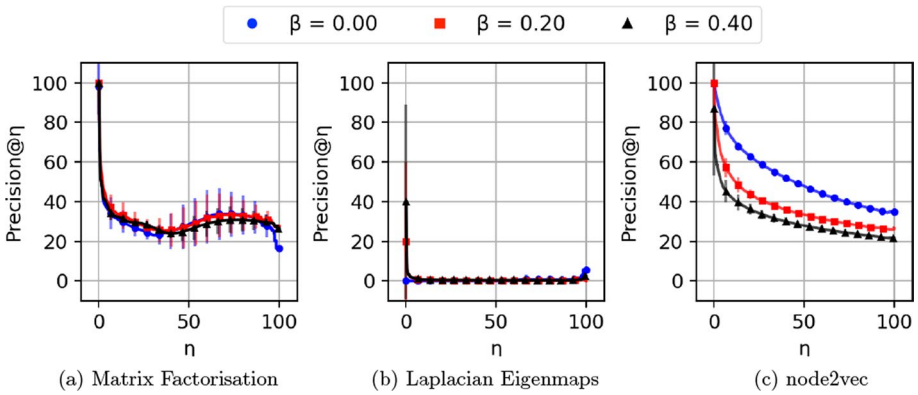
(a) Fraction of links vs. common neighbours

(b) Laplacian Eigenmaps

(c) Matrix Factorisation

(d) node2vec

Fig. 6 **a** Fraction of links of the Maayan-Vidal network (rewired $C = 0.20$) that share a given number of neighbours. Average rating of negative and test links with a given number of common neighbours derived by **b** Laplacian Eigenmaps, **c** Matrix Factorisation, and **d** node2vec



(a) Matrix Factorisation

(b) Laplacian Eigenmaps

(c) node2vec

Fig. 7 *Precision@η* obtained for network reconstruction, in average, applying each method to each generated data set ($\gamma = 2.25$). η is expressed in percentage

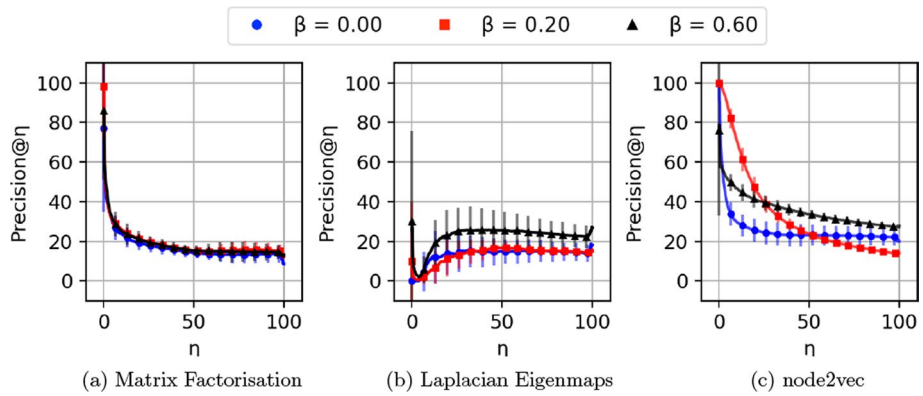


Fig. 8 *Precision@*η obtained for network reconstruction, in average, applying each method to each generated data set ($\gamma = 2.50$). η is expressed in percentage

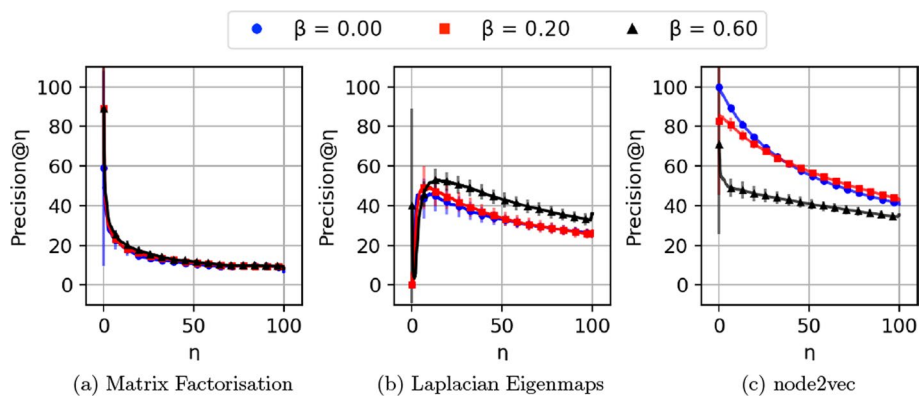


Fig. 9 *Precision@*η obtained for network reconstruction, in average, applying each method to each generated data set ($\gamma = 3.00$). η is expressed in percentage

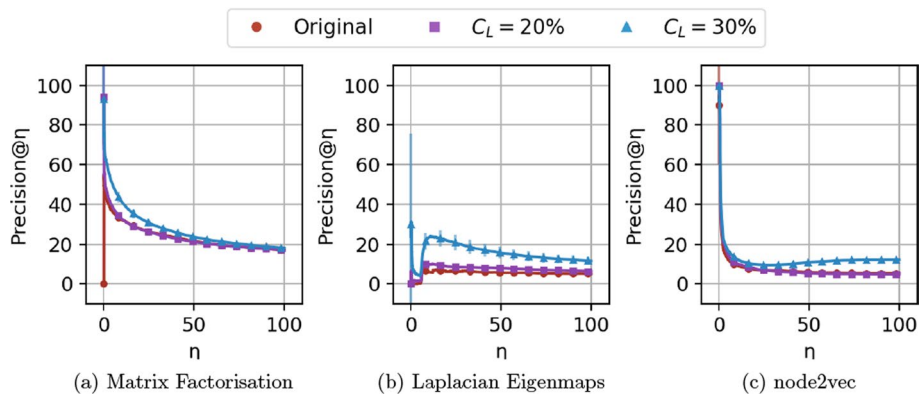


Fig. 10 *Precision@*η obtained for network reconstruction, in average, applying each method to each version of the *Hamsterster* data set (original and rewired). η is expressed in percentage

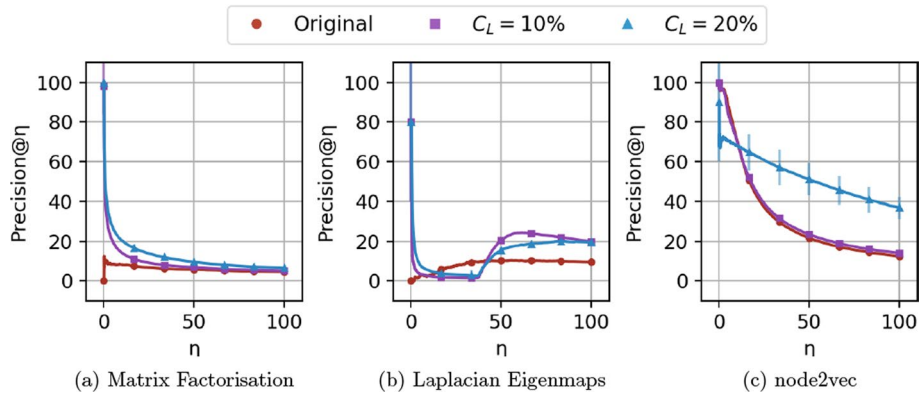


Fig. 11 *Precision@η* obtained for network reconstruction, in average, applying each method to each version of the Maayan-vidal data set (original and rewired). η is expressed in percentage

Acknowledgements
Not applicable.

Author contributions
All authors designed the research and wrote the manuscript. OR, XZ and HW analysed the results. OR performed the experiments and prepared all the figures and tables. All authors read and approved the final manuscript.

Funding
We thank NExTWORKx, a collaboration between TU Delft and KPN on future telecommunication networks, for the support.

Availability of data and materials
The datasets used are publicly available. More information can be found in the corresponding references.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Faculty of Electrical Engineering, Mathematics, and Computer Science, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands. [2]Research Center for Complexity Sciences, Hangzhou Normal University, Hangzhou 311121, People's Republic of China.

Received: 3 February 2022 Accepted: 8 May 2022
Published online: 03 June 2022

References

Albert R, Barabási A-L (2002) Statistical mechanics of complex networks. Rev Mod Phys 74:47–97. https://doi.org/10.1103/RevModPhys.74.47
Alstott J, Klymko C, Pyzza PB, Radcliffe M (2018) Local rewiring algorithms to increase clustering and grow a small world. J Complex Netw 7(4):564–584. https://doi.org/10.1093/comnet/cny032
Asikainen A, Iñiguez G, Ureña-Carrión J, Kaski K, Kivelä M (2020) Cumulative effects of triadic closure and homophily in social networks. Sci Adv 6(19):7310. https://doi.org/10.1126/sciadv.aax7310
Barabási A-L, Albert R (1999) Emergence of scaling in random networks. Science 286(5439):509–512. https://doi.org/10.1126/science.286.5439.509
Belkin M, Niyogi P (2001) Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Proceedings of the 14th international conference on neural information processing systems: natural and synthetic. NIPS'01. MIT Press, Cambridge, pp 585–591. https://doi.org/10.7551/mitpress/1120.003.0080
Bruch EE, Newman MEJ (2018) Aspirational pursuit of mates in online dating markets. Sci Adv 4(8):9815. https://doi.org/10.1126/sciadv.aap9815

Cao R-M, Liu S-Y, Xu X-K (2019) Network embedding for link prediction: the pitfall and improvement. Chaos (Woodbury, NY). https://doi.org/10.1063/1.5120724
Cui P, Wang X, Pei J, Zhu W (2019) A survey on network embedding. IEEE Trans Knowl Data Eng 31(5):833–852. https://doi.org/10.1109/TKDE.2018.2849727
da Fontoura CL, Travieso G, Rodrigues FA, Boas PRV, Antiqueira L, Viana MP, Rocha LEC (2011) Analyzing and modeling real-world phenomena with complex networks: a survey of applications. Adva Phys 60(3):329–412. https://doi.org/10.1080/00018732.2011.572452
Epasto A, Perozzi B (2019) Is a single embedding enough? learning node representations that capture multiple social contexts. In: The world wide web conference. WWW '19. Association for Computing Machinery, New York, pp 394–404. https://doi.org/10.1145/3308558.3313660
Feng X, Zhao JC, Xu K (2012) Link prediction in complex networks: a clustering perspective. Eur Phys J B 85(1):3. https://doi.org/10.1140/epjb/e2011-20207-x
Fortunato S (2010) Community detection in graphs. Phys Rep 486(3):75–174. https://doi.org/10.1016/j.physrep.2009.11.002
Foster DV, Foster JG, Grassberger P, Paczuski M (2011) Clustering drives assortativity and community structure in ensembles of networks. Phys Rev E 84:066117. https://doi.org/10.1103/PhysRevE.84.066117
Girvan M, Newman MEJ (2002) Community structure in social and biological networks. Proc Natl Acad Sci 99(12):7821–7826. https://doi.org/10.1073/pnas.122653799
Grover A, Leskovec J (2016) Node2Vec: scalable feature learning for networks. In: Proceedings of the 22Nd ACM SIGKDD international conference on knowledge discovery and data mining. KDD '16. ACM, New York, pp 855–864. https://doi.org/10.1145/2939672.2939754
Hanley JA, McNeil BJ (1982) The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology 143(1):29–36. https://doi.org/10.1148/radiology.143.1.7063747 (**PMID: 7063747**)
Herlocker JL, Konstan JA, Terveen LG, Riedl JT (2004) Evaluating collaborative filtering recommender systems. ACM Trans Inf Syst 22(1):5–53. https://doi.org/10.1145/963770.963772
Khosla M, Setty V, Anand A (2021) A comparative study for unsupervised network representation learning. IEEE Trans Knowl Data Eng 33(5):1807–1818. https://doi.org/10.1109/TKDE.2019.2951398
Koren Y (2008) Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining. KDD '08. Association for Computing Machinery, New York, pp 426–434. https://doi.org/10.1145/1401890.1401944
Koren Y, Bell R, Volinsky C (2009) Matrix factorization techniques for recommender systems. Computer 42(8):30–37. https://doi.org/10.1109/MC.2009.263
Kotu V, Deshpande B (2019) Chapter 11—Recommendation engines. In: Kotu V, Deshpande B (eds) Data science, 2nd edn. Morgan Kaufmann, Cambridge, pp. 343–394. https://doi.org/10.1016/B978-0-12-814761-0.00011-3
Kovács IA, Luck K, Spirohn K, Wang Y, Pollis C, Schlabach S, Bian W, Kim D-K, Kishore N, Hao T, Calderwood MA, Vidal M, Barabási A-L (2019) Network-based prediction of protein interactions. Nat Commun 10(1):1240. https://doi.org/10.1038/s41467-019-09177-y
Kunegis J (2013) KONECT: the Koblenz network collection. In: Proceedings of the 22nd international conference on World Wide Web. WWW '13 Companion. Association for Computing Machinery, New York, pp 1343–1350. https://doi.org/10.1145/2487788.2488173
Li C, Wang H, de Haan W, Stam CJ, Mieghem PV (2011) The correlation of metrics in complex networks with applications in functional brain networks. J Stat Mech Theory Exp 2011(11):11018. https://doi.org/10.1088/1742-5468/2011/11/p11018
Liao H, Zeng A, Zhang Y-C (2015) Predicting missing links via correlation between nodes. Physica A 436:216–223. https://doi.org/10.1016/j.physa.2015.05.009
Liben-Nowell D, Kleinberg J (2003) The link prediction problem for social networks. In: Proceedings of the twelfth international conference on information and knowledge management. CIKM '03. Association for Computing Machinery, New York, pp 556–559. https://doi.org/10.1145/956863.956972
Liu L, Qu B, Chen B, Hanjalic A, Wang H (2018) Modelling of information diffusion on social networks with applications to wechat. Physica A 496:318–329. https://doi.org/10.1016/j.physa.2017.12.026
Lü L, Zhou T (2011) Link prediction in complex networks: a survey. Physica A 390(6):1150–1170. https://doi.org/10.1016/j.physa.2010.11.027
Newman MEJ (2010) Networks: an introduction. Oxford University Press, Oxford. https://doi.org/10.1080/0022250X.2012.744247
Newman MEJ (2003) The structure and function of complex networks. SIAM Rev 45:167–256. https://doi.org/10.1137/S003614450342480
Orman K, Labatut V, Cherifi H (2013) In: Menezes R, Evsukoff A, González MC (eds) An empirical study of the relation between community structure and transitivity. Springer, Berlin, pp 99–110. https://doi.org/10.1007/978-3-642-30287-9_11
Ostroumova L, Ryabchenko A, Samosvat E (2013) Generalized preferential attachment: tunable power-law degree distribution and clustering coefficient. In: Bonato A, Mitzenmacher M, Prałat P (eds) Algorithms and models for the web graph. Springer, Cham, pp 185–202. https://doi.org/10.1007/978-3-319-03536-9_15
Pastor-Satorras R, Castellano C, Van Mieghem P, Vespignani A (2015) Epidemic processes in complex networks. Rev Mod Phys 87:925–979. https://doi.org/10.1103/RevModPhys.87.925
Peixoto TP (2022) Disentangling homophily, community structure, and triadic closure in networks. Phys Rev X 12:011004. https://doi.org/10.1103/PhysRevX.12.011004
Perozzi B, Al-Rfou R, Skiena S (2014) DeepWalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining. KDD '14. Association for Computing Machinery, New York, pp 701–710. https://doi.org/10.1145/2623330.2623732
Raghavan V, Jung G, Bollmann P (1989) A critical investigation of recall and precision as measures of retrieval system performance. ACM Trans Inf Syst 7:205–229. https://doi.org/10.1145/65943.65945

Rual J-F, Venkatesan K, Hao T, Hirozane-Kishikawa T, Dricot A, Li N, Berriz GF, Gibbons FD, Dreze M, Ayivi-Guedehoussou N, Klitgord N, Simon C, Boxem M, Milstein S, Rosenberg J, Goldberg DS, Zhang LV, Wong SL, Franklin G, Li S, Albala JS, Lim J, Fraughton C, Llamosas E, Cevik S, Bex C, Lamesch P, Sikorski RS, Vandenhaute J, Zoghbi HY, Smolyar A, Bosak S, Sequerra R, Doucette-Stamm L, Cusick ME, Hill DE, Roth FP, Vidal M (2005) Towards a proteome-scale map of the human protein–protein interaction network. Nature 437(7062):1173–1178. https://doi.org/10.1038/nature04209

Saito T, Rehmsmeier M (2015) The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. PLoS ONE 10(3):1–21. https://doi.org/10.1371/journal.pone.0118432

Tandon A, Albeshri A, Thayananthan V, Alhalabi W, Radicchi F, Fortunato S (2021) Community detection in networks using graph embeddings. Phys Rev E 103:022316. https://doi.org/10.1103/PhysRevE.103.022316

Torres L, Chan KS, Eliassi-Rad T, Estrada E (2020) GLEE: geometric laplacian eigenmap embedding. J Complex Netw 8(1):1–17. https://doi.org/10.1093/comnet/cnaa007

Torricelli M, Karsai M, Gauvin L (2020) weg2vec: event embedding for temporal networks. Sci Rep 10(1):7164. https://doi.org/10.1038/s41598-020-63221-2

Van Mieghem P, Wang H, Ge X, Tang S, Kuipers FA (2010) Influence of assortativity and degree-preserving rewiring on the spectra of networks. Eur Phys J B 76(4):643–652. https://doi.org/10.1140/epjb/e2010-00219-x

Wang C, Pan S, Yu CP, Hu R, Long G, Zhang C (2022) Deep neighbor-aware embedding for node clustering in attributed graphs. Pattern Recogn 122:108230. https://doi.org/10.1016/j.patcog.2021.108230

Wang D, Cui P, Zhu W (2016) Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. KDD '16. Association for Computing Machinery, New York, pp 1225–1234. https://doi.org/10.1145/2939672.2939753

Wharrie S, Azizi L, Altmann EG (2019) Micro-, meso-, macroscales: the effect of triangles on communities in networks. Phys Rev E. https://doi.org/10.1103/PhysRevE.100.022315

Winterbach W, Mieghem PV, Reinders M, Wang H, Ridder D (2013) Topology of molecular interaction networks. BMC Syst Biol 7(1):90. https://doi.org/10.1186/1752-0509-7-90

Zhan X-X, Li Z, Masuda N, Holme P, Wang H (2020) Susceptible-infected-spreading-based network embedding in static and temporal networks. EPJ Data Sci 9(1):30. https://doi.org/10.1140/epjds/s13688-020-00248-5

Zhang D, Yin J, Zhu X, Zhang C (2018) SINE: scalable incomplete network embedding. In: Tao D, Thuraisingham B (eds) 2018 IEEE international conference on data mining (ICDM 2018). Proceedings—IEEE international conference on data mining, ICDM. IEEE, Institute of Electrical and Electronics Engineers, New York, pp 737–746. https://doi.org/10.1109/ICDM.2018.00089

Publisher's Note