Systematic Biases in Link Prediction: comparing heuristic and graph embedding based methods

Aakash Sinha[1], Rémy Cazabet[2], and Rémi Vaudaine[3]

[1] Department of Computer Science and Engineering, I.I.T. Delhi, Hauz Khas, New Delhi - 110016, INDIA
[2] Univ Lyon, UCBL, CNRS, LIRIS UMR 5205, F-69621, Lyon, France
remy.cazabet@gmail.com
[3] Univ Lyon, UJM-Saint-Etienne, CNRS, SAINT-ETIENNE, France

Abstract. Link prediction is a popular research topic in network analysis. In the last few years, new techniques based on graph embedding have emerged as a powerful alternative to heuristics. In this article, we study the problem of systematic biases in the prediction, and show that some methods based on graph embedding offer less biased results than those based on heuristics, despite reaching lower scores according to usual quality scores. We discuss the relevance of this finding in the context of the *filter bubble* problem and the *algorithmic fairness* of recommender systems.

Keywords: graph embedding, link prediction, systematic biases, filter bubble

1 Introduction

Graph data occurs in various real-world applications such as social networks, biological networks, communication networks and many more.

In this article, we focus on the problem of *link prediction* (LP). LP is a classic problem on graph data, with numerous applications, from the identification of missing data to recommender systems. Several surveys have been written on the topic [2, 18, 19, 26]. These articles have notably discussed the best way to evaluate and compare LP methods, and have studied the properties, advantages and drawbacks of different approaches. Studied approaches can be **unsupervised** or **supervised**, and be based on **heuristics** (e.g. the number of common neighbors, Adamic Adar index, etc.) or other approaches (block modeling, random walks, etc.).

Graph embedding is a new technique that gained significant popularity amongst the research community in recent years. Embeddings convert graph data into vectors, creating a representation of nodes in a lower dimensional vector space, on top of which various prediction and detection algorithms can be applied. These embeddings can be used, for instance, in node classification [6, 14], community detection [12, 24] and role detection [22]. Recent surveys [9, 14, 11] present in details the rationale, the different methods and applications of graph embeddings.

In recent years, several articles (e.g.,[24, 15, 14]) have proposed to use **graph embedding** for link prediction. Such papers often claim that these approaches outperform

the state of the art. It must be noted, however, that these articles are focused on proposing new embedding approaches, and do not proceed to in-depth evaluation. In particular, they often do not use the quality scores recommended in the literature, and do not investigate the results behind computing such a score.

This article is organized in three parts

1. Using a rigorous evaluation framework, we compare quantitatively recent approaches based on graph embeddings on the task of link prediction with earlier methods
2. We evaluate systematic biases introduced by each approach according to three aspects: the distance in the graph, the degrees of nodes and the community structure
3. We discuss the potential effect of the observed biases in the context of the *filter bubble* problem and the *algorithmic fairness* of recommender systems.

2 Link Prediction Evaluation Framework

In this section, we define a rigorous framework to compare link prediction methods. According to our experience, many recent articles on link prediction do not specify precisely their experimental sections, leading to great difficulties in reproducing the reported results. We split the evaluation framework into three independent steps, each described in the corresponding section:

1. Creation of training and test sets
2. Link Prediction
3. Evaluation using an appropriate score function

2.1 Creation of learning and prediction sets

To evaluate link prediction methods, one starts from a network dataset, and split it into a **learning set** and a **prediction set**, even if the method to test is considered as unsupervised. The learning set is considered the *current state* of the network, based on which predictions are made. The prediction set is considered as the *future* evolution of the network, i.e. the list of edges that will appear and that should be predicted. In this article as in most of the literature, we focus only on *added* edges, and not *removed* ones.

Note that since methods are *supervised*, they need to split the learning set into a training and test set in order to learn where edges should appear. This is independent of the described distinction between learning and prediction set. In real applications, the input is an observed graph –that can be static or dynamic– and that graph corresponds to our learning set.

Static graph If the original dataset used for evaluation is a single, static graph, we remove randomly a fraction of edges of a given size. These removed edges constitute the **prediction set**. Following a common practice [14], we ensure that the resulting network is composed of a single connected component, using the following procedure:

- Remove randomly the desired number of edges
- Conserve only the largest component of the resulting graph

Dynamic graph For dynamic graphs, the order of edge apparition is known. A date is chosen to split the dataset in two: all edges appearing before or at the chosen date constitute the learning set, and all those appearing after the date constitute the prediction set. If the original data contains information on edges that disappear, this information is ignored. To stay coherent with the static case, the single connected component constraint also applies: in the learning set graph, only the largest connected component is conserved.

In our experiments, the learning set is composed of **80%** of the original dataset.

2.2 Link Prediction

A classifier is trained from examples to recognize the pairs of nodes that are the most likely to see edges appear between them. Examples are composed of 50% of *positive* examples (i.e. examples of pairs of nodes that are connected by a link) and 50% of *negative* examples (i.e. pairs of nodes that are **not** connected by a link). Both positive and negative examples are picked randomly from the learning set graph. In this article, we pick $\frac{1}{4}$ of edges in the learning set as positive examples, and an equal number of non-edges as negative examples.

The trained classifier could be used to predict, given an unseen pair of nodes, if it should be connected by an edge or not (Yes/No prediction). However, this does not make sense in realistic settings, since we often want to answer questions such as *which edges might appear in the next hours/day/month ?* or *What are the three most likely edges to appear connecting node n_x ?*. As a consequence, following previous works [2, 15], we assign to each pair of nodes a score corresponding to the *decision function* of the classifier for this sample.

For each experiment, we run the link prediction process 5 times and report the average and standard deviation values.

2.3 Choice of an appropriate score function

To evaluate the quality of the prediction provided by a link prediction method, it is necessary to use a relevant score function.

It must be noted that link prediction is characterized by an extreme class imbalance. On realistic networks of large size, the *density* is low ($<0.001\%$), thus the number of edges that will appear is much lower than the number of edges that will not appear.

To evaluate a model correctly, the prediction should be made on all possible $|V| * |V| - |E|$ edges. Since this test size can be very large for real graphs, we need to reduce the size of the test set by taking a random sample without introducing any or minimal bias. In this article, we fix a sample size of 500 000. Two important observations must be made about the constitution of this sample:

- Most scores yield very different results according to the fraction of negative examples in the sample. Unlike for classifier training, it is thus essential[26] to keep its original ratio, and not take a 50%/50% sample, which is highly unrealistic.

– Due to the extreme imbalance, the number of positive examples can be extremely low even in a large sample, leading to noise in the results. Therefore we fix a lower value on the number of positive samples (10 in this article).

Two metrics, the Average Precision (AP) and the Area Under Receiver Operating Characteristic curve (AUROC) have been identified [26, 19] to be relevant for this task. AUROC has the property of being independent of the fraction of positive examples in the test set, while AP is favored by some because it gives a higher importance to the first few predictions, that are the most useful in link prediction settings.

3 Methods evaluation

In this article, we present results on three graphs previously used in articles on graph-embedding techniques.

We selected large graphs, the first two being static and the last one dynamic. These graphs are FACEBOOK [17], ASTROPH [16] and VK [24]. Due to space constraint, the reader can refer to their original descriptions in the referenced articles.

3.1 Method based on Heuristics

The heuristics used in all experiments are Common Neighbors, Adamic Adar, Preferential attachment, Jaccard Coefficient, nodes degree (for both endpoints) and Resource allocation index (Table 1).

Heuristics can be used for unsupervised prediction if a single of them is used: the score of the heuristic is used directly to rank pairs of nodes. They can also be used for supervised prediction by using some or all of them as features, together with a classifier. In this article, we use all heuristics together and a logistic classifier (implementation of sklearn). This supervised approach has been shown to give the best results [1].

Heuristics	Definition
Common Neighbors	$\lvert\Gamma(u)\cap\Gamma(v)\rvert$
Adamic Adar	$\sum_{w\in\Gamma(u)\cap\Gamma(v)}\frac{1}{\log\lvert\Gamma(w)\rvert}$
Preferential attachment	$\lvert\Gamma(u)*\Gamma(v)\rvert$
Jaccard Coefficient	$\frac{\lvert\Gamma(u)\cap\Gamma(v)\rvert}{\lvert\Gamma(u)\cup\Gamma(v)\rvert}$
Resource allocation index	$\sum_{w\in\Gamma(u)\cap\Gamma(v)}\frac{1}{\lvert\Gamma(w)\rvert}$

Table 1: Heuristic scores for Link prediction. u,v represent the nodes, $\Gamma(u)$ represents the set of neighbors of node u.

3.2 Methods based on Graph embeddings

Different embedding algorithms preserve different properties (first/second/higher order proximities) and capture different features. The four embeddings we test are:

- Laplacian Eigenmaps (LE) [4]
- High-Order Proximity preserved Embedding (HOPE)[20]
- node2vec[15]
- Versatile Graph Embeddings from Similarity Measures (VERSE)[24]

Different operators are used to combine the vectors of nodes into edge vectors: *Hadamard* for LE and VERSE, *Normalized Hadamard* for HOPE and node2vec These operators are those observed by the authors to yield the best results. We use embeddings in 128 dimensions, as is a common practice in the literature.

Due to space constraints, we refer the reader to the original article for each method detailed description. We have chosen these methods due to their frequent use and good results observed in the recent literature, e.g., [14]. We have used the most common parameters. In particular for node2vec, we have systematically tested with parameter sets (p=4,q=0.5), (p=0.5, q=4), (p=1,q=1) and we report the highest score.

3.3 Results

Each method has been tested on each graph. Results are summarized in Table 2. We can observe that, compared with state of the art supervised heuristics approaches, graph embeddings do not yield clear better results. Using the AP score, heuristics always perform best. Using the ROC score, the VERSE algorithm is the only algorithm to obtain higher results.

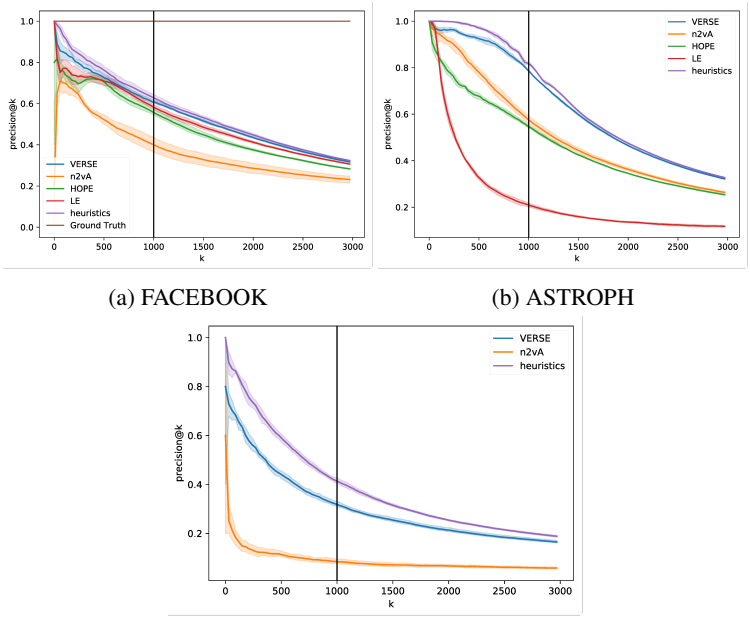
Graph	Method	Heuristics	LE	HOPE	n2v	VERSE
AP	FACEBOOK	**0.74**	0.50	0.68	0.44	0.60
	ASTROPH	**0.79**	0.06	0.43	0.22	0.48
	VK	**0.063**	-	-	0.006	0.021
ROC	FACEBOOK	**0.995**	0.994	0.981	0.988	**0.995**
	ASTROPH	0.988	0.922	0.943	0.973	**0.992**
	VK	0.87	-	-	0.77	**0.89**

Table 2: Results for each method on each graph. Results are significant (i.e. Variance is inferior to reported precision.

To confirm this result, we plot the *precision@k* score. In many applications, only the first few predictions are used, thus the relevance of this analysis.

To perform this experiment, we select randomly pairs of nodes that are not linked in the training set until we obtain 1000 positive examples, i.e., edges that do appear according to the ground truth. which allows us to preserve a balance between positive and negative examples coherent with the dataset.

Results are plotted in fig. 1. We observe that results are coherent with the scores obtained, i.e., heuristics and VERSE tend to give the best scores both for the first few predictions and for a realistic number of predictions (1000, corresponding to the number of real positive examples in the sample). In some settings, some algorithms do not yield useful predictions after the first few hundreds one (LE for ASTROPH dataset, node2vec for VK).



(a) FACEBOOK (b) ASTROPH

(c) VK

Fig. 1: Precision@k

4 Analysis of systematic biases

New edges in a network can appear for different reasons, and between nodes with different properties. For instance, in a social network such as Twitter, a new follower relationship might be due to two close friends following each other, to a user following a raising celebrity they recently discovered, to strangers connecting because they think they have a common topic of interest, etc.

A link prediction algorithm might have a tendency to predict some types of links more than others, a phenomenon that we call *systematic bias*. In this article, we focus on biases induced by the network topology. More particularly, we focus on three types of biases:

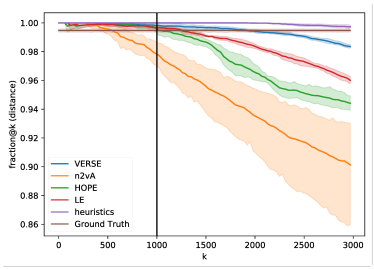
- Graph distance
- Node degree
- Community structure

To evaluate the biases, we study the evolution of the fraction of new edges verifying a given property. Since predicted edges are ordered, from most probable to less probable, we define the $fraction@k$, corresponding to the ratio of pairs of nodes satisfying this property among the k most likely edges. The dataset is selected as previously explained for the $precision@k$.

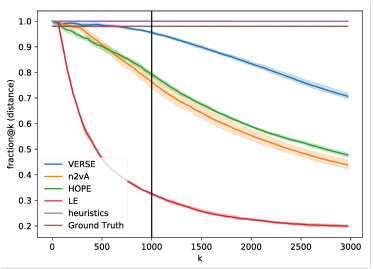
We define the reference value of $fraction@k$ as the value among all edges that do appear in the ground truth (positive examples in the test set). In the scenario of a perfect

prediction, the $fraction@k$ curve should follow the ground truth scenario until 1000 (corresponding to the real number of observed edges in the test sample), and then move towards the value corresponding to the whole dataset (all pairs of nodes not linked in the training set).

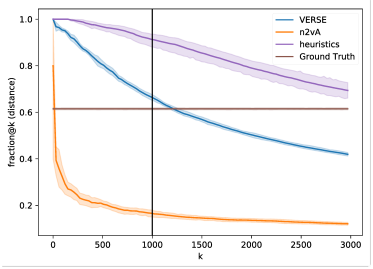
4.1 Graph distance



(a) FACEBOOK



(b) ASTROPH



(c) VK

Fig. 2: Ratio@k for the graph distance.

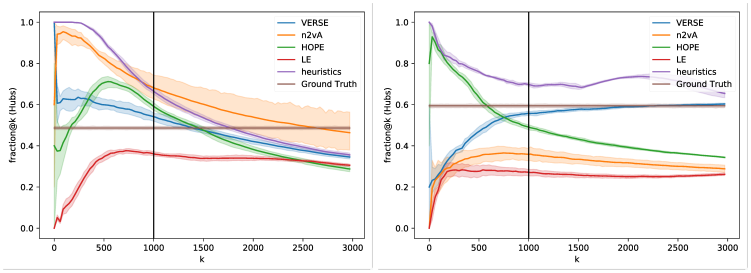
Edges can appear between nodes that were *close* or *far* in the graph in term of graph distance, i.e. length of the shortest path between them. It has been observed that most edges appear between nodes at a short distance, a phenomenon often called *triangle closure*. It is intuitively known in social network analysis by the saying ”friends of my friends are my friends”. Since the number of edges appearing at a distance more than two is usually very low, we consider only two cases:

- Short distance link: the new edge appear between nodes that were previously at distance two in the graph.
- Long distance link: the new edge appear between nodes that were previously at distance three or more.

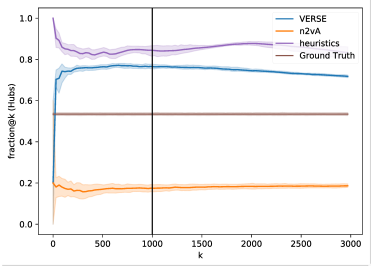
Fig. 2 presents the $fraction@k$ of short distance link for the different methods. We can make the following observations:

– In the ground truth, most edges appear between nodes at distance 2, although the value is much lower for VK dataset.
– The Heuristic-based approach is highly biased towards predicting short distance links.
– Most other approaches tend to be biased towards short distance links in the first (most probable) predictions, this value later decreases and the fraction often becomes lower than expected when the expected number of edges (1000) is reached.

4.2 Node degree



(a) FACEBOOK (b) ASTROPH



(c) VK

Fig. 3: Fraction@k for High degree nodes (hubs)

Most real networks have heterogeneous degree distributions, that can often be approximated by scalefree distributions [3]. In those networks, there is a small fraction of nodes of high degrees that concentrate most of the edges. We define this class of nodes, called *Hubs*, as the 10% of nodes of highest degrees.

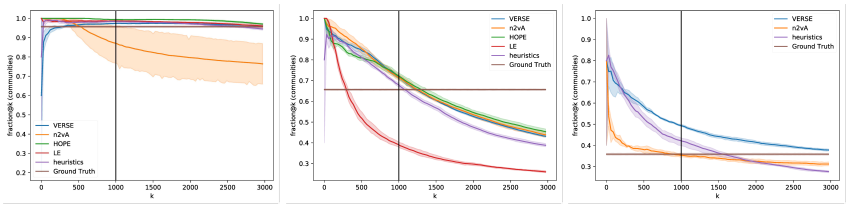
Fig. 3 presents the $fraction@k$ of new links that have at least a Hub among their endpoints.

We can make the following observations:

– In the three studied datasets, the $fraction@k$ is between 0.5 and 0.6
– The Heuristic-based approach is highly biased towards high $fraction@k$, i.e. predicting too many edges involving hubs.

– One method seems to be clearly biased towards underestimation (LE), most other methods do not have clear tendencies.
– The VERSE method is the closest to the Ground Truth at the realistic threshold (1000) in all 3 settings.

4.3 Community structure



(a) FACEBOOK (b) ASTROPH (c) VK

Fig. 4: Fraction@k for the community structure

Most real networks have mesoscale structures known as communities[13]. The field of community detection and analysis is very active, and there is an intricate relationship between link prediction and community structure[10]. Because communities are dense groups of nodes, edges are more likely to appear inside communities than between them, a property that can be used both for link prediction and for community discovery in dynamic settings[23].

We therefore distinguish between two types of edges, those appearing inside communities and those appearing between them.

To discover the community structure, we apply the Louvain algorithm [7], which is the most often used in the literature. This algorithm is based on Modularity optimization, thus, by definition, discover communities denser than the rest of the network. Note that the problem could be studied in more details using overlapping communities, that can introduce different effects, such as higher probabilities of edges on the overlapping parts [25]

Fig. 4 presents the $fraction@k$ of new links that appear inside communities, as found by the Louvain Algorithm.

We can make the following observations:

– In different datasets, the fraction of edges in the ground truth appearing inside communities varies widely.
– The Heuristic-based approach is systematically biased towards high $fraction@k$, i.e. predicting too many edges inside communities.
– Most methods (except n2v in FACEBOOK and LE in ASTROPH) are also biased towards overprediction of internal edges.
– The bias is particularly strong among the first few prediction

5 Discussion: effect of biases on recommender systems

In recent years, both the scientific community and the civil society have started to pay attention to the problem of biased results provided by machine learning algorithms in real-life applications. Two problems have particularly been identified:

- The filter bubble phenomenon is known to occur at least in two settings: in Online Social Networks such as Facebook and Twitter, and on web search engines such as Google or Bing. The problem comes from a reinforcement phenomenon: machine learning algorithms learn the preferences of users, and show them more results according to these preferences. As a consequence, users see results that are less diverse than they should, but rather biased towards the opinions or interests that the algorithm has inferred they had in the beginning. This process has been accused to amplify the *polarization* phenomenon in political and social opinion [21].
- The problem of fairness of algorithms [5] arrises in many real-life applications of machine learning algorithms. In its most famous occurrences, a decision is taken by an algorithm that will impact individuals (who gets a loan, a job, is accepted at a university, is put in a list of potential terrorists, etc.), but the decisions taken by the algorithms are unfair towards a category of people. Typically, the algorithm learns that an ethnic or social group –or, if not present in the data, another highly correlated attribute such as locations or name– correlates with unfavorable outcomes, and therefore learns to discriminate based on this property.

These two problems are now widely known, and solutions have been proposed to mitigate them[8], in particular by ensuring the preservation of some identified properties.

The systematic biases highlighted in the present article however have several interesting features:

- They arise without labeled data, but simply due to the network structure
- The problem is not that unwanted problematic correlations are preserved, but rather that some network properties that were not explicitly required to be preserved are lost.

We can note that the efficiency of the link prediction is not necessarily linked to the absence of bias, on the contrary, methods based on heuristics clearly yield the best results in term of overall link prediction compared with recent techniques based on embeddings. However, in term of biases, we have observed that heuristics usually suffer from the highest biases. Furthermore, those biases are constantly of the same types: they tend to favor nodes at a short distance in the graph, large nodes, and nodes belonging to the same communities. In other terms, these biases can be associated with a **loss in diversity**. In a social network, recommendations of new contacts will favor exaggeratedly either the most similar profiles (same community, closer distance) or the most popular profiles (hubs), but will fail to identify more nuanced possibilities that would add diversity. In product or content recommendation, an algorithm based on link prediction with heuristics will propose the items the most similar to those already consumed, or the most consumed overall, ignoring the relevant possibilities in-between.

New methods based on embeddings might be part of the solution to this problem, as some of them tend to reduce those biases. In particular, it seems that the recent VERSE algorithm, that offers the best overall results among embedding techniques for link prediction accord to scores and *precision@k*, also consistently show lesser biases than previous techniques.

On top of that, one could follow techniques already proposed[8] to fight against the fairness of algorithm issue, for instance by training separately for the different properties to preserve (e.g., training different classifiers to predict edges at distance 2 and at distance 3 or more).

6 Conclusion

We have shown that the most widely used techniques for link prediction suffer from systematic biases towards some network properties, and that those biases might play a role in the filter bubbles and algorithmic fairness problems.

We have also identified that some of the most recent techniques proposed based on graph embedding are able to somewhat reduce that problem, although they do not yet manage to improve over heuristic based methods for the overall quality of the link prediction.

We think that this work can be extended in two directions:

- On the one hand, we need to understand where do those biases come from. While it can be intuitively understood for heuristics –the features are not learned but selected according to prior knowledge, and they have been designed with triangle closure and preferential attachment properties in mind– the reason why feature learning based methods have similar biases is, to the best of our knowledge, unknown.
- On the other hand, we must develop link prediction methods adapted to mitigate or eliminate those systematic biases. Ideally, such a method should not only remove the three biased identified in this article, but ensure using statistical methods that the network formed by adding the predicted edges has a similar network structure profile than the original network.

References

1. Al Hasan, M., Chaoji, V., Salem, S., Zaki, M.: Link prediction using supervised learning. In: SDM06: workshop on link analysis, counter-terrorism and security (2006)
2. Al Hasan, M., Zaki, M.J.: A survey of link prediction in social networks. In: Social network data analytics, pp. 243–275. Springer (2011)
3. Barabási, A.L.: Scale-free networks: a decade and beyond. science **325**(5939), 412–413 (2009)
4. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Advances in neural information processing systems, pp. 585–591 (2002)
5. Berk, R., Heidari, H., Jabbari, S., Kearns, M., Roth, A.: Fairness in criminal justice risk assessments: the state of the art. arXiv preprint arXiv:1703.09207 (2017)
6. Bhagat, S., Cormode, G., Muthukrishnan, S.: Node classification in social networks. In: Social network data analytics, pp. 115–148. Springer (2011)

7. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment **2008**(10), P10,008 (2008)
8. Bolukbasi, T., Chang, K.W., Zou, J.Y., Saligrama, V., Kalai, A.T.: Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In: Advances in Neural Information Processing Systems, pp. 4349–4357 (2016)
9. Cai, H., Zheng, V.W., Chang, K.: A comprehensive survey of graph embedding: problems, techniques and applications. IEEE Transactions on Knowledge and Data Engineering (2018)
10. Clauset, A., Moore, C., Newman, M.E.: Hierarchical structure and the prediction of missing links in networks. Nature **453**(7191), 98 (2008)
11. Cui, P., Wang, X., Pei, J., Zhu, W.: A survey on network embedding. IEEE Transactions on Knowledge and Data Engineering (2018)
12. Fortunato, S.: Community detection in graphs. Physics reports **486**(3-5), 75–174 (2010)
13. Fortunato, S.: Community detection in graphs. Physics reports **486**(3-5), 75–174 (2010)
14. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: A survey. Knowledge-Based Systems **151**, 78–94 (2018)
15. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855–864. ACM (2016)
16. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graph evolution: Densification and shrinking diameters. ACM Transactions on Knowledge Discovery from Data (TKDD) **1**(1), 2 (2007)
17. Leskovec, J., Mcauley, J.J.: Learning to discover social circles in ego networks. In: Advances in neural information processing systems, pp. 539–547 (2012)
18. Lichtenwalter, R.N., Lussier, J.T., Chawla, N.V.: New perspectives and methods in link prediction. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 243–252. ACM (2010)
19. Lü, L., Zhou, T.: Link prediction in complex networks: A survey. Physica A: statistical mechanics and its applications **390**(6), 1150–1170 (2011)
20. Ou, M., Cui, P., Pei, J., Zhang, Z., Zhu, W.: Asymmetric transitivity preserving graph embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1105–1114. ACM (2016)
21. Pariser, E.: The filter bubble: What the Internet is hiding from you. Penguin UK (2011)
22. Ribeiro, L.F., Saverese, P.H., Figueiredo, D.R.: struc2vec: Learning node representations from structural identity. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 385–394. ACM (2017)
23. Rossetti, G., Cazabet, R.: Community discovery in dynamic networks: a survey. ACM Computing Surveys (CSUR) **51**(2), 35 (2018)
24. Tsitsulin, A., Mottin, D., Karras, P., Müller, E.: Verse: Versatile graph embeddings from similarity measures. In: Proceedings of the 2018 World Wide Web Conference on World Wide Web, pp. 539–548. International World Wide Web Conferences Steering Committee (2018)
25. Yang, J., Leskovec, J.: Community-affiliation graph model for overlapping network community detection. In: 2012 IEEE 12th International Conference on Data Mining, pp. 1170–1175. IEEE (2012)
26. Yang, Y., Lichtenwalter, R.N., Chawla, N.V.: Evaluating link prediction methods. Knowledge and Information Systems **45**(3), 751–782 (2015)