Efficient Node PageRank Improvement via Link-building using Geometric Deep Learning

VINCENZA CARCHIOLO, MARCO GRASSIA, ALESSANDRO LONGHEU, MICHELE MALGERI, and GIUSEPPE MANGIONI, Department of Electrical, Electronic and Computer Engineering, University of Catania

Centrality is a relevant topic in the field of network research, due to its various theoretical and practical implications. In general, all centrality metrics aim at measuring the importance of nodes (according to some definition of importance), and such importance scores are used to rank the nodes in the network, therefore the rank improvement is a strictly related topic. In a given network, the rank improvement is achieved by establishing new links, therefore the question shifts to which and how many links should be collected to get a desired rank. This problem, also known as *link-building* has been shown to be NP-hard, and most heuristics developed failed in obtaining good performance with acceptable computational complexity.

In this article, we present *LB–GDM*, a novel approach that leverages Geometric Deep Learning to tackle the link-building problem. To validate our proposal, 31 real-world networks were considered; tests show that *LB–GDM* performs significantly better than the state-of-the-art heuristics, while having a comparable or even lower computational complexity, which allows it to scale well even to large networks.

CCS Concepts: • **Mathematics of computing → Graph theory**; • **Computing methodologies → Network science**; **Supervised learning;**

Additional Key Words and Phrases: Link-building, best attachment, Machine Learning, ranking, Graph Attention Network, PageRank, complex networks

ACM Reference format:
Vincenza Carchiolo, Marco Grassia, Alessandro Longheu, Michele Malgeri, and Giuseppe Mangioni. 2023. Efficient Node PageRank Improvement via Link-building using Geometric Deep Learning. *ACM Trans. Knowl. Discov. Data.* 17, 3, Article 38 (February 2023), 22 pages.
https://doi.org/10.1145/3551642

1 INTRODUCTION

The increased computational capabilities opened new frontiers to the study of large-scale phenomena and also allows researchers to adopt a different point of view to analyze traditional problems. From this perspective, the use of network representation to model interactions among

All the authors contributed equally to this research.

This work has been partially supported by the project of University of Catania PIACERI, *PIAno di inCEntivi per la Ricerca di Ateneo.*

Authors' address: V. Carchiolo, M. Grassia, A. Longheu, M. Malgeri, and G. Mangioni, Department of Electrical, Electronic and Computer Engineering, University of Catania, V.le A. Doria, 6, Catania 95125, Italy; emails: {vincenza.carchiolo, marco.grassia, alessandro.longheu, michele.malgeri, giuseppe.mangioni}@unict.it.

1556-4681/2023/02-ART38 $15.00
https://doi.org/10.1145/3551642

several entities has quickly grown [44], and helps in understanding and managing different types of network-based phenomena, as in computer communications, transport infrastructures, online social systems, metabolic reactions, financial markets, and many others [61].

In many of these application scenarios, an important question is how to assess the importance of a node or, in other words, how to define a measure of its centrality. The definition of centrality measures is an important topic in the field of network research due to its various theoretical and practical implications. There are different measures of centrality depending on the point of view, the context, and the meaning of importance that we want to attribute to a node [26]. For instance, if we are looking for brokers then *betweenness centrality* [22] could be the best metric, whereas if the target is prestige the *in-degree* can be used, while *out-degree* is a good metric for activity. The eigenvector-based approach *PageRank* [78] is a widely used metric for powerful collaborative partners centrality [45].

In general, all the centrality measures try to answer the following basic question: *how to rank the nodes (whatever the node represents) according to their position in the network?* When rank is used, regardless of the reason, the improvement of the position is also a topic of large interest. In fact, a better positioning of the node in the network, with respect to a specific centrality metric, corresponds to a greater importance of the node in the modeled system, which translates, in many real-world contexts, into some form of advantage over the other nodes.

For example, Google used to rank the web pages via the PageRank metric to determine the order in which the results of a search were displayed. In a social context, the centrality of a node can be used to determine the ability of a person to influence others with his/her own opinions. In the international trade exchanges, a country that is more *central* than others plays a role of strategic importance in the market. It is therefore natural that in some contexts a node (i.e., the entity behind that it models) wants to increase its rank.

In general, rank improvement strongly depends on the metric behind the ranking and usually comes with a significant cost (e.g., time spent, money) that depends on many factors, so the question of what strategy a node should adopt in order to improve its rank arises. For example, in a social network, one might think of improving the rank of a given person by establishing new relationships (links) with other people (nodes). But then, two other questions would follow: "who is it better to establish new relationships with?" and "how many new relationships are needed to reach a given (target) rank?". The answer to these questions is not trivial and requires the formulation of a solution that involves cost optimization, which is, in some cases, computationally expensive.

Several studies [31, 34, 36, 74–76] focused on the improvement of the PageRank value of nodes when it is employed for the ranking,[1] which is common in the trust context, by creating new incoming links. Such a problem, known as *Link-Building problem* (*Backlink* or *Best in–attachment problem*), has been initially formulated by Olsen in [75]. Informally, given a node x and a rank goal t, it consists in finding the minimum set S of nodes of the network such that if we establish new links from these nodes to x its rank improves to the tth position. A brute force approach that evaluates all the possible subsets of the network's nodes and selects the smaller one is computationally expensive and practically unfeasible even for small networks (as will be detailed later). In fact, it has been shown in [74] that the link-building problem is NP-hard.

To tackle this problem, several heuristics that exploit domain knowledge to produce suboptimal but acceptable solutions have been proposed. However, there is still plenty of room for improvement as most of them are far from optimal and still are computationally expensive. We refer the reader to see [36] for a survey.

[1]In the following, we assign the nodes a rank according to the decreasing value of PageRank, so that the node with the highest value of PageRank is the first in the rank.

In this article, we present a novel approach named (**Link-Building solution based on *GDM*** (***LB–GDM***) that is based on Machine Learning—specifically on Geometric Deep Learning, the new research area that bridges Deep Learning and non-Euclidean data [28]—and that is inspired to the *GDM* framework, proposed in [53] and used to solve the network dismantling problem. In particular, we employ the *Graph Attention Network* [90] layers, whose capabilities have been extensively demonstrated in the literature [62]. One of the most important advantages of our method is that it scales well even to large networks thanks to its very low computational complexity.

The contribution of this article is not limited to a novel solution for the link-building problem, as the proposed methodology can be easily extended to other similar problems, such as those involving other centrality metrics.

To validate our proposal, we tested it on 31 real-world networks and analyzed the results in terms of performance, finding that our approach performs significantly better than the state-of-the-art heuristics while its computational complexity is comparable or even lower.

The article is organized as in the following. Section 2 discusses the state-of-the-art concerning the link-building problem, while Section 3 formally introduces the problem. We describe our proposed method, how it works, and its computational complexity in Section 4, whereas Section 5 illustrates the real-world networks used as test sets, the experiments carried out with such datasets, and the results. Finally, Section 6 provides some concluding remarks and ideas for future developments.

2 RELATED WORKS

The basic idea of in-linking dates back to the PageRank algorithm [78], introduced in 1998 by Google's founders Larry Page and Sergey Brin to assess the relevance P of a web page by exploiting links towards it coming from other pages, and still inspiring Google search engine [27]. Google itself describes the algorithm in [42] as follows: *PageRank works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more links from other websites.*

Many studies on PageRank exist, dealing with its computational issues or limitations [24, 25, 30, 57, 84] or aiming at introducing variations (optimizations or extensions) to the original formula [31, 43, 46, 94]. PageRank is still an important metrics, and it is currently applied in various areas related to document search, such as

- in web search, to display pages that are relevant to a query issued by users [86];
- in the implementation of the *Who to follow* service in Twitter [55], based on shared interests and common connections;
- in E-learning scenarios, to select useful resources in specific topics [33, 87];
- in cars and humans traffic prediction, through public streets and places ranking [56];
- in a recommendation network, to suggest reliable entities to interact with [38, 40].

In [51], the authors discuss other several domains—from chemistry to sports, literature, neuroscience, and others—where PageRank is employed.

Avrachenkov and Litvak [20] study the effect of PageRank when a given page establishes one or more links to other pages and propose a strategy for an optimal linking acquisition. They also discuss the impact of backlinks modification in some real applications such as the Google ranking of Web pages.

Olsen and others [75, 77] formalize the problem of link-building process, study its complexity and discuss the impact of the topology of the graph on the choice of potential new backlinks. Specifically, they find that link-building is W[1]-hard. Furthermore, they show that this problem is in the class of NP optimization problems that allow polynomial-time approximation algorithms

with approximation ratio bounded by a constant. In [74] the authors highlight that the problem is NP-hard if k is part of the input and present some algorithms for simple cases.

In [35] the authors present a method to achieve the highest rank possible first establishing the desired rank value and assigning a budget to limit the cost while increasing the rank. The process involves two stages: first, the node establishes a certain number of links with other existing nodes, then it tries to increase the rank keeping the cost below the budget as low as possible. However, a couple of questions arise:

— Which new nodes should it connect to?
— Should it preserve existing links or not?
— What are the costs associated with these actions?

In [32, 35] some heuristics to solve the link-building problem are discussed and a solution whose complexity is $O(|V| \cdot \log |V|)$ is proposed (where $|V|$ is the number of nodes in a network). Furthermore, the article presents some results from simulations on random and scale-free networks which highlight that better rank improvement comes by acquiring long-distance in-links whilst human intuition would suggest selecting neighbors. These results have been confirmed in [31] where the authors underline that the long-distance link approach achieves the best tradeoff between cost and increase in rank.

3 BACKGROUND AND FORMULATION

In this section, we first recall the PageRank formulation, then we formalize the link-building problem, and finally, we briefly describe some heuristics and their computational complexity.

3.1 PageRank

Here, we outline the basics of the PageRank algorithm [78].

In general, PageRank can be considered a centrality measure used to score entities in a network. It assumes that a network is described as a graph [24, 68, 91] where the nodes model entities (e.g., agents, people, devices, resources, and so on) and the directed links represent relationships between nodes (e.g., trustworthiness in a social network, or hyperlink in the web). To assign a score to each network's node, the PageRank employs a *random walker* that represents a web surfer. The surfer moves from a page to another by selecting one of its out-going links randomly: each link of a web page (the node) has the same probability $1/k^{out}$ of being followed, where k^{out} is the *out-degree* (the number of out-going links of the node). That is, the probability distribution is uniform. However, to cope with the problem of the so-called *sink nodes* [78]—i.e., those nodes with no out-going links that would trap the random walker—at each step the random walker also has a chance of jumping to a random node (that is, it does not follow any out-link).

To formalize such behavior, let us consider a network represented as a graph $G = (V, E)$ where V is the set of nodes (or vertices) and E is the directed links (edges) among nodes. The number of nodes in the graph is $n = |V|$ while the number of links is $e = |E|$. A link e_{ij} from a node i to a node j is represented by an ordered couple $(i, j) \in E$.

A common and useful way to represent a graph is through its *adjacency matrix* A (of $n \times n$ size), where each A_{ij} is 1 if there is a link going from node i to node j and 0 otherwise.

We also identify with N_i^{in} the in-neighborhood of the node i, i.e., the set of nodes $N_i^{in} = \{j \in V : \exists (j, i) \in E\}$, and with N_i^{out} the out-neighborhood of i, i.e., the set of nodes $N_i^{out} = \{j \in V : \exists (i, j) \in E\}$. In the following we indicate with k_i^{in} and k_i^{out} respectively the in and out degree of a generic node i, i.e., the number of links incoming/outgoing to/from i.

Formally:

$$k_i^{in} = \sum_{j \in N_i^{in}} A_{ji} \qquad k_i^{out} = \sum_{j \in N_i^{out}} A_{ij}. \tag{1}$$

Let us introduce the link matrix L $(n \times n)$ defined as

$$L_{ij} = \begin{cases} 1/k_i^{out} & \text{if } A_{ij} \neq 0 \\ 0 & \text{otherwise} \end{cases}. \tag{2}$$

The *sink vector* H $(n \times 1)$ is defined as

$$H_i = \begin{cases} 1 & \text{if } k_i^{out} = 0 \\ 0 & \text{otherwise} \end{cases}. \tag{3}$$

K is the *personalization vector* of size $1 \times n$. While this vector can be arbitrarily chosen as long as it is stochastic, a common choice is to make its values all equal to N^{-1}. $T = \mathbf{1}_{n \times 1}$ is the *teleportation vector*, where each element is 1.

As described in [78], the *transition matrix* M, used in the associated random walker problem, is derived from the link matrix, the sinks vector, the teleportation vector, and the personalization vector defined above:

$$M = d(L + HK) + (1 - d)\, TK, \tag{4}$$

where $d \in [0, 1]$ is called *damping factor* and, in the original implementation [78], it is set to 0.85.

The random surfer model of the PageRank can be mapped to a Markov chain in which the states are the networks' nodes and the transitions are the links between nodes. As we know from the Markov chain theory, the random walk probability vector at step n can be calculated as

$$P_n = M^{\mathsf{T}} P_{n-1}, \tag{5}$$

the related random walker problem can be calculated as

$$P = \left(\lim_{n \to \infty} M^n \right)^{\mathsf{T}} P_0 = \lim_{n \to \infty} (M^{\mathsf{T}})^n P_0 = M_\infty^{\mathsf{T}} P_0. \tag{6}$$

Each element $P^{(i)}$ of the probability vector is the PageRank value of a node i, whose interpretation is "the probability for the random surfer of being at a node at any given point in time during the walk".

The semantics is that the PageRank algorithm will assign high PageRank values to nodes that would appear more often in a random surfer walk, i.e., nodes with high PageRank are *relevant* nodes that will be more visited by surfers and this ranking came from the link structure of the graph.

Finally, in the following, we assign the nodes a rank according to the decreasing value of PageRank, so that the node with the highest value of PageRank in graph G is the first in the rank. We indicate with $R_G^{(i)}$ the position of the node i in such a list (e.g., $R_G^{(i)} = 1$ if node i has the highest value of PageRank in the graph G).

3.2 The Link Building Problem

The PageRank value of a node in a network depends on the nodes it is connected to. This means that a node x joining the network G can firstly choose its rank t by appropriately selecting the set of nodes in the network to be pointed by (i.e., its in-neighborhood N_x^{in}). As discussed in the previous sections, each link comes with a cost, and ideally the in-neighborhood of x should be the minimum necessary to reach the rank t in order to keep the cost as low as possible.

The *Link Building* problem can be formulated as the problem of finding the minimum set of nodes S[2] such that the $R_{G'}^{(x)} = t$ if we connect nodes in S to x. $G' = (V \cup \{x\}, E \cup (S \times \{x\}))$ is the graph obtained by adding the set of $s = |S|$ directed links from nodes in S to the node x.

Essentially, we establish s new links from the nodes in S in order to have x's rank value equal to t. More formally S is the minimum set such that:

$$S \subseteq V : R_{G'}^{(x)} = t. \tag{7}$$

The link building is an NP-hard problem [76], and, in general, it is computationally challenging to explore all the possible combinations of nodes in order to find the minimum set S. In fact, to compute the best rank of x with s nodes, we need to explore $\binom{n}{s}$ combinations. Note that for each combination we need (1) to compute the PageRank values of all network's nodes and (2) to sort nodes according to the decreasing value of PageRank. Moreover, given a target rank t, we do not know *a-priori* the minimum number of in-neighborhood nodes (i.e., the cardinality s of S) so that x rank $R_{G'}^{(x)} = t$. For this reason, it may be necessary to explore 2^n combinations in the worst case, which makes the computation of the optimal solution in real-life scenarios unfeasible. For instance, a network with only 100 nodes needs more than 10^{30} computations of the PageRank in the worst case. Due to these computational issues, it is necessary to find an approximation algorithm to choose a solution acceptably close to the optimum in a polynomial time.

3.3 State-of-the-art Heuristics

In this subsection, we provide an overview of the state-of-the-art heuristics in the literature [29, 31, 35, 37]. We also report the computational complexity of these heuristics, which often depends on the number of attachments s needed to reach the desired target. For a more in-depth analysis, we refer the reader to the original works referenced.

The link-building heuristics can be divided in problem-agnostic and problem-aware strategies.

3.3.1 Problem-agnostic Strategies. Heuristics that do not exploit any problem-specific knowledge belongs to this category, for instance:

- *Random.* Randomly choose the nodes to get in-links from. This approach is, of course, computationally inexpensive and its computational complexity only depends on the number of steps s needed to converge, i.e., it is $(O(s))$.
- *Degree based.* Nodes are chosen according to their in-degree (*In Degree*) or out-degree (*Out Degree*), or even a combination of the two. This family of strategies is $O(|E|)$, where $|E|$ is the number of edges in the network.
- *Long-distance based.* Nodes farther from the newcomer x are chosen first [31]. If classic Dijkstra's algorithm with Fibonacci heap is used, this approach is

$$O(|V| \cdot log(|V|) \ + \ |E|), \tag{8}$$

  where $|V|$ is the number of nodes in the network.

3.3.2 Problem-aware Strategies. These algorithms exploit specific information, e.g., focus on the node metric used for ranking. While these strategies tend to achieve better results, they usually have higher computational complexity. For instance, the classical link-building problem using PageRank has a final computational complexity with a multiplicative factor of $O(PR)$, where:

$$O(PR) = O(|V|^3), \tag{9}$$

[2]Also called optimal in-attachment set.

Table 1. Summary of Heuristics and their Computational Complexity

Strategy	Name	Complexity
Problem-agnostic	Random	$O(s)$
	Degree based (In- & Out-)	$O(\|E\|)$
	Long distance	$O(\|V\| \cdot log(\|V\|) + \|E\|)$
Problem-aware	Anticipated Value	$O(PR)$
	Current Rank	$O(s) \cdot O(PR)$
	Anticipated Out-degree	$O(s) \cdot O(PR)$
	Future Rank	$O(s) \cdot O(\|V\|) \cdot O(PR)$

if the exact Gauss method is used, or, using the approximated power method:

$$O(PR) = m\,|E|, \tag{10}$$

where m is the number of iterations needed to get a good approximation.

The problem-aware strategies are introduced in the following.

— *Anticipated Value.* This static strategy computes the value of the PageRank used for the ranking at the beginning of the link-building process and picks nodes with higher values first. The PageRank values are not recomputed during the process. Its computational complexity is

$$O(AnticipatedValue) = O(PR). \tag{11}$$

— *Current Rank.* Dynamic version of the *Anticipated Value* heuristic. That is, predictions are computed before any attachment. The computational complexity is

$$O(CurrentRank) = O(s) \cdot O(AnticipatedValue). \tag{12}$$

— *Anticipated Out-degree.* Similar to the *Current Rank* but the ratio between the node centrality metric and the out-degree is used. This heuristic is tailored for the PageRank algorithm as nodes with higher rank values and lower degrees are supposed to transfer most of the centrality value. Its computational complexity is

$$O(AnticipatedOutDegree) = O(s) \cdot O(PR). \tag{13}$$

— *Future Rank.* This strategy moves across local maxima by picking at each attachment step the node that provides the target node x with the maximum PageRank value. The computational complexity of this algorithm is

$$O(FutureRank) = O(s) \cdot O(|V|) \cdot O(PR), \tag{14}$$

which makes this heuristic feasible for small networks only, even if the power method is used to compute the PageRank. In fact, in this case:

$$O(FutureRank) = O(s) \cdot O(|V|) \cdot O(m\,|E|) > O(s) \cdot O(m \cdot\ |V|^2), \tag{15}$$

where s and m depend on the network and are not known *a-priori*, and assuming that the network is sparse but still has at least an edge for each node. Moreover, even if s is negligible and the power method converges early (i.e., m is small), the *FutureRank* still scales as the square of the nodes.

Table 1 summarizes the previously discussed heuristics and their complexity.

4 LB–GDM

In this section, we describe our method, how it works, its computational complexity, the model architecture, the node features used, and the training data.

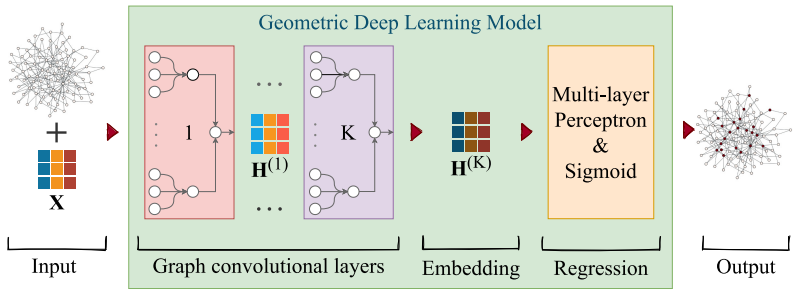

Fig. 1. LB-GDM's model architecture. Our Geometric Deep Learning model architecture as K Graph Neural Network layers (specifically, Graph Attention Network, GAT) coupled with a linear layer each—used as residual connections, not shown in the figure for sake of simplicity—followed by a MLP and a sigmoid activation function that perform regression on the nodes and return, for each, a value between 0 and 1. The model takes as input the network and its node features ($\mathbf{X}$), while $\mathbf{H^{(k)}}$ is the tensor of nodes' embedding for each convolutional layer k.

4.1 Rationale

The key idea behind our approach is to employ Geometric Deep Learning (i.e., Deep Learning on non-Euclidean data) to learn the probability a node i belongs to the optimal in-attachment set S. For this purpose, we use ***Graph Neural Network*** (**GNN**) layers that generalize the convolution operation behind the success of the (classical Deep Learning) **Convolutional Neural Networks** (**CNNs**) to graph-structured data. Specifically, in analogy to their Euclidean parent, Graph Neural Network layers aggregate the (input) user-defined node features with the ones from its neighborhood using a function learned during the training phase. Formally, each layer k generates for every node i some higher-level features (also known as *node embedding*) as

$$\mathbf{h_i^{(k)}} = AGG_k\left(\mathbf{h_i^{(k-1)}}, \left\{\mathbf{h_j^{(k-1)}}, \forall j \in N_i^{in}\right\}\right), \tag{16}$$

where AGG_k is a generic trainable aggregation function, N_i^{in} is the in-neighborhood of node i and $\mathbf{h_i^{(0)}} = \mathbf{x_i}$ are the input node features. The aggregation process can be seen as a message passing process and can be performed multiple times iteratively, i.e., stacking K layers together, the output embedding of a node depends on all of its K-hop neighbors; this effectively captures some local characteristic of the node that can be fed to (Euclidean) Machine Learning algorithms (like Multi-Layer Perceptrons) for, e.g., regression or classification tasks.

4.2 Model Architecture and Complexity

The proposed architecture, summarized in Figure 1, is a stack of a variable number of Graph Neural Network layers—specifically ***Graph Attention Network*** (**GAT**) [54, 90] layers—and a **Multi-Layer Perceptron** (**MLP**) that performs regression and outputs the prediction value p_n for each node n.

Each Graph Neural Network layer is coupled with a linear layer that works as a residual connection (i.e., the input to the layer is fed to both and then outputs are sum together) and the **Exponential Linear Unit** (**ELU**) activation function is applied to the output. After the regression phase, we use a sigmoid activation function so that the p_n values are in the $[0, 1]$ range. The model parameters can be found in Appendix A.

The choice of the Graph Attention Network layers comes with a number of advantages:

— They process the whole neighborhood of each node, effectively aggregating the local information using the so-called *heads*, while many graph convolutional layers perform sampling;
— They learn and assign a relative-importance value to each neighboring node thanks to the attention mechanism borrowed from Natural Language Processing area and used to scale the features incoming from that node;
— Like the multiple filters in the classical convolutional networks, the aggregation phase can be performed multiple times and the results are concatenated or sum together;
— They scale well, considering their low computational complexity—$O(GAT) = O(h(|V| \cdot F \cdot F' + |E| \cdot F'))$ where $|V|$ and $|E|$ are the number of nodes and links in the network, h is the number of *heads*, and F and F' are the number of input and output features respectively—, because the number of operations carried out by the convolution operator depends on the number of links in the network.

Regarding the node features, we employ computationally cheap node features for each node as we extend the ones seen in [53] for the direct case, and do not recompute the features or the predictions during the link-building process. Specifically, we use the in- and out- degree and their ratio, the K−core number, and the local-clustering coefficient. These features are both local (the degrees), second-order (the clustering coefficient), and global (the k−coreness), and provide useful information when propagated to the L-hops neighbors. More in detail, for each node $i \in V$, we compute each feature as follows:

— The in- and out-degree of i, k_i^{in} and k_i^{out}, are computed as in Equation (1), whereas the ratio is computed as $k_{\text{ratio}} = k_{\text{in}}/(k_{\text{out}} + 1)$;
— The local-clustering coefficient of i is computed by using the definition given in [73]:

$$C_i = \frac{\text{number of triangles connected to vertex } i}{\text{number of triples centered on vertex } i}.$$

Informally, it quantifies how close is the neighborhood of i to being a clique (i.e., a complete sub-graph);
— Given a graph G, a k−core is a maximal connected sub-graph of G in which all vertices have a degree at least k [47, 58], the k−coreness of a vertex i is k if it belongs to a k−core but not to a $(k + 1)$−core.

The computational complexity of our approach depends on the Graph Neural Network layers used and on the input features computed during the pre-processing phase:

$$O(\text{LB–GDM}) \approx max\{O(\text{GAT}),\ O(\text{Features})\}.$$

The node features are

$$O(\text{Features}) \approx max\{O(\text{Features})\} = O(K\text{-Core}), \tag{17}$$

where:

$$O(K\text{-core}) = O(|V| + |E|). \tag{18}$$

Moreover, considering that we use a small (and constant) number of features, the computational complexity of GAT layers can be approximated to

$$O(\text{GAT}) \approx O(h(|V| + |E|)). \tag{19}$$

Summing up, the total time complexity of LB–GDM can be approximated to

$$O(\text{LB–GDM}) \approx O(h(|V| + |E|)). \tag{20}$$

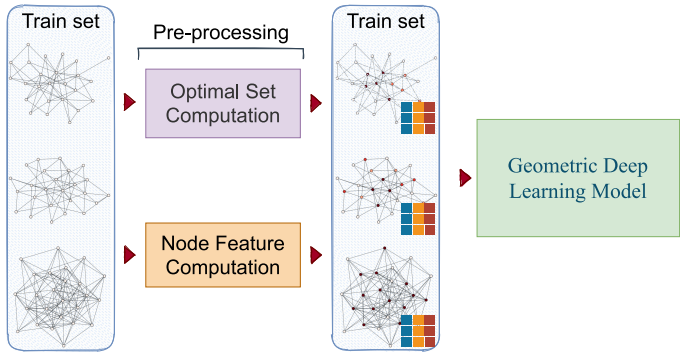

Fig. 2. Dataset generation and training phase. The models are trained on small synthetic networks, where exploring all the solutions to find the optimal one(s) is feasible in a reasonable time. In particular, for the training target, we find the optimal solutions—i.e., the smallest S set(s)—that provide the first position to a newcomer node, and assign to each node i a score p_i equal to the ratio of optimal solutions i belongs to. Indeed, this is performed during a pre-processing step to avoid re-computation during every training.

which can be approximated further to $O(|E|)$ or to $O(|V|)$ for sparse networks, considering the low number of heads h used.

4.3 Training and Generalization

The Graph Neural Network layers show remarkable inductive capabilities, as they can learn from very small networks (e.g., tens of nodes) and generalize to way larger ones (even millions of nodes) [92]. For this reason, we build the training set from small synthetic networks generated using the Static Power law model. In particular, we generate 100 networks of 20 nodes and 100 links each, with different in- and out- degree distribution combinations ($2.0 \leq \gamma^{in}, \gamma^{out} \leq 3.0$) [52]. In Figure 2 the train set generation and the training phases are shown.

In particular, nodes in the train networks are assigned a learning target value that depends on their belonging to any optimal set S for that network, i.e., to the sets of minimum cardinality that grant the target rank $t = 1$. Specifically, each node i of the train networks has a label value $y_i = S_i/S_t$, where S_t and S_i are, respectively, the total number of optimal sets and the number of those sets i belongs to. That is, $y_i \in [0, 1]$. This training target aims at teaching the model what nodes are more relevant to reach the top rank position.

After training the model, it can generalize to new networks providing a score for each node. In detail, given a network, the link-building process with our approach, depicted in Figure 3, consists of getting the node predictions p_i (i.e., computing the node features and feeding the network plus the features to the model), sorting the nodes in descending p_i order and creating a link from the nodes on the top to the target node x until the in-attachment target rank is reached.

5 EXPERIMENTS

5.1 Test Networks

As mentioned in the introduction, we test our proposal on 31 real-world directed networks covering a wide spectrum of domains—such as social, hyperlink, citation, vote and technological —, and with up to $2.5M$ nodes and $15.8M$ edges. We provide information about the networks used in our experiments as name, category, and references in Table 2, and some topological measures in Table 3. More specifically, in the latter, $|V|$ and $|E|$ are, respectively, the number of nodes and edges. *Avg. degree* is the nodes' average degree. With *Reciprocity*, we mean the fraction of links that are

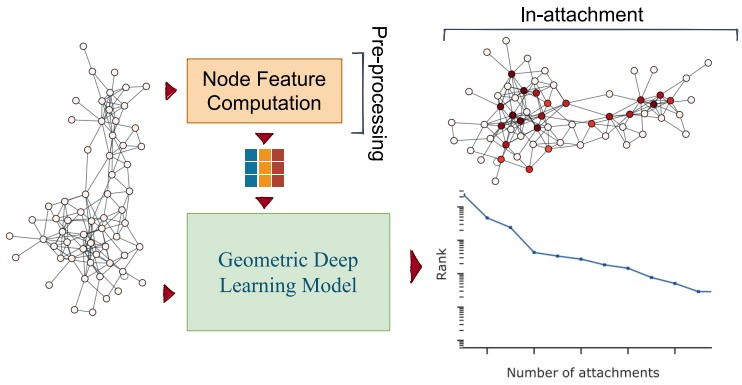

Fig. 3. Generalization to new networks. After training the model, it can generalize to (previously unseen) networks and provide a score for each node, used to rank them and build the attachment set S. It is worth mentioning that, in our experiments, we use the same training set described previously, independently of the target. The motivation is that we aim at teaching to the models what are the best nodes to get linked by.

reciprocal (i.e., given a link e_{ij} between the node i and j, the link e_{ji} also exists in the network). The *Density* is the ratio between the number of actual links and the number of all possible links (i.e., $|V| * (|V| - 1)$ in the case of directed networks). The *Strongly Connected Components* are the subnetworks such that there is a path between each pair of vertices, while the *Weakly Connected Components* are subnetworks such that there is a path between each pair of vertices, ignoring the directionality of links. The *Average Local Clustering Coefficient* measures the fraction of possible triangles that exist through a given node, while the *Transitivity* is the fraction of all possible triangles. We also analyze the *Degree assortativity* [72] of nodes. Moreover, we compute the node-level Pearson's correlation coefficient between the degree and the ***Average Nearest Neighbor Degree (ANND)*** [88]:

$$\text{ANND}_{\text{i}}^{\alpha/\beta} = \frac{\sum_j^{\mathcal{N}_i^\alpha} d_j^\beta}{k_i^\alpha}, \tag{21}$$

where $\alpha, \beta \in [\text{in}, \text{out}]$, i.e., we consider the different in-out link directions, $\mathcal{N}_i^\alpha$ is the α neighborhood of i and d_j^β is the degree of node j. We also report the maximum and average K*–core number*, as defined above.

5.2 Results

To evaluate the performance of our approach to the link-building problem, we choose the ***Area Under the Curve*** (**AUC**) as it accounts for how well a heuristic performs during the entire process, and allows for an extensive comparison on large test sets. For each heuristic and network, we calculate the AUC value using Simpson's rule on the $y(l) = R_G^{(i)}(l)/|V|$ curve, where $R_G^{(i)}(l)$ is node i's rank in the function of the number of in-links created l; the lower AUC the better (average) ranking during the in-attachment.

We test our approach on the datasets described above and stop the in-attachment when the newcomer node x reaches the first position. The cumulative results, i.e., the sum of the AUC values of all the networks (the lower, the better), for each heuristic are shown in Figure 4 in increasing cumulative AUC order, and a subset of the attachment curves is shown in Figures 5 and 6. We also report the full results in tabular form in Table 4, where, in order to facilitate the comparison, values

Table 2. Real-world Test Networks Name, Category, and References

Network	Name	Category	References
academia_edu	Academica.edu	Social	[50]
advogato	Advogato trust network	Social	[2, 69]
anybeat	Anybeat social network (2013)	Social	[49]
as-caida20040105	CAIDA Internet AS (2004/01/05)	Computer	[17]
bitcoin_alpha	Bitcoin Alpha	Trust	[59]
caida_20071105	CAIDA Internet AS (2007/11/05)	Computer	[17]
cfinder-google	Google.com internal	Hyperlink	[12, 79]
cit-HepPh	arXiv hep-ph	Citation	[3, 64]
cit-HepTh	arXiv hep-th	Citation	[4, 64]
dblp-cite	DBLP	Citation	[7, 66]
ego-gplus	Google+ (NIPS)	Social	[11, 70]
foldoc	FOLDOC	Hyperlink	[9, 23]
friendfeed	FriendFeed	Social	[39, 67]
google_plus	Google+ (2013)	Social	[50]
google_web	Old Google web graph (2002)	Hyperlink	[65]
moreno_blogs	Political blogs	Hyperlink	[5, 18]
moreno_health	Adolescent friendships	Friendship	[1, 71]
notre_dame_web	Notre Dame webgraph	Hyperlink	[19]
openflights	OpenFlights (Patokallio)	Infrastructure	[13]
p2p-Gnutella31	Gnutella hosts (31 Aug 2002)	Computer	[10, 85]
physician_friend	Physicians trust	Trust	[14, 41]
slashdot_zoo	Slashdot Zoo friend-foe network (2009)	Trust	[60]
soc-Epinions1	Epinions	Trust	[8, 83]
soc-sign-bitcoinotc	Bitcoin OTC	Trust	[16, 59]
stackoverflow_a2q	Q&A: StackOverflow (2016)	Social	[80]
stanford_web	Stanford webgraph	Hyperlink	[65]
subelj_cora	Cora	Citation	[6, 89]
superuser_all	Q&A: SuperUser (2016)	Social	[80]
trec_web	TREC WT10g webgraph (2003)	Hyperlink	[21]
twitter	Twitter	Social	[39, 67]
wiki-Vote	Wikipedia elections	Vote	[15, 63]

must be interpreted as the percentage of the AUC value scored by LB–GDM for the same network (i.e., values greater than 100 mean that LB–GDM outperforms the heuristic and vice-versa).

Our results show that not only LB–GDM performs significantly better than the state-of-the-art heuristics, reaching the first position with fewer links (smaller S) and achieving better rank during the whole process, but also that its computational complexity is comparable or even lower, as discussed in the previous sections.

According to these results, the closer heuristics are *Anticipated Value* and its dynamic version *Current Rank*. Both score an AUC value ~246% larger while requiring the computation of the PageRank. Regarding the *Future Rank* heuristic, while it exhibits excellent performance, its high computational complexity allows only evaluation on a subset of smaller test networks where it finishes in a reasonable time. The comparison, available in Table 4, shows that LB–GDM is able to match its performance or provide similar AUC values in many networks, and that, on average, its AUC is just 2.0% higher, which is remarkable considering its extremely lower computational time complexity.

The implementation of our models relies on PyTorch Geometric library [48] on top of PyTorch [81]. The graph data structures and their plots, the link creation algorithms, and the PageRank computation are implemented using the graph-tool [82] library.

Table 3. Topological Analysis of the Real-world Test Networks

Network	$\|V\|$	$\|E\|$	Avg. degree	Reciprocity	Density	Strongly Connected Components		Weakly Connected Components		Clustering Coefficient		Degree assortarivity	Avg. Nearest Neighbour Degree (ANND) [88] assortativity				K-core num.	
						Num.	Size largest	Num.	Size largest	Avg. Local	Transitivity		In-In	In-Out	Out-In	Out-Out	Max	Avg
academia_edu	200.2 K	1.4 M	13.964	0.537	0.000035	49.7 K	147.5K	2	200.2 K	0.176	0.012	−0.001	0.061	0.045	0.01	0.003	37	7.3
advogato	6.5 K	47.1 K	14.417	0.333	0.001103	3.4 K	3.1 K	1.4 K	5.0 K	0.148	0.065	0.03	0.119	0.046	0.09	0.145	31	7.5
anybeat	12.6 K	67.1 K	10.605	0.535	0.000419	4.0 K	8.5 K	1	12.6 K	0.18	0.012	−0.056	−0.053	−0.057	−0.046	−0.046	49	5.5
as-caida20040105	16.3 K	65.9 K	8.087	1	0.000248	1	16.3 K	1	16.3 K	0.233	0.008	−0.042	−0.042	−0.042	−0.042	−0.042	38	4.2
bitcoin_alpha	3.8 K	24.2 K	12.787	0.832	0.00169	540	3.2 K	5	3.8 K	0.158	0.064	−0.108	−0.104	−0.107	−0.05	−0.052	34	6.7
caida_20071105	26.5 K	106.8 K	8.065	1	0.000152	1	26.5 K	1	26.5 K	0.208	0.007	−0.036	−0.036	−0.036	−0.036	−0.036	44	4.1
cfinder-google	15.8 K	170.3 K	21.612	0.255	0.000686	3.4 K	12.4 K	1	15.8 K	0.343	0.174	−0.089	0.001	−0.029	−0.097	0.309	125	13.8
cit-HepPh	34.5 K	421.5 K	24.404	0.003	0.000353	21.6 K	12.7 K	61	34.4 K	0.143	0.101	0.119	0.266	0.159	0.137	0.4	30	12.8
cit-HepTh	27.8 K	352.8 K	25.406	0.003	0.000457	20.1 K	7.5 K	143	27.4 K	0.157	0.133	0.135	0.199	0.15	0.243	0.495	37	13.3
dblp-cite	12.6 K	49.7 K	7.899	0.004	0.000314	12.3 K	240	40	12.5 K	0.06	0.021	−0.041	0.058	0.008	0.342	0.142	12	4.1
ego-gplus	23.6 K	39.2 K	3.322	0.002	0.00007	23.6 K	50	4	23.6 K	0.109	0.001	−0.059	0.119	0.033	0.503	0.248	13	1.7
foldoc	13.4 K	120.2 K	18.005	0.479	0.000674	71	13.3K	1	13.4 K	0.298	0.228	0.14	0.105	0.204	0.461	0.429	12	10.6
friendfeed	5.5 K	31.9 K	11.523	0	0.00104	3.4 K	2.1K	40	5.5 K	0.156	0.105	−0.098	0.004	0.019	−0.171	−0.06	28	6.0
google_plus	211.2 K	1.5 M	14.254	0.483	0.000034	87.8 K	86.8 K	1.7 K	201.9 K	0.112	0.145	0.293	0.398	0.328	0.252	0.233	162	7.9
google_web	916.4 K	5.1 M	11.141	0.307	0.000006	412.5 K	434.8 K	43.5 K	855.8 K	0.353	0.45	−0.036	0.011	0.04	−0.003	0.382	64	6.8
moreno_blogs	1.2 K	19.0 K	31.082	0.243	0.012707	422	793	2	1.2 K	0.218	0.198	−0.221	−0.018	−0.064	−0.04	0.091	43	16.7
moreno_health	2.5 K	13.0 K	10.216	0.388	0.002013	368	2.2 K	1	2.5 K	0.12	0.142	0.522	0.447	0.428	0.597	0.579	10	5.9
notre_dame_web	54.0 K	296.2 K	10.978	0.39	0.000102	1	54.0 K	1	54.0 K	0.377	0.076	−0.044	−0.011	−0.017	−0.047	−0.047	50	6.2
openflights	3.4 K	37.6 K	21.953	0.976	0.003206	44	3.4 K	8	3.4 K	0.469	0.249	0.097	0.096	0.096	0.098	0.098	62	11.8
p2p-Gnutella31	62.6 K	147.9 K	4.726	0	0.000038	48.4 K	14.1 K	12	62.6 K	0.003	0.001	−0.246	0.009	0.011	0.766	0.713	6	2.5
physician_friend	0.2 K	0.5 K	4.439	0.328	0.009777	136	20	4	110	0.155	0.241	0.289	0.533	0.532	0.628	0.718	5	2.9
slashdot_zoo	79.1 K	515.4 K	13.028	0.185	0.000082	51.0 K	27.0 K	5	79.1 K	0.042	0.026	−0.02	0.033	0.055	0.025	0.122	84	6.6
soc-Epinions1	75.9 K	508.8 K	13.412	0.405	0.000088	42.2 K	32.2 K	2	75.9 K	0.11	0.066	0.03	0.129	0.021	0.087	0.103	91	6.9
soc-sign-bitcoinotc	5.9 K	35.6 K	12.104	0.792	0.001029	1.1 K	4.7 K	4	5.9 K	0.151	0.045	−0.107	−0.103	−0.11	−0.04	−0.043	35	6.3
stackoverflow_a2q	2.5 M	15.8 M	12.823	0.002	0.000003	2.1 M	410.7 K	45.3 K	2.4 M	0.008	0.001	−0.012	0.087	0.035	0.043	0.018	98	6.5
stanford_web	150.5 K	1.6 M	20.943	0.232	0.00007	1	150.5 K	1	150.5 K	0.412	0.416	−0.023	−0.001	−0.004	−0.039	0.123	86	12.0
subelj_cora	23.2 K	91.5 K	7.9	0.051	0.000171	18.1 K	4.0 K	1	23.2 K	0.146	0.128	−0.032	0.26	0.238	0.06	0.375	15	4.3
superuser_all	194.1 K	854.4 K	8.804	0.327	0.000023	108.3 K	83.8 K	3.2 K	189.2 K	0.079	0.004	−0.02	0.022	−0.004	0.002	−0.001	78	4.5
trec_web	470.4 K	3.0 M	12.807	0.372	0.000014	1	470.4 K	1	470.4 K	0.503	0.01	−0.022	−0.017	0.006	−0.015	−0.01	178	7.4
twitter	5.7 K	42.3 K	14.844	0	0.001301	2.4 K	3.3 K	41	5.6 K	0.089	0.055	0.106	0.195	0.148	0.116	0.865	40	7.8
wiki-Vote	7.1 K	103.7 K	29.147	0.056	0.002049	5.8 K	1.3 K	24	7.1 K	0.082	0.053	0.053	0.588	0.573	0.031	0.063	56	15.1

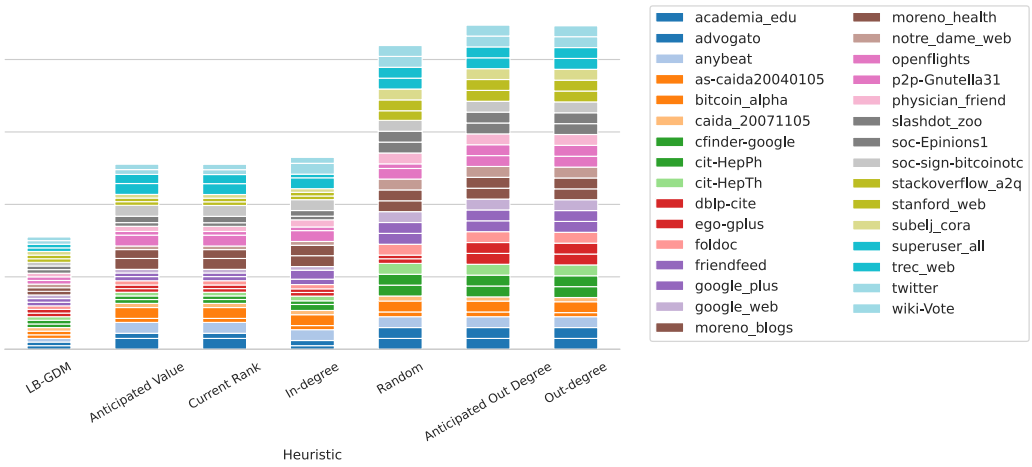

Fig. 4. Link-building in real-world networks. Per-method cumulative AUC of link-building in real-world networks. The lower, the better. The target rank is the first position. Each value is scaled to the one of our approaches (LB–GDM) for the same network. Note that some values are clipped to 3x to improve visualization.

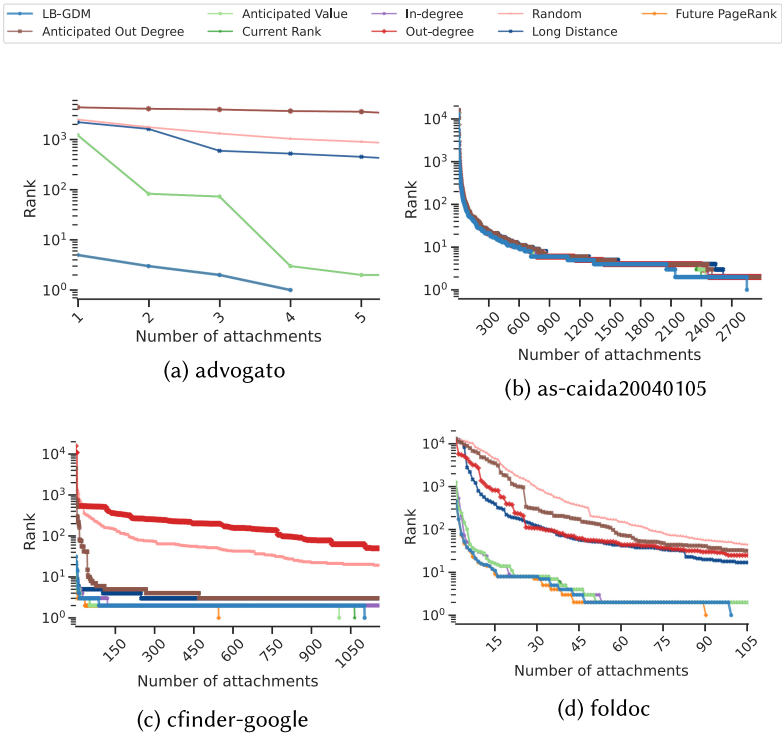

Fig. 5. Link-building in real-world networks (part 1/2). Attachment curves of the networks in our test set. The y-axis value is the rank of the target node as a function of the number of in-links added. LB–GDM performs better than the cutting-edge heuristics as not only it provides better AUC (i.e., the rank is - on average - lower with the same number of new links), but also requires fewer links to reach the target in many cases.

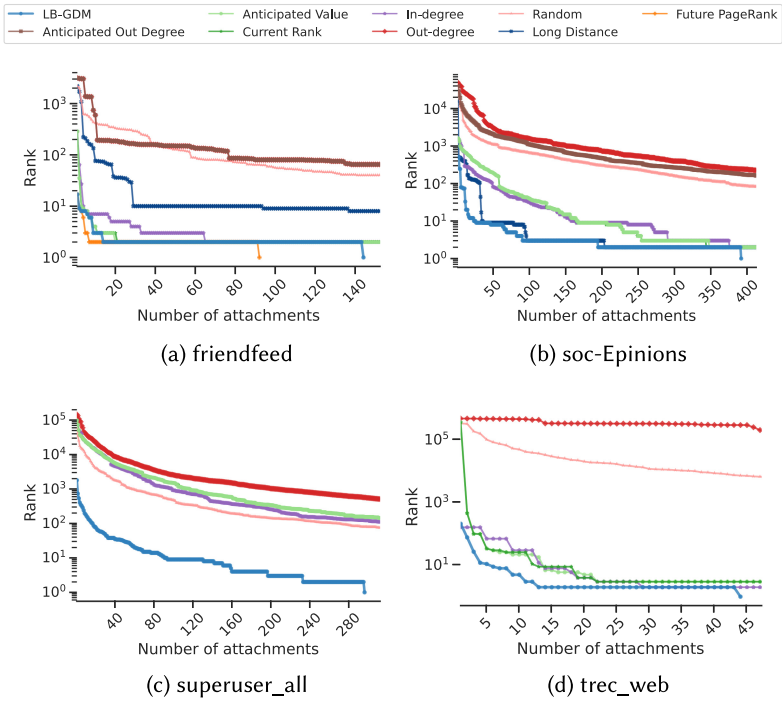

Fig. 6. Link-building in real-world networks (part 2/2). Attachment curves of the networks in our test set. The y-axis value is the rank of the target node as a function of the number of in-links added. LB–GDM performs better than the cutting-edge heuristics as not only it provides better AUC (i.e., the rank is - on average - lower with the same number of new links), but also requires fewer links to reach the target in many cases.

5.3 Insights and Explanation

Machine Learning has been widely used to approach many important problems, often with super-human performances. Unfortunately, complex models (e.g., convolutional ones) are a black-box that is difficult to interpret, but trying to understand their predictions (i.e., *explaining the model*) is a hot topic in the field. For instance, opening a window on the black-box can improve the trust in the models and, thus, make them suitable for critical applications; it can help to improve the model itself (e.g., by feeding better features, or by tuning the model architecture), and, more importantly, can also help to deepen the understanding of the approached problem with insights that come from the data and that may not be intuitive or immediate to humans. In this section, we try to explain our Geometric Deep Learning models. While such a task is hard, as each model combines both the node features and the network topology to produce the output, there are some explanation tools available for Graph Neural Networks. In particular, we use *GNNExplainer* [93], which, given a trained model and a target node, learns a feature and an edge masks, such that, when applied to the computational sub-graph[3] of the node, the predictions are as close as possible to the ones made in the full sub-graph. More in detail, the feature mask is used to scale the input node features and quantifies the relevance of a feature in the prediction, while the edge mask quantifies the contribution of each edge in the sub-graph to the final prediction.

[3]The computational sub-graph of a node is its K-hop in-neighborhood (with K being the number of GNN layers in the model) where the message passing propagation is performed.

Table 4. Full Results Table

Network \ Heuristic	LB–GDM	Future PageRank	Anticipated Value	Current Rank	In Degree	Long Distance	Random	Anticipated Out Degree	Out Degree
academia_edu	100.0	-	413.9	413.9	100.0	370.3	855.2	5,981.5	5,981.5
advogato	100.0	100.0	141.9	142.0	142.2	306.0	626.5	3,438.1	3,438.1
anybeat	100.0	96.5	308.5	307.9	311.0	190.1	315.2	944.2	748.7
as-caida20040105	100.0	-	110.9	110.6	112.7	129.2	131.3	130.4	112.7
bitcoin_alpha	100.0	97.1	974.9	975.4	980.9	612.3	868.8	1,252.2	1,221.6
caida_20071105	100.0	-	114.7	114.6	116.9	124.3	139.9	116.9	116.9
cfinder-google	100.0	95.8	106.3	107.5	165.6	179.5	1,466.0	369.7	4,079.7
cit-HepPh	100.0	100.0	100.0	100.0	100.6	248.2	498.6	1,870.5	15,992.2
cit-HepTh	100.0	100.0	100.9	100.8	130.6	328.6	1,081.8	2,544.6	13,528.7
dblp-cite	100.0	100.0	100.1	100.1	100.0	208.2	128.7	712.9	2,757.2
ego-gplus	100.0	100.0	100.0	100.0	100.0	216.1	100.0	1,120.6	1,121.4
foldoc	100.0	99.3	117.6	117.7	118.7	890.5	2,379.8	1,757.8	885.3
friendfeed	100.0	95.9	114.9	113.8	149.0	395.5	1,274.4	1,870.4	1,870.9
google_plus	100.0	100.0	100.0	100.0	247.8	456.5	797.3	26,185.6	26,185.6
google_web	100.0	-	100.2	100.2	100.3	-	748.0	1,787.5	1,787.5
moreno_blogs	100.0	100.0	346.3	354.4	310.5	308.7	1,487.2	5,212.8	4,957.8
moreno_health	100.0	100.0	245.4	244.9	286.7	298.2	1,002.1	1,145.1	1,145.1
notre_dame_web	100.0	-	99.3	99.2	105.4	-	1,308.6	2,	
027.9	1,996.2								
openflights	100.0	72.6	406.7	406.2	492.8	172.3	291.8	493.1	493.1
p2p-Gnutella31	100.0	100.0	100.8	100.8	100.6	170.7	127.9	473.9	2,600.8
physician_friend	100.0	100.0	143.7	141.1	198.0	415.9	539.7	617.9	584.8
slashdot_zoo	100.0	100.0	100.7	100.7	100.6	239.8	373.8	3,291.1	3,291.1
soc-Epinions1	100.0	-	176.2	176.2	155.2	160.4	1,048.0	1,587.5	3,026.3
soc-sign-bitcoinotc	100.0	98.7	1,068.9	1,069.4	1,034.2	311.6	919.1	1,487.9	1,474.7
stackoverflow_a2q	100.0	-	101.9	101.9	101.9	-	265.7	32,477.1	32,444.3
stanford_web	100.0	-	100.5	100.5	100.8	-	2,690.6	3,922.0	3,921.8
subelj_cora	100.0	99.9	100.4	100.4	100.7	428.0	1,011.9	676.6	5,814.9
superuser_all	100.0	-	1,121.9	1,122.1	1,073.4	-	513.5	2,090.4	2,090.3
trec_web	100.0	-	251.2	251.2	100.4	-	1,151.4	11,539.3	11,539.3
twitter	100.0	98.9	133.5	134.2	456.2	431.1	873.9	3,929.4	3,929.4
wiki-Vote	100.0	100.0	147.2	147.2	162.9	166.2	443.5	9,016.9	8,930.8
Average	100.0	97.8	246.7	246.9	253.4	310.3	821.3	4,195.9	5,421.6

Per-method cumulative AUC of link-building in real-world networks. The lower, the better. To improve readability, for each network and method, we report the result as the percentage of the value scored by LB–GDM for the same network. That is, if the value is greater than 100 LB–GDM outperforms the method, and is outperformed otherwise. Regarding the Future PageRank and Long Distance heuristics, some networks are omitted as the heuristic would not complete in the reasonable time (i.e., less than two weeks on our server-grade hardware).

In Figure 7, we report an example of the most important edges (in black) of the explanation sub-graphs for the *moreno_health* network, chosen because smaller than the others in the dataset (thus, making the plots more readable). While the sub-graph has non-trivial connection patterns, meaning that the model is leveraging the topology for the predictions, we notice that it includes many links from well-connected first-hop neighboring nodes (predicted to have a lower-probability of being in the optimal set) to the highest probability node. This indicates that the model is effectively learning the PageRank dynamics, since for a node to have a high PageRank, its neighbors must be well-connected. We also notice that, when available, the model tends to prefer high in-degree sinks first, even though it does not attach to them exclusively and other nodes are picked among them, as proven by the better scores of our approach. To further improve the detection of the best sinks, we have also included the ratio between the normalized in and out-degree. This new feature provided a large performance boost that does not affect the computational complexity of the approach.

Regarding the input node features, we show the node feature importance mask for a subset of the networks in Figure 8. Specifically, each plot shows the relative importance of the input node

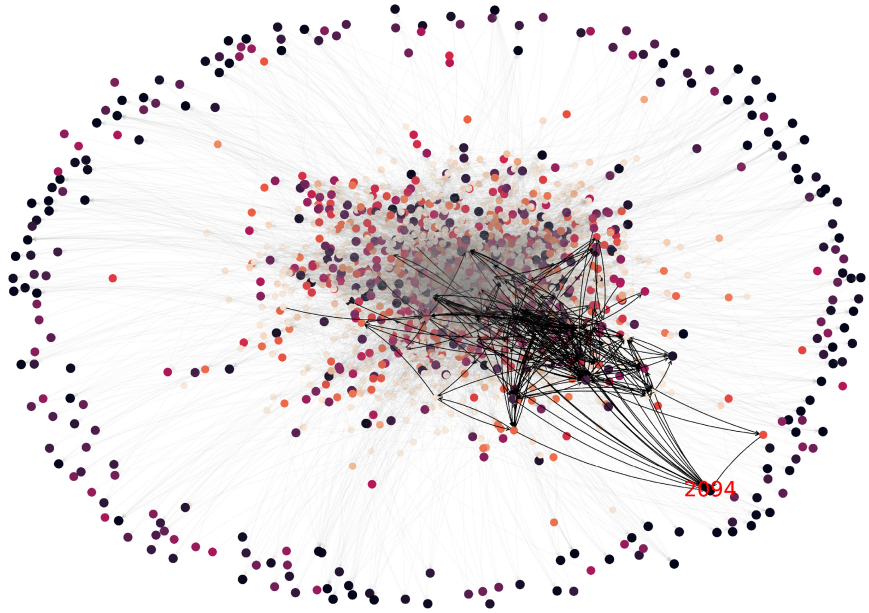

Fig. 7. Explanation sub-graph of the first attachment predicted by the model for the *moreno_health* network. The node color represents the predicted score (the darker, the higher), while the edges in black are the ones that belong to the explanation subgraph returned by *GNNExplainer* for node 2,094, the one with a higher predicted score.

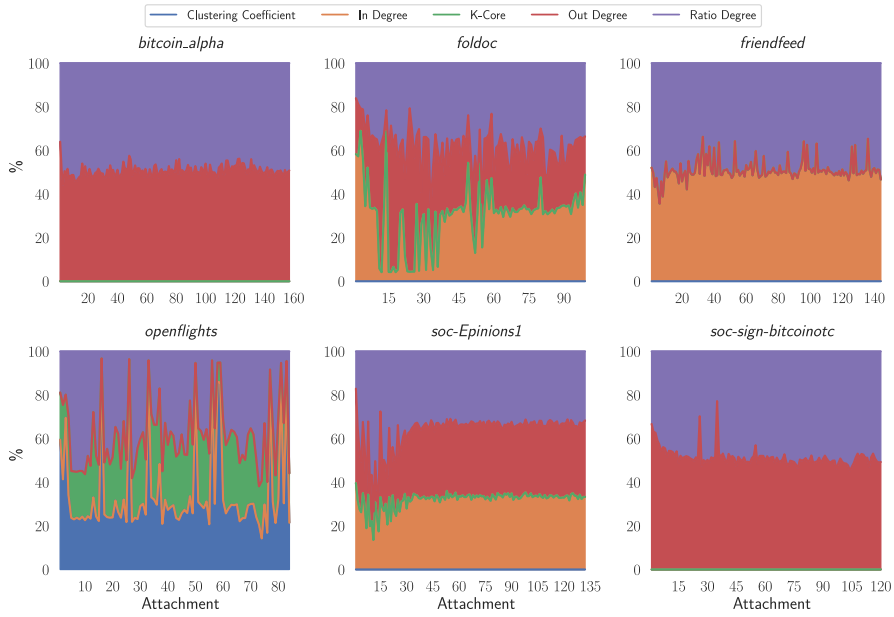

Fig. 8. Relative feature importance of a subset of the test networks. The importance of each feature is different for each attachment and network, meaning it depends on the complex topological patterns of the node.

features for each attachment. For instance, in the *bitcoin_alpha* network to make predictions the only used nodes' features are the *Out Degree* and the *Ratio Degree*. However, as the Reader may notice, the same model may use the node features differently, even for consecutive nodes, which makes translating the models into non-trainable algorithms not feasible.

6 CONCLUSIONS & FUTURE WORKS

In this work, we proposed a solution to the link-building problem based on Geometric Deep Learning. Tests conducted on several real-world networks show that our low computational complexity approach outperforms the state-of-the-art algorithms, proving its validity. Moreover, the methodology proposed is general and can be applied to other similar problems. As future work, we plan to generalize the link-building problem to other node centrality measures, such as betweenness, closeness, or eigenvector.

APPENDIX

A MODEL PARAMETERS

We perform a grid search to test various combinations of model parameters, reported here, and select the models that better fit the attachment target.

- Graph Neural Network layers: *Graph Attention Networks* (GAT):
  - Number of layers: from 1 to 3;
  - Output channels for each layer: 5, 10, 20, 30, 40, or 50, sometimes with a decreasing value between consecutive layers;
  - Multi-head attentions: 1, 5, 10, 15, 20, or 30 with concatenation. The last layer's heads are sum together;
  - Dropout probability: fixed to 0.3;
  - Leaky ReLU angle of the negative slope: fixed to 0.2;
  - Each layer learns an additive bias;
  - Each layer is coupled with a linear layer with the same number of input and output channels in order to create residual connections from one layer to the one below;
  - Activation function: ELU. The input at each convolutional layer is the sum between the output of the GAT and the linear layers;
- Regressor: Multi-Layer Perceptron:
  - Number of layers: from 1 to 4;
  - Number of neurons per layer: 20, 30, 40, 50, or 100, sometimes with a decreasing value between consecutive layers.
- Learning rate: fixed to 10^{-5};
- Epochs: we train each model for 50 epochs;

REFERENCES

[1] 2017. Adolescent health network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/moreno_health.
[2] 2017. Advogato network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/advogato.
[3] 2017. arXiv hep-ph network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/cit-HepPh.
[4] 2017. arXiv hep-th network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/cit-HepTh.
[5] 2017. Blogs network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/moreno_blogs.
[6] 2017. Cora network dataset – KONECT. (October 2017). Retrieved from http://konect.cc/networks/subelj_cora.

[7] 2017. DBLP network dataset – KONECT. (October 2017). Retrieved from http://konect.cc/networks/dblp-cite.
[8] 2017. Epinions network dataset – KONECT. (October 2017). Retrieved from http://konect.cc/networks/soc-Epinions1.
[9] 2017. FOLDOC network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/foldoc.
[10] 2017. Gnutella (31) network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/p2p-Gnutella31.
[11] 2017. Google+ (NIPS) network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/ego-gplus.
[12] 2017. Google.com internal network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/cfinder-google.
[13] 2017. OpenFlights (Patokallio) network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/openflights.
[14] 2017. Physicians network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/moreno_innovation.
[15] 2017. Wikipedia elections network dataset – KONECT. (October 2017). Retrieved 6 July, 2019 from http://konect.cc/networks/elec.
[16] 2018. Bitcoin OTC network dataset – KONECT. (Febraury 2018). Retrieved 6 July, 2019 from http://konect.cc/networks/soc-sign-bitcoinotc.
[17] 2022. The CAIDA AS Relationships Dataset. (2022). Retrieved 6 July, 2019 from https://www.caida.org/catalog/datasets/as-relationships/.
[18] Lada A. Adamic and Natalie Glance. 2005. The political blogosphere and the 2004 US election: Divided they blog. In *Proceedings of the 3rd International Workshop on Link Discovery*. 36–43.
[19] Réka Albert, Hawoong Jeong, and Albert-László Barabási. 1999. Diameter of the World-Wide Web. *Nature* 401, 6749 (1999), 130–131. DOI: https://doi.org/10.1038/43601
[20] K. Avrachenkov and N. Litvak. 2006. The effect of new links on Google Pagerank. *Stochastic Models* 22, 2 (2006), 319–331. Retrieved from http://doc.utwente.nl/63648/.
[21] Peter Bailey, Nick Craswell, and David Hawking. 2003. Engineering a multi-purpose test collection for Web retrieval experiments. *Information Processing and Management* 39, 6 (2003), 853–871. https://doi.org/10.1016/S0306-4573(02)00084-5
[22] Alain Barrat, Marc Barthelemy, Romualdo Pastor-Satorras, and Alessandro Vespignani. 2004. The architecture of complex weighted networks. *Proceedings of the National Academy of Sciences* 101, 11 (2004), 3747–3752.
[23] V. Batagelj, A. Mrvar, and M. Zaversnik. 2002. Network analysis of texts. In *Proceedings of the Language Technologies*. 143–148.
[24] Pavel Berkhin. 2005. A survey on PageRank computing. *Internet Mathematics* 2, 1 (7 2005), 73–120.
[25] Monica Bianchini, Marco Gori, and Franco Scarselli. 2005. Inside PageRank. *ACM Transactions on Internet Technology* 5, 1 (2005), 92–128. DOI: https://doi.org/10.1145/1052934.1052938
[26] Stephen P. Borgatti and Martin G. Everett. 2006. A graph-theoretic perspective on centrality. *Social Networks* 28, 4 (2006), 466–484. DOI: https://doi.org/10.1016/j.socnet.2005.11.005
[27] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems* 30, 1 (1998), 107–117. DOI: https://doi.org/10.1016/S0169-7552(98)00110-X
[28] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. 2017. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine* 34, 4 (2017), 18–42. DOI: https://doi.org/10.1109/MSP.2017.2693418
[29] M. Buzzanca, V. Carchiolo, A. Longheu, M. Malgeri, and G. Mangioni. 2017. Dealing with the best attachment problem via heuristics. In *Proceedings of the Intelligent Distributed Computing X*. Costin et al. Badica (Eds.), Vol. 678. Springer International Publishing, Cham, 205–214. DOI: https://doi.org/10.1007/978-3-319-48829-5_20
[30] Marco Buzzanca, Vincenza Carchiolo, Alessandro Longheu, Michele Malgeri, and Giuseppe Mangioni. 2018. Black hole metric: Overcoming the pagerank normalization problem. *Information Sciences* 438 (2018), 58–72.
[31] Vincenza Carchiolo, Marco Grassia, Alessandro Longheu, Michele Malgeri, and Giuseppe Mangioni. 2018. Climbing ranking position via long-distance backlinks. *11th International Conference on Internet and Distributed Computing Systems*. Springer International Publishing, 100–108. DOI: https://doi.org/10.1007/978-3-030-02738-4_9
[32] V. Carchiolo, M. Grassia, A. Longheu, M. Malgeri, and G. Mangioni. 2018. Long distance in-links for ranking enhancement. In *Proceedings of the Intelligent Distributed Computing XII*. Javier Del Ser, Eneko Osaba, Miren Nekane Bilbao, Javier J. Sanchez-Medina, Massimo Vecchio, and Xin-She Yang (Eds.), Springer International Publishing, Cham, 3–10.
[33] Vincenza Carchiolo, Alessandro Longheu, and Michele Malgeri. 2010. Reliable peers and useful resources: Searching for the best personalised learning path in a trust- and recommendation-aware environment. *Information Sciences* 180, 10 (2010), 1893–1907. DOI: https://doi.org/10.1016/j.ins.2009.12.023 Special Issue on Intelligent Distributed Information Systems.

[34] Vincenza Carchiolo, Alessandro Longheu, Michele Malgeri, and Giuseppe Mangioni. 2012. Gain the best reputation in trust networks. In *Proceedings of the Intelligent Distributed Computing V*. F.M.T. Brazier, Kees Nieuwenhuis, Gregor Pavlin, Martijn Warnier, and Costin Badica (Eds.), Studies in Computational Intelligence, Vol. 382. Springer, Berlin, 213–218. DOI: https://doi.org/10.1007/978-3-642-24013-3_21

[35] Vincenza Carchiolo, Alessandro Longheu, Michele Malgeri, and Giuseppe Mangioni. 2014. A heuristic to explore trust networks dynamics. In *Proceedings of the Intelligent Distributed Computing VII*. Filip Zavoral, Jason J. Jung, and Costin Badica (Eds.), Springer International Publishing, Cham, 67–76.

[36] Vincenza Carchiolo, Alessandro Longheu, Michele Malgeri, and Giuseppe Mangioni. 2015. The cost of trust in the dynamics of best attachment. *Computing and Informatics* 34, 1 (2015), 167–184.

[37] Vincenza Carchiolo, Alessandro Longheu, Michele Malgeri, and Giuseppe Mangioni. 2015. The effect of topology on the attachment process in trust networks. In *Proceedings of the Intelligent Distributed Computing VIII*. Springer, 377–382.

[38] Vincenza Carchiolo, Alessandro Longheu, Michele Malgeri, and Giuseppe Mangioni. 2015. Searching for experts in a context-aware recommendation network. *Computers in Human Behavior* 51 (2015), 1086–1091. DOI: https://doi.org/10.1016/j.chb.2015.03.028

[39] Fabio Celli, F. Marta L. Di Lascio, Matteo Magnani, Barbara Pacelli, and Luca Rossi. 2010. Social network data and practices: The case of Friendfeed. In *Proceedings of the International Conference on Social Computing, Behavioral Modeling and Prediction (Lecture Notes in Computer Science)*. Springer, Berlin.

[40] Li Chen, Guanliang Chen, and Feng Wang. 2015. Recommender systems based on user reviews: The state of the art. *User Modeling and User-Adapted Interaction* 25, 2 (2015), 99–154. DOI: https://doi.org/10.1007/s11257-015-9155-5

[41] James Coleman, Elihu Katz, and Herbert Menzel. 1957. The diffusion of an innovation among physicians. *Sociometry* 20, 4 (1957), 253–270.

[42] ©Google. 2019. Facts about Google and Competition. Retrieved 6 July, 2019 from https://web.archive.org/web/20111104131332/https://www.google.com/competition/howgooglesearchworks.html.

[43] Balázs Csanád Csáji, Raphaël M. Jungers, and Vincent D. Blondel. 2009. PageRank optimization by edge selection. *Discrete Applied Mathematics* 169 (2014), 73–87. https://doi.org/10.1016/j.dam.2014.01.007

[44] L. da F. Costa, F. A. Rodrigues, G. Travieso, and P. R. Villas Boas. 2007. Characterization of complex networks: A survey of measurements. *Advances in Physics* 56, 1 (2007), 167–242. DOI: https://doi.org/10.1080/00018730601170527

[45] Kousik Das, Sovan Samanta, and Madhumangal Pal. 2018. Study on centrality measures in social networks: A survey. *Social Network Analysis and Mining* 8, 1 (2018), 1–18. DOI: https://doi.org/10.1007/s13278-018-0493-2

[46] Cristobald de Kerchove, Laure Ninove, and Paul Van Dooren. 2007. Maximizing PageRank via outlinks. *Linear Algebra and its Applications* 429, 5–6 (2008), 1254–1276.

[47] S. N. Dorogovtsev, A. V. Goltsev, and J. F. F. Mendes. 2006. *k*-core organization of complex networks. *Physical Review Letters* 96, 4 (2006), 040601. DOI: https://doi.org/10.1103/PhysRevLett.96.040601

[48] Matthias Fey and Jan E. Lenssen. 2019. Fast graph representation learning with PyTorch geometric. In *Proceedings of the ICLR Workshop on Representation Learning on Graphs and Manifolds*.

[49] Michael Fire, Rami Puzis, and Yuval Elovici. 2013. *Link Prediction in Highly Fractional Data Sets*. Springer New York. DOI: https://doi.org/10.1007/978-1-4614-5311-6_14

[50] Michael Fire, Lena Tenenboim-Chekina, Rami Puzis, Ofrit Lesser, Lior Rokach, and Yuval Elovici. 2014. Computationally efficient link prediction in a variety of social networks. *ACM Transactions on Intelligent Systems and Technology* 5, 1 (2014), 25 pages. DOI: https://doi.org/10.1145/2542182.2542192

[51] David F. Gleich. 2014. PageRank beyond the web. *SIAM Review* 57, 3 (2015), 321–363. https://doi.org/10.1137/140976649

[52] K.-I. Goh, B. Kahng, and D. Kim. 2001. Universal behavior of load distribution in scale-free networks. *Physical Review Letters* 87, 278701 (2001). DOI: https://doi.org/10.1103/physrevlett.87.278701

[53] Marco Grassia, Manlio De Domenico, and Giuseppe Mangioni. 2021. Machine learning dismantling and early-warning signals of disintegration in complex systems. *Nature Communications* 12, 1 (2021), 5190. DOI: https://doi.org/10.1038/s41467-021-25485-8

[54] Marco Grassia and Giuseppe Mangioni. 2021. wsGAT: Weighted and signed graph attention networks for link prediction. In *Proceedings of the International Conference on Complex Networks and Their Applications*. Springer, 369–375.

[55] Pankaj Gupta, Ashish Goel, Jimmy Lin, Aneesh Sharma, Dong Wang, and Reza Zadeh. 2013. WTF: The who to follow service at Twitter. In *Proceedings of the 22nd International Conference on World Wide Web*. Retrieved from http://dl.acm.org/citation.cfm?id=2488388.2488433.

[56] Bin Jiang, Sijian Zhao, and Junjun Yin. 2008. Self-organized natural roads for predicting traffic flow: A sensitivity study. *Journal of Statistical Mechanics: Theory and Experiment* 2008, 07 (2008), P07008. DOI: https://doi.org/10.1088/1742-5468/2008/07/p07008

[57] Sepandar Kamvar, Ar Kamvar, Taher Haveliwala, and Gene Golub. 2003. *Adaptive Methods for the Computation of PageRank*. Technical Report. Stanford University.
[58] Maksim Kitsak, Lazaros K. Gallos, Shlomo Havlin, Fredrik Liljeros, Lev Muchnik, H. Eugene Stanley, and Hernán A. Makse. 2010. Identification of influential spreaders in complex networks. *Nature Physics* 6, 11 (2010), 888–893.
[59] Srijan Kumar, Francesca Spezzano, V. S. Subrahmanian, and Christos Faloutsos. 2016. Edge weight prediction in weighted signed networks. In *Proceedings of the 16th International Conference on Data Mining*. 221–230.
[60] Jérôme Kunegis, Andreas Lommatzsch, and Christian Bauckhage. 2009. The slashdot zoo: Mining a social network with negative edges. In *Proceedings of the 18th International Conference on World Wide Web* . Association for Computing Machinery, New York, NY, 741–750. DOI: https://doi.org/10.1145/1526709.1526809
[61] Vito Latora, Vincenzo Nicosia, and Giovanni Russo. 2017. *Complex Networks: Principles, Methods and Applications*. Cambridge University Press.
[62] John Boaz Lee, Ryan A. Rossi, Sungchul Kim, Nesreen K. Ahmed, and Eunyee Koh. 2019. Attention models in graphs: A survey. *ACM Transactions on Knowledge Discovery from Data* 13, 6 (2019), 1–25.
[63] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. 2010. Governance in social media: A case study of the Wikipedia promotion process. In *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media*.
[64] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data* 1, 1 (2007), 1–40.
[65] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. 2008. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics* 6, 1 (2008), 29–123. DOI: 10.1080/15427951.2009.10129177
[66] Michael Ley. 2002. The DBLP computer science bibliography: Evolution, research issues, perspectives. In *Proceedings of the International Symposium on String Processing and Information Retrieval*. 1–10.
[67] Matteo Magnani and Luca Rossi. 2011. The ML-model for multi-layer social networks. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining*. IEEE Computer Society, 5–12.
[68] S. Marsh. 1994. *Formalising Trust as a Computational Concept*. Technical Report. University of Stirling. PhD thesis.
[69] Paolo Massa, Martino Salvetti, and Danilo Tomasoni. 2009. Bowling alone and trust decline in social network sites. In *Proceedings of the 9th IEEE International Conference on Dependable, Autonomic and Secure Computing*. 658–663.
[70] Julian McAuley and Jure Leskovec. 2012. Learning to discover social circles in ego networks. In *Proceedings of the Advances in Neural Information Processing Systems* 548–556.
[71] James Moody. 2001. Peer influence groups: Identifying dense clusters in large networks. *Social Networks* 23, 4 (2001), 261–283.
[72] M. E. J. Newman. 2003. Mixing patterns in networks. *Physical Review E* 67, 2 (2003), 026126. DOI: https://doi.org/10.1103/PhysRevE.67.026126
[73] Mark E. J. Newman. 2003. The structure and function of complex networks. *SIAM Review* 45, 2 (2003), 167–256.
[74] Martin Olsen. 2008. The computational complexity of link building. In *Proceedings of the Computing and Combinatorics*. Xiaodong Hu and Jie Wang (Eds.), Springer, Berlin, 119–129.
[75] Martin Olsen. 2010. Maximizing PageRank with new backlinks. In *Proceedings of the Algorithms and Complexity*, Tiziana Calamoneri and Josep Diaz (Eds.), Springer, Berlin, 37–48.
[76] Martin Olsen and Anastasios Viglas. 2014. On the approximability of the link building problem. *Theoretical Computer Science* 518 (2014), 96–116. DOI: https://doi.org/10.1016/j.tcs.2013.08.003
[77] Martin Olsen, Anastasios Viglas, and Ilia Zvedeniouk. 2012. An approximation algorithm for the link building problem. arXiv:1204.1369. Retrieved from http://arxiv.org/abs/1204.1369
[78] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1998. The PageRank Citation Ranking: Bringing Order to the Web. (1998). Retrieved 6 July, 2019 from http://citeseer.ist.psu.edu/article/page98pagerank.html.
[79] Gergely Palla, Illés J. Farkas, Péter Pollner, Imre Derényi, and Tamás Vicsek. 2007. Directed network modules. *New Journal of Physics* 9, 6 (2007), 186.
[80] Ashwin Paranjape, Austin R. Benson, and Jure Leskovec. 2017. Motifs in temporal networks. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery, New York, NY, 601–610. DOI: https://doi.org/10.1145/3018661.3018731
[81] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Proceedings of the Advances in Neural Information Processing Systems*. H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Curran Associates, Inc., 8024–8035. Retrieved from http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[82] Tiago P. Peixoto. 2014. The graph-tool python library. *Figshare* (2014). DOI: https://doi.org/10.6084/m9.figshare.1164194

[83] Matthew Richardson, Rakesh Agrawal, and Pedro Domingos. 2003. Trust management for the semantic web. In *Proceedings of the International Semantic Web Conference*. 351–368.

[84] Mathew Richardson and Pedro Domingos. 2002. The intelligent surfer: Probabilistic combination of link and content information in PageRank. In *Proceedings of the Advances in Neural Information Processing Systems*. MIT Press. Retrieved from http://citeseer.ist.psu.edu/460350.html.

[85] Matei Ripeanu, Ian Foster, and Adriana Iamnitchi. 2002. Mapping the Gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. In *IIPTPS'01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*. 85–93.

[86] Antonio J. Roa-Valverde and Miguel-Angel Sicilia. 2014. A survey of approaches for ranking on the web of data. *Information Retrieval* 17, 4 (2014), 295–325. DOI: https://doi.org/10.1007/s10791-014-9240-0

[87] Jesus Serrano-Guerrero, Francisco Romero, and José Olivas. 2013. Hiperion: A fuzzy approach for recommending educational activities based on the acquisition of competences. *Information Sciences* 248 (2013), 114–129.

[88] Tiziano Squartini, Giorgio Fagiolo, and Diego Garlaschelli. 2011. Randomizing world trade. I. A binary network analysis. *Physical Review E* 84, 4 (2011). DOI: https://doi.org/10.1103/physreve.84.046117

[89] Lovro Šubelj and Marko Bajec. 2013. Model of complex networks based on citation dynamics. In *Proceedings of the WWW Workshop on Large Scale Network Analysis*. 527–530.

[90] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lió, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rJXMpikCZ.

[91] Frank E. Walter, Stefano Battiston, and Frank Schweitzer. 2008. A model of a trust-based recommendation system on a social network. *Journal of Autonomous Agents and Multi-Agent Systems* 16 (2008), 57. DOI: 10.1007/s10458-007-9021-x

[92] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S. Yu Philip. 2021. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 32, 1 (2021), 4–24. DOI: 10.1109/TNNLS.2020.2978386

[93] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. GNNExplainer: Generating Explanations for Graph Neural Networks. *Advances in Neural Information Processing Systems* Article 829 (2019), 9244–9255. DOI: https://doi.org/10.48550/ARXIV.1903.03894

[94] A. O. Zhirov, O. V. Zhirov, and D. L. Shepelyansky. 2010. Two-dimensional ranking of Wikipedia articles. *Eur. Phys. J. B* 77 (2010), 523.

Received 26 December 2021; revised 15 June 2022; accepted 19 July 2022