Hiding Relationships in a Social Network

Marcin Waniek
University of Warsaw
vua@mimuw.edu.pl

Tomasz P. Michalak
University of Oxford
& University of Warsaw
tomasz.michalak@cs.ox.ac.uk

Talal Rahwan
Masdar Institute of Science
and Technology
trahwan@gmail.com

ABSTRACT

Various algorithms have been proposed for link prediction to date. For all their benefits, such algorithms raise serious privacy concerns, as they could be used to expose a connection between two individuals who wish to keep their relationship private. We investigate the ability of such individuals to strategically alter their connections so as to increase the probability that some of their connections remain unidentified by link prediction algorithms. We prove that this problem is NP-complete. Despite this hardness, we propose two relatively-effective heuristics that can easily be implemented by members of the general public on existing social media.

1. INTRODUCTION

The Internet and social media have fueled enormous interest in social network analysis. Researchers and practitioners alike have focused on developing new social network analysis tools, such as centrality measures, community-detection algorithms, and classification algorithms, just to name a few [36]. Our on-line social data, email and telephone traffics, or even financial data are used to reveal personal information that is otherwise confidential [6]. This raises both privacy and security related questions as our on-line data may be of value not only to enterprises and public entities but also to cyber criminals, especially since such criminals are increasingly relying on network science to harvest and analyse information for malicious purposes [4].

One of the key research challenges in social network analysis is the link prediction problem [25, 28]. Intuitively, based on the current structure of the network, this problem involves predicting the connections that are most likely to be created in the future, [25]. An alternative interpretation of this problem is to identify the connections that are *hidden* from the observer, either due to scarcity of data, or due to the deliberate concealment of information [8]. Link prediction has many applications, such as providing recommendations to customers in e-commerce [12], discovering the interactions between proteins in biological networks [9], and finding hidden connections between terrorists [2] or criminals [39].

A plethora of different link prediction algorithms have been proposed in the literature (see the works by [25, 28, 3] for an extensive overview). In essence, the aim of all such algorithms is to estimate the likelihood that there exists a not-yet-discovered edge between two seemingly-disconnected nodes, or the likelihood that an edge will be formed between those two nodes in the future [16]. The mainstream class of link-prediction algorithms is based on *similarity indices* [28]. As the name suggests, such indices measure the similarity between any two disconnected nodes in a network by analysing its topology. The underlying assumption behind similarity-based algorithms is that the greater the similarity between two nodes, the greater the likelihood of having a link between them.

If used with malicious or mischievous intent, social network analysis tools—and link prediction algorithms in particular—may constitute a serious threat to both the privacy and the security of the general public. Importantly, not only can such tools use the data that has been willingly disclosed by its owners, but they can also use private data that has not been disclosed at all. In particular, it has been shown that such private data can be inferred from the publicly disclosed data, and that is by performing an *attribute inference attack*. As the name suggests, such an attack infers the missing or partial attributes of network nodes [46, 30]. In this article, we are particularly interested in a special type of such attacks, called the *link reconstruction attack*, whereby a link prediction algorithm is used to identify missing or hidden links [15].

Given such concerns, a number of studies recommended that social network users conceal some of their attributes, as a countermeasure against attribute inference attacks [27, 18]. Such recommendations include also concealing links. In this context, although a number of studies in the literature have argued *why* there is a need to conceal one's private links, they unfortunately did not specify *how* this should be done.

Against this background, given a "*seeker*" who is running a link prediction algorithms, and "*evaders*" who want to hide some of their connections, we study how the evaders can make those connections harder for the seeker to identify. This research question matters because, on one hand, it may assist the general public in protecting their privacy from intrusion by private and public entities; on the other hand, it may mitigate (at least to a some extent) the threats posed by cyber criminals. It may also assist law-enforcement agencies in understanding how criminals and terrorists could evade social network analysis tools, especially given their increasing reliance on social-media survival strategies [32, 20].

Appears in: *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

With this in mind, our contributions can be summarised as follows:

- We formally introduce the problem of evading link prediction algorithms;
- We prove that the problem is NP-complete given nine link prediction algorithms that are widely studied in the literature;
- We present polynomial-time heuristics that can easily be implemented by members of the general public on existing social media; those heuristics are empirically evaluated, both on real-life as well as randomly-generated networks, with varying levels of success.

2. RELATED WORK

In recent years, various research questions pertaining to privacy and security on social networks have attracted a lot of attention in the literature. A comprehensive overview of the main on-line threats and possible countermeasure can be found in the work by Fire et al. [15]. In addition to the more common threats such as malware, phishing attacks, spammers, cross-site scripting, and internet frauds, the authors discuss more elaborate schemes such as clickjacking, de-anonymization attacks, face recognition attacks, identity clone attacks, inference attacks, information and location leakages, fake profiles, and socware. The available countermeasures that are explicitly dedicated to the protection of privacy and security in social networks include both commercial software packages—which are predominantly focused on the most common threats listed above—as well as academic solutions, designed for the relatively newer, more elaborate schemes. Those latter solutions include, among others, applications that provide a more user-friendly access to the privacy settings in social media, and specialized software for detecting cloned and fake profiles. Additionally, Fire et al. recommend removing unnecessary personal information from social media, declining friendship requests from strangers, and refraining from publishing one's current location. While our study is complementary to all such security measures, it is based on fundamentally different approach, where the evaders *strategically* modify their social connections in order to mislead the seeker.

In this article we focus on the widely-studied class of local link prediction algorithms, which are based on similarity indices (see Section 3.2 for more details). The main advantage of such algorithms is their scalability. However, various other approaches have been proposed in the literature, *e.g.*, using supervised learning methods [26], or focusing on hierarchical structures in the network and how they can improve the link prediction process [11]. A general survey of the literature can be found in the work by Lü and Zhou [28]. Another survey, more focused on social rather than general networks, is given by Al Hasan and Zaki [3].

Apart from privacy and personal security concerns, the issue of evading link prediction algorithms is also important in the context of covert network analysis. In fact, various link prediction algorithms have been applied to the analysis of criminal networks [33]. Attention has been paid particularly to the co-offending networks, where the aim is to predict which criminals might be accomplices, based on the history of their previous cooperation [8, 39]. A number of authors advocated the use of link prediction algorithms to the analysis of terrorist networks [10, 42, 2]. Our work contributes to this line of research by modeling how criminals and terrorists could evade social network analysis tools.

A similar topic of how to evade social network analysis tools is studied by Waniek et al. [40] who consider evading centrality measures and community detection algorithms.

Our work can also be related to measuring the robustness of link prediction algorithms [45], *i.e.*, the investigation of how the result of such an algorithm changes when some of the network's edges are added or removed. The essential difference between this line of work and ours is that the choice of which edges to be added or removed in our case is strategic, *i.e.*, focused on hiding certain connections. In contrast, when measuring robustness, the addition or removal of edges is carried out randomly.

3. PRELIMINARIES

In this section we describe the necessary background, as well as the basic concepts and notation used throughout the article.

3.1 Network Notation

Let $G = (V, E) \in \mathbb{G}$ be a network, where V is the set of nodes, E is the set of edges, and $\mathbb{G}$ is the set of all possible networks. The edge between a pair of nodes, $v, w \in V$, will be denoted by (v, w). We will restrict our attention to *undirected* networks, and so we will not distinguish between (v, w) and (w, v). Furthermore, we do not consider self-loops, *i.e.*, edges of the form (v, v). We will use the term "*non-edge*" to refer to any edge, $(v, w) : v \neq w$, that is *not* in E; the set of all non-edges will be denoted by $\bar{E}$. That is, $\bar{E} = \{(v, w) : v, w \in V, v \neq w, (v, w) \notin E\}$. For any node, $v \in V$, the set of neighbours of v in G will be denoted by $N_G(v)$, *i.e.*, $N_G(v) = \{w \in V : (v, w) \in E\}$. By $N_G(v, w)$ we will denote the set of common neighbours of v and w, *i.e.*, $N_G(v, w) = N_G(v) \cap N_G(w)$. The *degree* of v will be denoted by $d_G(v)$, *i.e.*, $d_G(v) = |N_G(v)|$. Whenever it is clear from the context, we will omit the graph subscript, *e.g.*, we will often write $N(v)$ and $d(v)$ instead of $N_G(v)$ and $d_G(v)$, respectively.

3.2 Link Prediction Algorithms

For any given pair of nodes, a link prediction algorithm estimates the likelihood that there exist a not-yet-discovered edge between those two nodes, or that an edge will form between the two nodes in the future [16]. Many link prediction algorithms are based on *similarity indices*, also known as *kernels* [37]. In particular, given a network, $G = (V, E)$, a *similarity index* is a function, $s : \bar{E} \to \mathbb{R}$, which assigns to each non-edge, $e \in \bar{E}$ a score indicating the probability of e forming in the future, or the probability of e being a not-yet-discovered edge in the network [16]. For any similarity index, s, and any non-edge, $(v, w) \in \bar{E}$, we will often write $s(v, w)$ instead of $s((v, w))$ to improve readability.

An important class of link prediction algorithms are those based on *local* similarity indices, *i.e.*, indices that account for only local information pertaining to the non-edge in question. As such, the algorithms based on local similarity indices are typically computationally tractable and can be used for efficient analysis of even large networks [28]. In this article, we will consider the following local similarity in-

Table 1: Formulae of the local similarity indices.

Similarity Index	Score
Common Neighbours [31]	$s^{\mathrm{CN}}(v,w) = \lvert N(v,w)\rvert$
Salton [35]	$s^{\mathrm{Sal}}(v,w) = \frac{\lvert N(v,w)\rvert}{\sqrt{d(v)d(w)}}$
Jaccard [19]	$s^{\mathrm{Jac}}(v,w) = \frac{\lvert N(v,w)\rvert}{\lvert N(v)\cup N(w)\rvert}$
Sørensen [38]	$s^{\mathrm{Sør}}(v,w) = \frac{2\lvert N(v,w)\rvert}{d(v)+d(w)}$
Hub Promoted [34]	$s^{\mathrm{HPI}}(v,w) = \frac{\lvert N(v,w)\rvert}{\min(d(v),d(w))}$
Hub Depressed [34]	$s^{\mathrm{HDI}}(v,w) = \frac{\lvert N(v,w)\rvert}{\max(d(v),d(w))}$
Leicht-Holme-Newman [23]	$s^{\mathrm{LHN}}(v,w) = \frac{\lvert N(v,w)\rvert}{d(v)d(w)}$
Adamic-Adar [1]	$s^{\mathrm{AA}}(v,w) = \sum_{u\in N(v,w)} \frac{1}{\log(d(u))}$
Resource Allocation [47]	$s^{\mathrm{RA}}(v,w) = \sum_{u\in N(v,w)} \frac{1}{d(u)}$

dices, taken from the survey by Lü and Zhou [28][1]: Common Neighbours [31], Salton [35], Jaccard [19], Sørensen [38], Hub Promoted [34], Hub Depressed [34], Leicht-Holme-Newman [23], Adamic-Adar [1] and Resource Allocation [47]. Table 1 presents the formula of each index. Importantly, we will denote the set of all the aforementioned similarity indices as $\mathcal{S}$. That is,

$$\mathcal{S} = \{s^{\mathrm{CN}}, s^{\mathrm{Sal}}, s^{\mathrm{Jac}}, s^{\mathrm{Sør}}, s^{\mathrm{HPI}}, s^{\mathrm{HDI}}, s^{\mathrm{LHN}}, s^{\mathrm{AA}}, s^{\mathrm{RA}}\}.$$

3.3 Performance Evaluation Metrics

The most common metrics for evaluating the performance of a similarity index are: *Area under ROC curve (AUC)* [14] and *Area under Precision-Recall curve (PR)* [29]. To compute any of those metrics for a given similarity index, s, the set of edges is typically divided into a training set, T, and a probe set, Q, such that $T \cup Q = E$ and $T \cap Q = \emptyset$. The network (V, T) serves as input to the similarity index, s, which produces a ranking of the elements of $Q \cup \bar{E}$. After that, one can use either AUC or PR to express the quality of this ranking using a single number. To explain how this number is computed, we need some additional notation. Let σ_k denote the top k elements of $Q \cup \bar{E}$ when ranked according to s, and let $\widehat{n} = |Q \cup \bar{E}|$. Next, we explain how the aforementioned number is computed using either AUC or PR, before explaining an alternative metric called *Average Precision*, denoted by AP.

Area under ROC curve (AUC): For any given T and Q, $AUC(E, Q)$ is the area under the plot consisting of the following points:

$$\left\{\left(\frac{|\sigma_k \cap \bar{E}|}{|\bar{E}|}, \frac{|\sigma_k \cap Q|}{|Q|}\right)\right\}_{k=1}^{\widehat{n}}$$

$AUC(E, Q)$ can be interpreted as the probability that s assigns a higher score to a randomly chosen edge from Q than to a randomly chosen non-edge from $\bar{E}$ (ties are broken at random). That is:

$$AUC(E,Q) = \frac{|\{(e_1,e_2) \in Q \times \bar{E} : s(e_1) > s(e_2)\}|}{|Q||\bar{E}|} + \frac{\frac{1}{2}|\{(e_1,e_2) \in Q \times \bar{E} : s(e_1) = s(e_2)\}|}{|Q||\bar{E}|}$$

Area under Precision-Recall curve (PR): For any given T and Q, $PR(E, Q)$ is the area under the plot consisting of the following points:

$$\left\{\left(\frac{|\sigma_k \cap Q|}{|Q|}, \frac{|\sigma_k \cap Q|}{k}\right)\right\}_{k=1}^{\widehat{n}}$$

Average Precision (AP): Since the PR value is not well-defined for plots that are not continuous, we use for such plots the *average precision*, AP, described as one of the most robust by Boyd *et al.* [7]. Taking into account the possibility of equal scores, the average precision value becomes:

$$AP(E,Q) = \frac{1}{|Q|} \sum_{e_0 \in Q} \frac{|T^+ \cap Q| + 1 + \frac{1}{2}|T^= \cap Q|}{|T^+| + 1 + \frac{1}{2}|T^=|}$$

where $T^+ = \{e \in Q \cup \bar{E} : s(e) > s(e_0)\}$ and $Q^+ = \{e \in Q \cup \bar{E} : s(e) > s(e_0)\}$ and $T^= = \{e \in (Q \cup \bar{E}) \setminus \{e_0\} : s(e) = s(e_0)\}$.

4. THEORETICAL ANALYSIS

Definitionfl‹Evading Link Prediction. This problem is defined by a tuple, $(G, s, f, H, b, \widehat{A}, \widehat{R})$, where $G = (V, E) \in \mathbb{G}$ is a network, $s : \bar{E} \to \mathbb{R}$ is a similarity index, $f \in \{AUC, AP\}$ is a performance evaluation metric, $H \subset E$ is the set of edges[2] to be hidden, $b \in \mathbb{N}$ is a budget specifying the maximum number of edges that can be modified (i.e., added or removed), $\widehat{A} \subseteq \bar{E}$ is the set of edges that can be added, and $\widehat{R} \subseteq E \setminus H$ is the set of edges that can be removed. The goal is then to identify two sets of edges, $A^* \subseteq \widehat{A}$ and $R^* \subseteq \widehat{R}$, such that the resulting set, $E^* = (E \cup A^*) \setminus R^*$, is in:

$$\underset{E' \in \left\{(E \cup A) \setminus R \,:\, A \subseteq \widehat{A},\ R \subseteq \widehat{R},\ |A|+|R| \leq b\right\}}{\arg\min} f(E', H).$$

Let us comment on the above definition. We introduced the sets $\widehat{A}$ and $\widehat{R}$ to model scenarios in which the evaders can only modify the network in a limited manner. This could be the case, for example, when certain connections are more costly to establish than others, or when the evaders want to avoid removing certain critical edges, or avoid connecting to certain individuals. Likewise, we introduced the "budget",

[1]The only local similarity index in [28] that is excluded from our analysis is the Preferential Attachment Index. This index is based on the assumption that the degree distribution follows a power law—an assumption that does not hold for many of the networks on which we conduct our experiments.

[2]Note that, from the perspective of the evaders, H is a subset of E. In contrast, from the perspective of the seeker, we have: $H \subseteq \bar{E}$. Throughout the article, we will assume that $H \subseteq E$.

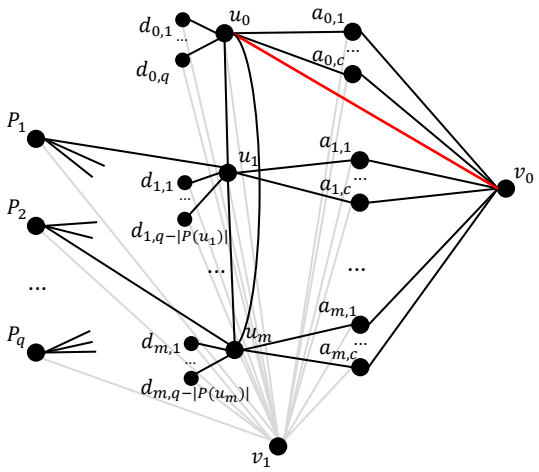

Figure 1: An illustration of the $\Gamma(c, P)$ network. Edges connecting v_1 with other nodes are grayed out to improve readability. The red edge is the one to be hidden.

b, to model scenarios in which the evaders can only perform a limited number of modifications.

Having defined our computational problem, we are now ready to present our main hardness results. Basically, we will prove that the problem of evading link prediction is NP-complete for all the similarity indices described in Table 1 and for both the AUC and AP metrics. To this end, we need to first define a certain network, denoted by $\Gamma(c, P)$, which will be used later on in our NP-completeness proofs.

Definition 1. The $\Gamma(c, P)$ Network. Let $U = \{u_1, \ldots, u_m\}$ be a set of m elements, and let $P = \{P_1, \ldots, P_q\}$ be a cover of U containing q subsets that are each smaller than U. That is, $\forall_i P_i \subset U$ and $\bigcup_{P_i \in P} P_i = U$. Then, given a constant, $c \in \mathbb{N}$, the network $\Gamma(c, P)$ is created as follows:

- **The set of nodes:**: For every $P_i \in P$, we create a single node, denoted by P_i. Moreover, for every $u_i \in \{u_0, \ldots, u_m\}$, we create a node denoted by u_i, as well as c nodes denoted by $a_{i,1}, \ldots, a_{i,c}$, and $q - |P(u_i)|$ nodes denoted by $d_{i,1}, \ldots, d_{i,q-|P(u_i)|}$, where $P(u_i) = \{P_j \in P : u_i \in P_j\}$. Additionally, we create three nodes, v_0, v_1, and u_0, as well as c nodes, $a_{0,1}, \ldots, a_{0,c}$, and q nodes, $d_{0,1}, \ldots, d_{0,q}$.
- **The set of edges:** First, we create the edge (u_0, v_0). After that, for every $P_j \in P$ we create the edge (P_j, v_1), as well as the edges (P_j, u_i) for every $u_i \in P_j$. Moreover, for every $u_i \in U$ we create the edge (u_i, v_1), as well as the edges (u_i, u_j) for every $u_j \in \{u_{i+1}, \ldots, u_m\}$ (this way, the nodes in $\{u_0, \ldots, u_m\}$ form an $(m+1)$-clique). Furthermore, for every $d_{i,j}$ we create the edges $(d_{i,j}, u_i)$ and $(d_{i,j}, v_1)$. Finally, for every $a_{i,j}$ we create the edges $(a_{i,j}, u_i)$, $(a_{i,j}, v_0)$ and $(a_{i,j}, v_1)$.

Figure 1 provides an illustration of the $\Gamma(c, P)$ network. The following lemma is built around this network.

Lemma 1. Consider a $\Gamma(c, P)$ network for which $m \geq 5$ and $|P_i| = 3$ for all $P_i \in P$. Furthermore, let $\widehat{A} = \{(P_i, v_0) : P_i \in P\}$, and for every $A \subseteq \widehat{A}$, let $P_A = \{P_i \in P : (P_i, v_0) \in A\}$, and let $P_A(u_j) = \{P_i \in P_A : u_j \in P_i\}$. Now, consider the network $(V, E) = \Gamma(c, P)$. For every similarity index, $s \in \mathcal{S}$, there exists some constant, $c \in \mathbb{N}$, such that:

(a) for every network, $(V, E \cup A) : A \subseteq \widehat{A}$, and every non-edge, $(u_i, v_0) : i \in \{0, \ldots, m\}$, we have $s(u_i, v_0) = s(u_0, v_0)$ if $P_A(u_i) = \emptyset$, and $s(u_i, v_0) > s(u_0, v_0)$ otherwise;

(b) for every non-edge, $(P_i, v_0) : i \in \{0, \ldots, m\}$, we have:

- $\forall_{(V,E\cup A):A\subseteq\widehat{A},(P_i,v_0)\notin A}\ s(P_i, v_0) < s(u_0, v_0)$.

(c) for every non-edge, $e \in \bar{E} \backslash \{(u_0, v_0), \ldots, (u_m, v_0), (P_1, v_0), \ldots, (P_q, v_0)\}$, either:

- $\forall_{(V,E\cup A):A\subseteq\widehat{A}}\ s(e) > s(u_0, v_0)$, or
- $\forall_{(V,E\cup A):A\subseteq\widehat{A}}\ s(e) = s(u_0, v_0)$, or
- $\forall_{(V,E\cup A):A\subseteq\widehat{A}}\ s(e) < s(u_0, v_0)$.

Sketch of proof. First of all, note that the following holds:

- $\forall_{(V,E\cup A):A\subseteq\widehat{A},(P_i,v_0)\notin A}\ \forall_i\ N(P_i, v_0) = \emptyset$;
- $\forall_{(V,E\cup A):A\subseteq\widehat{A}}\ \forall_{i,j}\ N(v_0, d_{i,j}) = \emptyset$.

This implies that for every similarity index, $s \in \mathcal{S}$, we have:

- $\forall_{(V,E\cup A):A\subseteq\widehat{A},(P_i,v_0)\notin A}\ \forall_i\ s(P_i, v_0) = 0$;
- $\forall_{(V,E\cup A):A\subseteq\widehat{A}}\ \forall_{i,j}\ s(v_0, d_{i,j}) = 0$.

One can also verify that for every $s \in \mathcal{S}$ the following holds:

- $\forall_{(V,E\cup A):A\subseteq\widehat{A}}\ s(u_0, v_0) > 0$.

This implies that point (b) holds, and that point (c) holds for every non-edge of the form $(v_0, d_{i,j})$. We still need to prove the correctness of point (a), as well as the correctness of point (c) for every non-edge of the form:

(i) (v_0, v_1)
(ii) (u_i, P_j) for $u_i \notin P_j$
(iii) $(u_i, a_{j,l})$ for $i \neq j$
(iv) $(u_i, d_{j,l})$ for $i \neq j$
(v) (P_i, P_j) for $i \neq j$
(vi) $(P_i, a_{j,l})$
(vii) $(P_i, d_{j,l})$
(viii) $(a_{i_1,j_1}, a_{i_2,j_2})$
(ix) $(a_{i_1,j_1}, d_{i_2,j_2})$
(x) $(d_{i_1,j_1}, d_{i_2,j_2})$

To this end, first note that the following holds for every graph, $(V, E \cup A) : A \subseteq \widehat{A}$, and every $a_{i,j}$, $d_{i,j}$, P_i in that graph:

- $d(a_{i,j}) = 3$ (because $a_{i,j}$ is connected to v_0, v_1 and u_i);
- $d(d_{i,j}) = 2$ (because $d_{i,j}$ is connected to v_1 and u_i);
- $4 \leq d(P_i) \leq 5$ (because P_i is connected to v_1 and to every $u_j \in P_i$, where we assume in Lemma 1 that $|P_i| = 3$. Also, if $P_i \in A$, then P_i would also be connected to v_0 in $(V, E \cup A)$).

Also note that $u_0 \notin P_i$ for every $P_i \in P$. Therefore, for any given $A \subseteq \widehat{A}$, we have: $P_A(u_0) = 0$. In what follows, we will use the aforementioned facts without referring back to them.

Next, we present the proof for only one similarity index, namely *Common Neighbours*, s^{CN}, which simply counts the number of nodes that are neighbors to both v and w. The proofs for all the similarity indices in $\mathcal{S}$ follow a similar reasoning.

In particular, for s^{CN}, we choose $c = 6$. Then, to prove the correctness of point (a), it suffices to note that for every $(V, E \cup A) : A \subseteq \widehat{A}$ we have:

$$s^{\text{CN}}(u_j, v_0) = 6 + |P_A(u_j)|, \quad \forall u_j \in \{u_0, \ldots, u_m\}.$$

Moving on to point (c), note that $\forall_{(V,E\cup A):A\subseteq\widehat{A}}\, s^{\text{CN}}(u_0, v_0) = 6$ and that the following holds for every $(V, E \cup A) : A \subseteq \widehat{A}$:

(i) $s^{\text{CN}}(v_0, v_1) = 6(m + 1) + |A| > s^{\text{CN}}(u_0, v_0)$, because the common neighbours of u_0 and v_1 are all the nodes of the form $a_{i,j}$ and all the nodes of the form P_i where $(P_i, v_0) \in A$;

(ii) $s^{\text{CN}}(u_i, P_j) \leq 4 < s^{\text{CN}}(u_0, v_0)$, because the common neighbours of u_i and P_j consist of v_1 and every $u_l \in P_j : l \neq i$ (note that we assume in Lemma 1 that $|P_j| = 3$, and u_i may or may not be an element of P_j);

(iii) $s^{\text{CN}}(u_i, a_{j,l}) = 2 < s^{\text{CN}}(u_0, v_0)$, because the common neighbours of u_i and $a_{j,l}$ are v_1 and u_j;

(iv) $s^{\text{CN}}(u_i, d_{j,l}) = 2 < s^{\text{CN}}(u_0, v_0)$, because the common neighbors of u_i and $d_{j,l}$ are v_1 and u_j;

(v) $s^{\text{CN}}(P_i, P_j) \leq 5 < s^{\text{CN}}(u_0, v_0)$, because the common neighbours of P_i and P_j consist of v_1, and possibly v_0 (if $\{(P_i, v_0), (P_j, v_0)\} \subseteq A$), as well as the every element in $P_i \cap P_j$ (there can be at most 3 such elements, since we assume in Lemma 1 that $|P_i| = |P_j| = 3$, and we place no restrictions on having $P_i = P_j$);

(vi) $s^{\text{CN}}(P_i, a_{j,l}) \leq 3 < s^{\text{CN}}(u_0, v_0)$, because the common neighbours of P_i and $a_{j,l}$ consist of v_1, and possibly v_0 (if $(P_i, v_0) \in A$) and possibly u_j (if $i = j$);

(vii) $s^{\text{CN}}(P_i, d_{j,l}) \leq 2 < s^{\text{CN}}(u_0, v_0)$, because the common neighbours of P_i and $d_{j,l}$ consist of v_1 and possibly u_j (if $i = j$);

(viii) $s^{\text{CN}}(a_{i_1,j_1}, a_{i_2,j_2}) \leq 3 < s^{\text{CN}}(u_0, v_0)$, because the common neighbours of a_{i_1,j_1} and a_{i_2,j_2} consist of v_1 and v_0 and possibly u_{i_1} (if $i_2 = i_1$);

(ix) $s^{\text{CN}}(a_{i_1,j_1}, d_{i_2,j_2}) \leq 2 < s^{\text{CN}}(u_0, v_0)$, because the common neighbours of a_{i_1,j_1} and d_{i_2,j_2} consist of v_1 and possibly u_{i_1} (if $i_2 = i_1$);

(x) $s^{\text{CN}}(d_{i_1,j_1}, d_{i_2,j_2}) \leq 2 < s^{\text{CN}}(u_0, v_0)$, because the common neighbours of d_{i_1,j_1} and d_{i_2,j_2} consist of v_1 and possibly u_{i_1} (if $i_2 = i_1$);

This concludes the sketch of the proof of Lemma 1. □

Having defined the problem of evading link prediction and the $\Gamma(c, P)$ network, and having proven the correctness of Lemma 1, we are now ready to present our main theorem.

THEOREM 1. *The problem of evading link prediction is NP-complete for both AUC (the Area under ROC curve) and AP (the Average Precision), and for every similarity index in $\mathcal{S}$, i.e., Common Neighbours (s^{CN}) [31], Salton (s^{Sal}) [35], Jaccard (s^{Jac}) [19], Sørensen ($s^{\text{Sør}}$) [38], Hub Promoted (s^{HPI}) [34], Hub Depressed (s^{HDI}) [34], Leicht-Holme-Newman (s^{LHN}) [23], Adamic-Adar (s^{AA}) [1] and Resource Allocation (s^{RA}) [47].*

PROOF. The problem is trivially in NP, since computing AUC and AP before and after the addition of a given $A^* \subseteq \widehat{A}$ and the removal of a given $R^* \subseteq \widehat{R}$ can be done in polynomial time for every similarity index in $\mathcal{S}$.

Next, we prove that the problem is NP-hard. To this end, we propose a reduction from the NP-complete *3-Set-Cover* problem. The decision version of this problem is defined by a universe $U = \{u_1, \ldots, u_m\}$ and a collection of sets $P = \{P_1, \ldots, P_q\}$ such that $\forall_i P_i \subset U \wedge |P_i| = 3$, where the goal is to determine whether there exist $x \leq b$ elements of P the union of which equals U. Let us assume that $m \geq 5$, as all other cases can be easily solved in polynomial time. Let us also introduce the following notation:

- $n = |\bar{E}|$
- $n^{(<)} = |\{e \in \bar{E} : s(e) < s(u_0, v_0)\}|$;
- $n^{(=)} = |\{e \in \bar{E} : s(e) = s(u_0, v_0)\}|$;
- $n^{(>)} = |\{e \in \bar{E} : s(e) > s(u_0, v_0)\}|$.

Now, for any given similarity index, $s \in \mathcal{S}$, consider the following instance of the problem of evading link prediction, $(G, s, f, H, b, \widehat{A}, \widehat{R})$, where:

- $G = (V, E) = \Gamma(c, P)$, where $c \in \mathbb{N}$ is chosen to be a constant for which points (a) and (c) of Lemma 1 are satisfied (the lemma states that such a constant exists);
- s is the similarity index under consideration;
- f is either the AUC or AP metric;
- $H = \{(u_0, v_0)\}$;
- b is the parameter of the *3-Set-Cover* problem (where the goal is to determine whether there exist $x \leq b$ elements of P the union of which equals U);
- $\widehat{A} = \{(P_i, v_0) : P_i \in P\}$;
- $\widehat{R} = \emptyset$.

For every $(V, E \cup A) : A \subseteq \widehat{A}$, we know from the definition of AUC that:

$$AUC(E \cup A, H) = \frac{|\{e \in \bar{E} \setminus A : s(e) < s(u_0, v_0)\}|}{|\bar{E} \setminus A|} + \frac{\frac{1}{2}|\{e \in \bar{E} \setminus A : s(e) = s(u_0, v_0)\}|}{|\bar{E} \setminus A|}. \quad (1)$$

We also know from the definition of AP that:

$$AP(E \cup A, H) = \frac{1}{|\Theta^+| + 1 + \frac{1}{2}|\Theta^=|}. \quad (2)$$

where $\Theta^+ = \{e \in \bar{E} \setminus A : s(e) > s(u_0, v_0)\}$ and $\Theta^= = \{e \in \bar{E} \setminus A : s(e) = s(u_0, v_0)\}$.
Observe that $\bar{E}$ is the set of non-edges in (V, E), whereas $\bar{E} \setminus A$ is the set of non-edges in $(v, E \cup A)$. Now, let $U_A = \{u_i : \exists_{P_j \in P_A} u_i \in P_j\}$. Point (b) of Lemma 1 implies that:

$$\{e \in \bar{E} \setminus A : s(e) < s(u_0, v_0)\} = \{e \in \bar{E} : s(e) < s(u_0, v_0)\} \cup A. \quad (3)$$

On the other hand, point (a) of Lemma 1 implies that:

$$\begin{aligned}&\left\{e \in \bar{E} \setminus A : s(e) = s(u_0, v_0)\right\}\\ &= \left\{e \in \bar{E} : s(e) = s(u_0, v_0)\right\} \setminus U_A. \end{aligned} \quad (4)$$

$$\begin{aligned}&\left\{e \in \bar{E} \setminus A : s(e) > s(u_0, v_0)\right\}\\ &= \left\{e \in \bar{E} : s(e) > s(u_0, v_0)\right\} \cup U_A. \end{aligned} \quad (5)$$

Equations (1), (3) and (4) imply that:

$$AUC(E \cup A, H) = \frac{n^{(<)} - |A| + \frac{n^{(=)} - |U_A|}{2}}{n - |A|} \quad (6)$$

On the other hand, equations (2) and (5) imply that:

$$\begin{aligned}AP(E \cup A, H) &= \frac{1}{n^{(>)} + |U_A| + 1 + \frac{n^{(=)} - |U_A|}{2}}\\ &= \frac{1}{n^{(>)} + 1 + \frac{n^{(=)} + |U_A|}{2}}. \end{aligned} \quad (7)$$

Equations (6) and (7) imply that both AUC and AP decrease with $|U_A|$. Thus, for each of these two metrics, an optimal choice of A is one that maximizes $|U_A|$; this happens when $U_A = U$. For any choice of A such that $U_A = U$, the following holds: $\forall_{u_j \in U} \exists_{(P_i, v_0) \in A} u_j \in P_i$. Such an optimal choice of A constitutes a solution to the problem of evading link prediction where $\widehat{R} = \emptyset$. It also corresponds directly to a solution to the *3-Set-Cover* problem. With this reduction, we conclude the proof of Theorem 1. □

5. HEURISTIC FOR EVADING LINK PREDICTION

Since finding the optimal solution to problem of evading link prediction turned out to be intractable, we now focus our efforts towards developing heuristic solutions that can be computed in polynomial time. Our heuristic algorithms (which will be presented in sections 5.2 and 5.3) are based on our analysis of how adding or removing a single edge affects the scores of non-edges; this analysis is presented in the following subsection.

5.1 Effects of Adding or Removing an Edge

By looking at the formulae of the different similarity indices in $\mathcal{S}$ (see Table 1), one can see that the score of every non-edge, $(v, w) \in \bar{E}$, depends solely on the following:

(i) the number of common neighbours of the non-edge.[3] More formally, $s(v, w)$ depends on $|N(v, w)|$. This observation affects every similarity index in $\mathcal{S}$. More precisely, for every $s \in \mathcal{S}$, the score $s(v, w)$ *increases* with $|N(v, w)|$;

(ii) the degree of each end of the non-edge, but only if both ends have at least one common neighbour. More formally, $s(v, w)$ depends on $d(v)$ and $d(w)$ if $N(v, w) \neq \emptyset$. This observation does not affect the similarity indices s^{CN} (*Common Neighbours*), s^{AA} (*Adamic-Adar*) and s^{RA} (*Resource Allocation*). As for every other similarity index, i.e., $s \in \mathcal{S} \setminus \{s^{\text{CN}}, s^{\text{AA}}, s^{\text{RA}}\}$, the score $s(v, w)$ *decreases* with $d(v)$ and $d(w)$ if $N(v, w) \neq \emptyset$.[4]

(iii) the degree of every common neighbour of the non-edge. More formally, $s(v, w)$ depends on $d(x) : x \in N(v, w)$. This observation only affects the indices s^{AA} (*Adamic-Adar*) and s^{RA} (*Resource Allocation*). To be more precise, for every $s \in \{s^{\text{AA}}, s^{\text{RA}}\}$, the score $s(v, w)$ *decreases* with $d(x) : x \in N(v, w)$.

Based on the above three observations, the *addition* of an edge, (u_1, u_2), can only affect the scores of three types of non-edges:

1. a non-edge $(u_1, x) : x \in N(u_2) \setminus N(u_1)$ (such a non-edge is affected by the addition of (u_1, u_2), which adds a new common neighbour of x and u_1; that common neighbour is u_2), or analogously a non-edge $(u_2, x) : x \in N(u_1) \setminus N(u_2)$. For every such non-edge, the addition of (u_1, u_2) *increases* every $s \in \mathcal{S}$; see observation (i);

2. a non-edge $(u_1, x) : N(u_1, x) \neq \emptyset$ (such a non-edge is affected by the addition of (u_1, u_2), which increases the degree of one end of the non-edge, namely u_1), or analogously a non-edge $(u_2, x) : N(u_2, x) \neq \emptyset$. For every such non-edge, the addition of (u_1, u_2) *decreases* every $s \in \mathcal{S} \setminus \{s^{\text{CN}}, s^{\text{AA}}, s^{\text{RA}}\}$; see observation (ii);

3. a non-edge $(x, y) : x, y \in N(u_1)$ (such a non-edge is affected by the addition of (u_1, u_2), which increases the degree of a common neighbour of x and y, namely u_1), or analogously a non-edge $(x, y) : x, y \in N(u_2)$. For every such non-edge, the addition of (u_1, u_2) *decreases* every $s \in \{s^{\text{AA}}, s^{\text{RA}}\}$; see observation (iii).

Conversely, the *removal* of an edge, (u_1, u_2), can only affect the scores of:

1. a non-edge $(u_1, x) : x \in N(u_2) \setminus N(u_1)$, or a non-edge $(u_2, x) : x \in N(u_1) \setminus N(u_2)$; for every such non-edge, the removal of (u_1, u_2) *decreases* every $s \in \mathcal{S}$;

2. a non-edge $(u_1, x) : N(u_1, x) \neq \emptyset$, or a non-edge $(u_2, x) : N(u_2, x) \neq \emptyset$; for every such non-edge, the removal of (u_1, u_2) *increases* every $s \in \mathcal{S} \setminus \{s^{\text{CN}}, s^{\text{AA}}, s^{\text{RA}}\}$;

3. a non-edge $(x, y) : x, y \in N(u_1)$, or a non-edge $(x, y) : x, y \in N(u_2)$; for every such non-edge, the removal of (u_1, u_2) *increases* every $s \in \{s^{\text{AA}}, s^{\text{RA}}\}$.

5.2 The OTC Heuristic

Based on our analysis in Section 5.1, we propose a heuristic algorithm that "hides" the edges in H by increasing the similarity scores of the non-edges in $\bar{E}$. This way, if the seeker were to run a link-prediction algorithm, whereby all the non-edges in $\bar{E}$ are ranked according to some similarity index, then our heuristic would increase the ranking of the non-edges in $\bar{E}$, thereby reducing the likelihood that a non-edge $e \in H$ is deemed to be a not-yet-discovered edge, or the likelihood that e is deemed likely to form in the future.

In particular, we propose what we call the *Open-Triad-Creation* (OTC) algorithm, which is based on *adding* edges to the network. The purpose behind every such addition is to increase the number of *open triads*, thereby increasing the number of common neighbors of the missing edge in each such triad. The rationale comes from the fact that by increasing the number of common neighbours of a non-edge,

[3] By "common neighbours of a non-edge", we mean the common neighbours of both ends of that non-edge.

[4] Regarding s^{Jac}—the *Jaccard* index [19]—note that: $|N(v) \cup N(w)| = d(v) + d(w) - |N(v, w)|$.

ALGORITHM 1: The *Open-Triad-Creation* (OTC) algorithm

Input: A network, (V, E), a budget, $b \in \mathbb{N}$, a set of edges that can be added, $\widehat{A} \subseteq \bar{E}$, and a set of edges to be hidden, $H \subset E$.

1 $A' \leftarrow \Big\{(v,w) \in \widehat{A} : \big(\exists u \in N(v) : (u,v) \in H\big) \vee \big(\exists u \in N(w) : (u,w) \in H\big)\Big\}$;

2 **for** $i = 1, \ldots, b$ **do**
3 **for** $(v,w) \in A'$ **do**
4 **if** $\exists_{u \in V}((v,u) \in E \setminus H \wedge (w,u) \in H) \vee ((w,u) \in E \setminus H \wedge (v,u) \in H)$ **then**
5 $\sigma_{(v,w)} \leftarrow -\infty$;
6 **else**
7 $\sigma_{(v,w)} \leftarrow \big|\big(N_{(V,E\setminus H)}(v) \cup N_{(V,E\setminus H)}(w)\big) \setminus N_{(V,E\setminus H)}(v,w)\big|$;
8 **end**
9 **end**
10 $(v^*, w^*) \leftarrow \arg\max_{(v,w) \in A'} \sigma_{(v,w)}$;
11 **if** $\sigma_{(v^*,w^*)} > -\infty$ **then**
12 $E \leftarrow E \cup (v^*, w^*)$;
13 **end**
14 **end**

its score increases according to *every similarity index* in $\mathcal{S}$ (see Section 5.1). Moreover, every time an edge is added to the network, the degree of each end increases. Now if any such end, $w \in V$, happens to also be an end of some edge $(w,u) \in H$, then the increase in $d(w)$ would decrease the score of (w,u) according to every similarity index in $\mathcal{S} \setminus \{s^{\text{CN}}, s^{\text{AA}}, s^{\text{RA}}\}$ (again see Section 5.1).

The pseudo code of the heuristic is presented in Algorithm 1. Specifically, in Line 1, out of all the non-edges that can be added (i.e., all the edges in $\widehat{A}$), the algorithm narrows the search to only the subset, $A' \subseteq \widehat{A}$, in which every non-edge has at least one end that belongs to some edge in H. In lines 3 to 9, the algorithm computes for every non-edge, $(v,w) \in A'$, a score, $\sigma_{(v,w)}$, which reflects the gain from adding (v,w) to the network. Here, lines 4 and 5 ensure that the algorithm does not increase the number of common neighbours of some edge in H. Line 7 counts the non-edges whose number of common neighbours will increase as a result of adding (v,w). Lines 10 to 12 chooses the non-edge with the highest score, and add it to the network. This entire process is repeated until the budget, b, runs out.

The complexity of a naive implementation of OTC is $\mathcal{O}(b|H||V|^2)$. In more detail, computing a score, $\sigma_{(v,w)}$, for each non-edge, $(v,w) \in A'$, can be done in time linear in $|V|$ for each of the $|H||V|$ non-edges. Searching for a non-edge in A' with the maximal score takes $b|H||V|$ operations. Finally, updating the scores after adding each of the b edges can be done in time linear in $|V|$.

5.3 The CTR Heuristic

We now propose an alternative heuristic that focuses decreasing the scores of the edges in H, rather than increasing the scores of the non-edges in $\bar{E}$ (which was the case with OTC). In particular, we propose what we call the *Closed-Triad-Removal* (CTR) algorithm, which is based on *removing* edges from the network (unlike OTC, which was based on *adding* edges). The purpose behind every such removal is to decrease the number of *closed triads* that contain an edge $(v,w) \in H$, thereby decreasing the number of common neighbours of (v,w). The rationale comes from the fact that by decreasing the number of common neighbours of (v,w), its score decreases according to *every similarity index* in $\mathcal{S}$ (see Section 5.1). However, this comes at a cost; it reduces the degree of one end of (v,w), which increases its score according to every similarity index in $\mathcal{S} \setminus \{s^{\text{CN}}, s^{\text{AA}}, s^{\text{RA}}\}$ (again see Section 5.1).

ALGORITHM 2: The *Closed-Triad-Removal* (CTR) algorithm

Input: A network, (V, E), a budget, $b \in \mathbb{N}$, a set of edges that can be removed, $\widehat{R} \subseteq E \setminus H$, and a set of edges to be hidden, $H \subset E$.

1 $R' \leftarrow \Big\{(v,w) \in \widehat{R} : \big(\exists u \in N(v) : (u,v) \in H\big) \vee \big(\exists u \in N(w) : (u,w) \in H\big)\Big\}$;

2 **for** $i = 1, \ldots, b$ **do**
3 **for** $(x,y) \in R'$ **do**
4 $\sigma_{(x,y)} \leftarrow 0$;
5 **end**
6 **for** $(v,w) \in H$ **do**
7 **for** $u \in N(v,w)$ **do**
8 **if** $(v,u) \in E \setminus H \wedge (w,u) \in E \setminus H$ **then**
9 **if** $(v,u) \in R'$ **then** $\sigma_{(v,u)} \leftarrow \sigma_{(v,u)} + 1$;
10 **if** $(w,u) \in R'$ **then** $\sigma_{(w,u)} \leftarrow \sigma_{(w,u)} + 1$;
11 **end**
12 **end**
13 **end**
14 $(v^*, w^*) \leftarrow \arg\max_{(v,w) \in R'} \sigma_{(v,w)}$;
15 **if** $\sigma_{(v^*,w^*)} > 0$ **then**
16 $E \leftarrow E \setminus (v^*, w^*)$;
17 **end**
18 **end**

The pseudo code of the CTR heuristic is presented in Algorithm 2. Specifically, in Line 1, out of all the edges that can be removed (i.e., all the edges in $\widehat{R}$), the algorithm narrows the search to only the subset, $R' \subseteq \widehat{R}$, in which every edge has at least one end that belongs to some edge in H. After that, in lines 3 to 13, the algorithm computes for every edge, $(x,y) \in R'$, a score, $\sigma_{(x,y)}$, which reflects the gain from removing (x,y) from the network. This is done by simply counting the number of closed triads that contain (x,y) and two other edges, one of which is in H. The edge with the greatest gain is chosen in line 14, and removed from the network in line 16.

The complexity of a naive implementation of CTR (which utilizes a hash table) is $\mathcal{O}(b|H||V|)$. This is because for every $(v,w) \in H$ the algorithm considers updates the score of every $(v,u) : u \in N(v)$ (there are at most $|V|$ such edges) and every $(w,u) : u \in N(w)$ (again there are at most $|V|$ such edges); this process is repeated b times.

Notice that when $|H| = \omega(\log(|V|))$, an implementation utilizing a *priority queue* is faster, with a complexity of $\mathcal{O}(|H||V| + b|V| \log(|H||V|))$.

6. EMPIRICAL EVALUATION

In this section, we describe the experiment design and present the simulation results for both the OTC heuristic and the CTR heuristic.

6.1 Datasets

We test our heuristics on randomly-generated networks as well as real networks. As for the former ones, they are

generated using the following standard models:

- *Scale-free* networks generated using the Barabasi-Albert model [5]. We denote such a network by $ScaleFree(n, d)$, where n is the number of nodes and d is the number of links added with each node;
- *Small-world* networks generated using the Watts-Strogatz model [41]. We denote such a network by $SmallWorld(n, d, p)$, where n is the number of nodes, d is the average degree and p is the rewiring probability;
- *Random graphs* generated using the Erdos-Renyi model [13]. We denote such a network by $RandomGraph(n, d)$, where n is the number of nodes and d is the expected average degree.

We now describe the real-life datasets used in our experiments:

- Facebook [24]—three fragments of the Facebook social network, containing 61 nodes, 272 edges (small fragment), 333 nodes, 2523 edges (medium fragment) and 786 nodes, 14027 edges (large fragment) respectively;
- Madrid terrorist network [17]—the network of terrorists behind the 2004 Madrid bombing, consisting of 70 nodes and 98 edges;
- Bali terrorist network [17]—the network of terrorists behind the 2002 Bali attack, consisting of 17 nodes and 63 edges;
- WTC terrorist network [22]—the network of terrorists behind the 9/11 attacks, consisting of 36 nodes and 64 edges;
- Zachary's Karate Club [43]—the social network of participants of a university karate club, consisting of 34 nodes and 78 edges;
- Les Misérables [21]—the network of co-occurances of characters in Victor Hugo's novel "Les Misérables", consisting of 77 nodes and 254 edges;
- Greek blogs [44]—a network of Greek political blogs, consisting of 142 nodes and 354 edges.

6.2 Experimental Design

Every experiment consists of a network, a link prediction algorithm, and a set of edges to be hidden, H. The network is either randomly generated using a certain model, or taken from our dataset of real-life networks. The algorithm is based on one of the similarity indices in $\mathcal{S}$ (see Section 3.2 for a formal definition of these indices). The set of edges to be hidden, i.e., H, consists of 10% of network's edges, chosen uniformly at random (we chose 10% at it is a typical size of the probe set when evaluating link predication algorithms [45, 28]). Finally, we run either OTC or CTR, with the budget being $b = |H|$. In each step of the algorithm, *i.e.*, at the end of every iteration in the main loop, we record the value of the AUC and AP metrics (see Section 3.3 for a formal definition of those metrics). Each such experiment is repeated 50 times, and the average results are reported along with the 95% confidence intervals. The same experiment is then repeated, but with $|H|$ consisting of 10 (rather than 10%) of the network's edges, again chosen uniformly at random.

6.3 Simulation Results

Figure 2 present the relative change in the AUC and AP value during the execution of the heuristics. Every point in a subplot represents the average over 50 simulations, with error bars representing the 95% confidence intervals. As can be seen, the heuristics are able to reduce the AUC and AP values, with varying levels of success.

Figures 3 to 6 evaluate the resilience of each algorithm given different networks and different heuristics, and that is in terms of AUC and AP. Results are shown here for the case where $b = |H| = \frac{|E|}{10}$. More specifically, in each figure, columns represent algorithms, rows represent networks, and the color of each cell represents the relative change in the evaluation metric (be it AUC or AP) after running a heuristic (be it OTC or CTR); this color represents the average result taken over 50 experiments. Rows and columns are sorted ascendingly based on the sum of the values therein. As can be seen, the degree to which an algorithm can be fooled depends heavily on the heuristic being used. For example, the similarity index *Leicht-Holme-Newman*[23], is the easiest to fool when using the OTC, but the hardest to fool when using CTR.

7. DISCUSSION & FUTURE WORK

In this article, we introduced and analysed the problem of evading link prediction algorithms. We proved that this problem is NP-complete for nine local link prediction algorithms that are widely studied in the literature. Given this hardness, we focused our efforts on developing scalable heuristics that can be applied by members of the general public; the heuristics do not require any involved technical knowledge nor massive processing power, and can readily be implemented on existing social media.

As a first step in the study of how to strategically evade link prediction algorithms, we focused on a basic, general model which can be extended in the future work in various directions. First, our model assumes that the evaders are strategic while the "*seeker*" (who is using the link prediction algorithms) is not, i.e., he or she is unaware of any potential strategic efforts by the evaders. While this is a rather strong assumption, we believe that it corresponds well to the current state of the art in the literature and industrial practices. This is because most existing social-network-analysis tools were built around the assumption that individuals or groups in a network do not act strategically to evade those tools on purpose. Our work can be considered as a step towards relaxing this assumption.

Second, our model assumes that the seeker's knowledge is restricted to the topology of the network. The motivation behind this assumption is twofold: (1) the most fundamental social network analysis tools—including the link prediction algorithms studied in this article—are all based solely on the topology of the network; (2) the exclusion of domain knowledge makes the model more general, as it can be applied to any network. Nevertheless, there may be cases where the seeker has additional, domain-dependent information that might be used in conjunction with social-network-analysis tools, e.g., as in the case of covert networks. In such cases, further investigation is still needed n order to understand the extent to which the evaders can protect themselves against the seeker.

Third, although our heuristic algorithms appear to be ef-

fective in practice, they do not provide any worst-case guarantees on solution quality. This is because our primary goal was to develop algorithms that are scalable and applicable by lay individuals who want to protect their privacy; such individuals typically do not know the topology of the entire network, nor do they have the ability the rewire links between two complete strangers. Undoubtedly however, scalability and applicability are achieved at the expense of solution quality. As such, there is room to develop more advanced algorithms that compute near-optimal solutions.

Fourth, we assume that the evaders do not know the exact algorithm that the seeker is using. As such, we focused on developing heuristics that perform reasonably well on a wide range of link prediction algorithms. Alternatively, one can focus on developing a dedicated heuristic for each algorithm. Every such specialized heuristic would have a narrower scope, but in return would likely outperform the more generic ones developed in this article.

Finally, another interesting direction is to investigate whether there exist special classes of networks for which the problem of evading link prediction in the optimal way can easily be solved.

REFERENCES

[1] L. A. Adamic and E. Adar. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003.

[2] M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. In *SDM06: workshop on link analysis, counter-terrorism and security*, 2006.

[3] M. Al Hasan and M. J. Zaki. A survey of link prediction in social networks. In *Social network data analytics*, pages 243–275. Springer, 2011.

[4] Y. Altshuler, N. Aharony, Y. Elovici, A. Pentland, and M. Cebrian. Stealing reality: when criminals become data scientists (or vice versa). In *Security and Privacy in Social Networks*, pages 133–151. Springer, 2013.

[5] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[6] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan. Mining email social networks. In *Proceedings of the 2006 international workshop on Mining software repositories*, pages 137–143. ACM, 2006.

[7] K. Boyd, K. H. Eng, and C. D. Page. Area under the precision-recall curve: Point estimates and confidence intervals. In *ECMLPKDD*, pages 451–466. Springer, 2013.

[8] P. L. Brantingham, M. Ester, R. Frank, U. Glässer, and M. A. Tayebi. Co-offending network mining. In *Counterterrorism and Open Source Intelligence*, pages 73–102. Springer, 2011.

[9] V. Cannistraci, G. Alanis-Lobato, and T. Ravasi. From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks. *Scientific reports*, 3, 2013.

[10] T. Carpenter, G. Karakostas, and D. Shallcross. Practical issues and algorithms for analyzing terrorist networks. In *Proceedings of the Western Simulation MultiConference*, 2002.

[11] A. Clauset, C. Moore, and M. E. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191):98–101, 2008.

[12] S. F. Crone and D. Soopramanien. Predicting customer online shopping adoption-an evaluation of data mining and market modelling approaches. In *DMIN*, pages 215–221, 2005.

[13] P. Erdős and A. Rényi. On random graphs i. *Publ. Math. Debrecen*, 6:290–297, 1959.

[14] T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[15] M. Fire, G. Katz, L. Rokach, and Y. Elovici. Links reconstruction attack. In *Security and Privacy in Social Networks*, pages 181–196. Springer, 2013.

[16] L. Getoor and C. P. Diehl. Link mining: a survey. *ACM SIGKDD Explorations Newsletter*, 7(2):3–12, 2005.

[17] B. Hayes. Connecting the dots can the tools of graph theory and social-network studies unravel the next big plot? *American Scientist*, 94(5):400–404, 2006.

[18] R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. Preventing private information inference attacks on social networks. *IEEE TKDE*, 25(8):1849–1862, 2013.

[19] P. Jaccard. *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz, 1901.

[20] N. F. Johnson, M. Zheng, Y. Vorobyeva, A. Gabriel, H. Qi, N. Velasquez, P. Manrique, D. Johnson, E. Restrepo, C. Song, and S. Wuchty. New online ecology of adversarial aggregates: Isis and beyond. *Science*, 352(6292):1459–1463, 2016.

[21] D. E. Knuth. *The Stanford GraphBase: a platform for combinatorial computing*, volume 37. Addison-Wesley Reading, 1993.

[22] V. E. Krebs. Mapping networks of terrorist cells. *Connections*, 24(3):43–52, 2002.

[23] E. A. Leicht, P. Holme, and M. E. Newman. Vertex similarity in networks. *Physical Review E*, 73(2):026120, 2006.

[24] J. Leskovec and J. J. Mcauley. Learning to discover social circles in ego networks. In *Advances in neural information processing systems*, pages 539–547, 2012.

[25] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[26] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla. New perspectives and methods in link prediction. In *Proceedings of the 16th ACM SIGKDD*, pages 243–252. ACM, 2010.

[27] J. Lindamood, R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. Inferring private information using social network data. In *Proceedings of the 18th international conference on World wide web*, pages 1145–1146. ACM, 2009.

[28] L. Lü and T. Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, 2011.

[29] C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*, volume 999. MIT Press, 1999.

[30] A. Mislove, B. Viswanath, K. P. Gummadi, and P. Druschel. You are who you know: Inferring user

profiles in online social networks. In *Proceedings of the Third ACM WSDM'10*, pages 251–260, New York, NY, USA, 2010. ACM.

[31] M. E. Newman. Clustering and preferential attachment in growing networks. *Physical review E*, 64(2):025102, 2001.

[32] A. Nordrum. Pro-ISIS Online Groups Use Social Media Survival Strategies to Evade Authorities, 2016.

[33] A. Potgieter, K. A. April, R. J. Cooke, and I. O. Osunmakinde. Temporality in link prediction: Understanding social complexity. *Emergence: Complexity and Organization*, 11(1):69, 2009.

[34] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A.-L. Barabási. Hierarchical organization of modularity in metabolic networks. *science*, 297(5586):1551–1555, 2002.

[35] G. Salton and M. J. McGill. Introduction to modern information retrieval. 1986.

[36] J. Scott. *Social network analysis*. Sage, 2012.

[37] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[38] T. Sørensen. {A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons}. *Biol. Skr.*, 5:1–34, 1948.

[39] M. A. Tayebi, L. Bakker, U. Glasser, and V. Dabbaghian. Locating central actors in co-offending networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 171–179. IEEE, 2011.

[40] M. Waniek, T. Michalak, T. Rahwan, and M. Wooldridge. Hiding individuals and communities in a social network. *arXiv preprint arXiv:1608.00375*, 2016.

[41] D. J. Watts and S. H. Strogatz. Collective dynamics of ï£¡small-worldï£¡ networks. *nature*, 393(6684):440–442, 1998.

[42] J. Xu and H. Chen. Untangling criminal networks: A case study. In *International Conference on Intelligence and Security Informatics*, pages 232–248. Springer, 2003.

[43] W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, pages 452–473, 1977.

[44] K. Zafiropoulos. Connectivity practices and activity of greek political blogs. *Future Internet*, 4(3):719–736, 2012.

[45] P. Zhang, X. Wang, F. Wang, A. Zeng, and J. Xiao. Measuring the robustness of link prediction algorithms under noisy environment. *Scientific reports*, 6, 2016.

[46] E. Zheleva and L. Getoor. To join or not to join: The illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 531–540, New York, NY, USA, 2009. ACM.

[47] T. Zhou, L. Lü, and Y.-C. Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71(4):623–630, 2009.

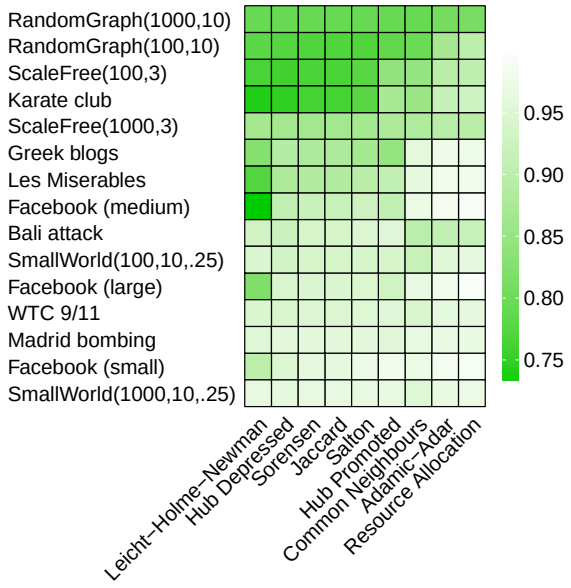

Figure 3: Relative changes in the AUC values after running the OTC heuristic, given $b = |H| = \frac{|E|}{10}$.

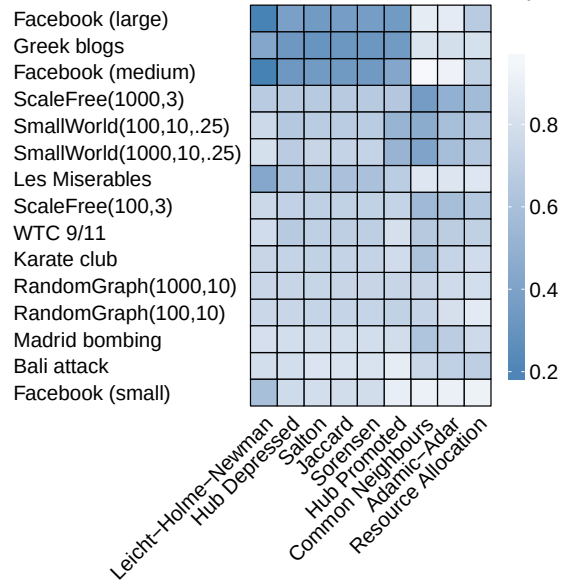

Figure 4: Relative changes in the AP values after running the OTC heuristic, given $b = |H| = \frac{|E|}{10}$.

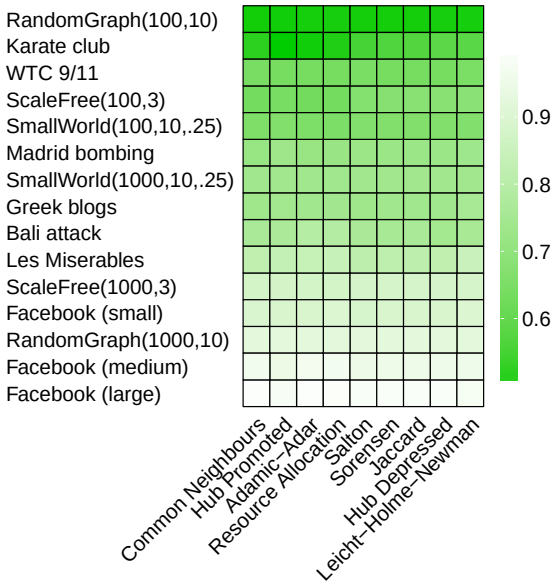

Figure 5: Relative changes in the AUC values after running the CTR heuristic, given $b = |H| = \frac{|E|}{10}$.

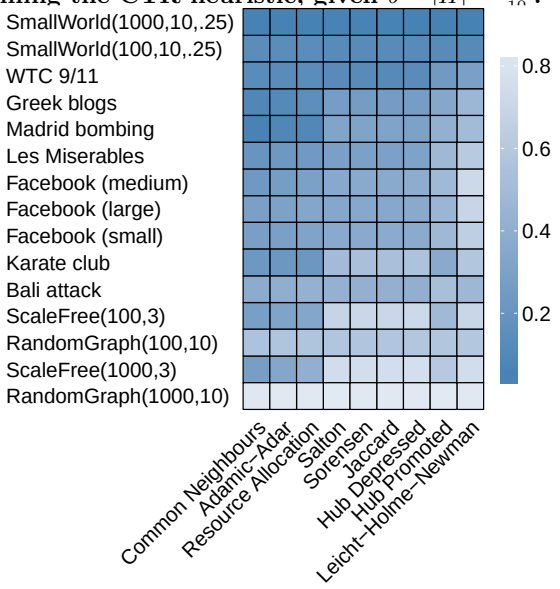

Figure 6: Relative changes in the AP values after running the CTR heuristic, given $b = |H| = \frac{|E|}{10}$.

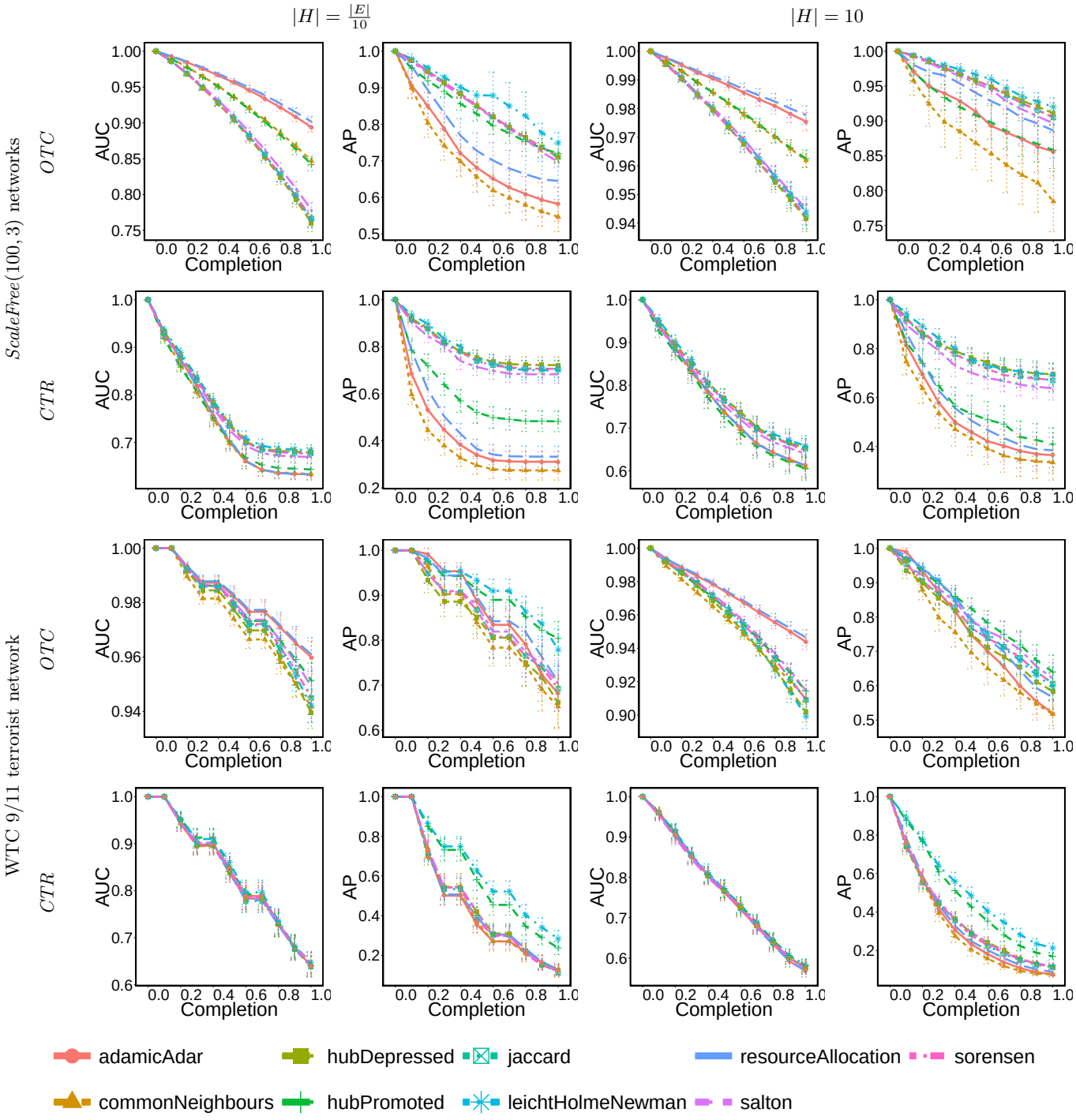

Figure 2: The Area under the ROC curve (AUC) and the Average Precision (AP) during the execution of *OTC* and *CTR* (with $b = |H|$). Results are shown for the WTC 9/11 terrorist network, and for an average over 50 Scale-free networks consisting of 100 nodes each (error bars representing the 95% confidence intervals).