Computationally Efficient Link Prediction in a Variety of Social Networks

MICHAEL FIRE, LENA TENENBOIM-CHEKINA, RAMI PUZIS, OFRIT LESSER, LIOR ROKACH, and YUVAL ELOVICI, Ben-Gurion University of the Negev

Online social networking sites have become increasingly popular over the last few years. As a result, new interdisciplinary research directions have emerged in which social network analysis methods are applied to networks containing hundreds of millions of users. Unfortunately, links between individuals may be missing either due to an imperfect acquirement process or because they are not yet reflected in the online network (i.e., friends in the real world did not form a virtual connection). The primary bottleneck in link prediction techniques is extracting the structural features required for classifying links. In this article, we propose a set of simple, easy-to-compute structural features that can be analyzed to identify missing links. We show that by using simple structural features, a machine learning classifier can successfully identify missing links, even when applied to a predicament of classifying links between individuals with at least one common friend. We also present a method for calculating the amount of data needed in order to build more accurate classifiers. The new *Friends measure* and *Same community* features we developed are shown to be good predictors for missing links. An evaluation experiment was performed on ten large social networks datasets: Academia.edu, DBLP, Facebook, Flickr, Flixster, Google+, Gowalla, TheMarker, Twitter, and YouTube. Our methods can provide social network site operators with the capability of helping users to find known, offline contacts and to discover new friends online. They may also be used for exposing hidden links in online social networks.

Categories and Subject Descriptors: I.2.6 [**Computing Methodologies**]: Artificial Intelligence—*Learning, Analogies*; J.4 [**Computer Applications**]: Social and Behavioral Sciences—*Sociology*

General Terms: Experimentation, Algorithms

Additional Key Words and Phrases: Link prediction, hidden links, social networks, supervised learning, training set size, imbalanced dataset, Academia.edu, DBLP, Facebook, Flickr, Flixster, Google+, Twitter, YouTube, TheMarker Cafe

ACM Reference Format:
Fire, M., Tenenboim-Chekina, L., Puzis, R., Lesser, O., Rokach, L., and Elovici, Y. 2013. Computationally efficient link prediction in a variety of social networks. ACM Trans. Intell. Syst. Technol. 5, 1, Article 10 (December 2013), 25 pages.
DOI: http://dx.doi.org/10.1145/2542182.2542192

1. INTRODUCTION

In recent years, online social networks have grown exponentially and offer individuals with similar personal and business interests the possibility of meeting and networking. Social networks create new opportunities to develop friendships, share ideas, and conduct business. Online social networking services, such as Facebook, Google+, Twitter, and Flickr, just to name a few, have become part of the daily life of millions of people

Authors' addresses: M. Fire (corresponding author), L. Tenenboim-Chekina, R. Puzis, O. Lesser, L. Rokach, and Y. Elovici, Telekom Innovation Laboratories at Ben-Gurion University of the Negev and Department of Information Systems Engineering, Ben Gurion University at the Negev, Be'er Sheva, 84105, Israel; email: mickyfi@gmail.com.

DOI: http://dx.doi.org/10.1145/2542182.2542192

around the world. The enormous growth of these networks has resulted in several research directions that examine the structural and behavioral properties of large-scale social networks.

Typically, researchers collect social network data using Web crawler software. Such crawlers, however, may sometimes collect only partial information. This is due to various causes, such as an attempt to access broken Web links, efforts by the social network operator to block various subscribers, communication failures, etc. Consequently, those depending upon the Web crawler might find themselves with only partial information about the set of links within a social network. In such cases, heuristic techniques for uncovering hidden links missed by the Web crawler are useful for completing the network structure. Detection of hidden links is also very practical for friend suggestion mechanisms used by online social networks. In such cases, hidden links may consist of existing social ties that have not yet been established in the particular social network [Liben-Nowell and Kleinberg 2007; Hasan et al. 2006; Doppa et al. 2009; Sa and Prudencio 2010; Song et al. 2009; Cukierski et al. 2011].

The problem of predicting the existence of hidden links or the creation of new ones in social networks is commonly referred to as the *Link Prediction* problem. Link prediction has many applications within and without the domain of social networks. For example, in bioinformatics, link prediction can be used to find interactions between proteins [Airoldi et al. 2006]; in e-commerce it can help build recommendation systems [Huang et al. 2005]; and in the security domain link prediction can assist in identifying hidden groups of terrorists or criminals [Hasan et al. 2006] or even compromise social networks users' privacy [Fire et al. 2012a]. Since the Link Prediction problem is relevant to different scenarios, several algorithms have been proposed in recent years. Some of the solutions are based on supervised machine learning and selecting relevant features, Bayesian probabilistic models, relational Bayesian networks, or linear algebraic methods. Further details on these approaches can be found in a thorough survey written by Zaki and Hasan [Hasan and Zaki 2011].

In this article, we investigate solutions to the Link Prediction problem based on machine learning classifiers trained on a set of easy-to-compute topological features. In addition to a set of well-known features, such as Katz measure [Katz 1953], *Common-friends* feature, and *Jaccard's coefficient* [Tan et al. 2005], we discuss two new topological features, later described in detail in Section 4.1. The first feature, the *Friends measure* introduced in Fire et al. [2011], is a variation of the Katz measure which estimates how well the friends of two users know each other. The second feature, the *same-community* feature introduced in this article, estimates whether two social network users are members of the same community in the social network. Classifiers based on these features are trained and evaluated using ten different social networks, five of which are directed: Academia.edu[1], Flickr[2], Google+[3], Twitter[4], and YouTube[5], and five are undirected: DBLP[6], Flixster[7], Facebook[8], Gowalla[9], and TheMarker Cafe[10]. Further information on these social networks is presented in Section 3. The quality

[1]http://www.academia.edu.
[2]http://www.flickr.com.
[3]https://plus.google.com.
[4]http://www.twitter.com.
[5]http://www.youtube.com.
[6]http://www.informatik.uni-trier.de/~ley/db/.
[7]http://www.flixster.com.
[8]http://www.facebook.com.
[9]http://gowalla.com/.
[10]http://cafe.themarker.com/.

of the features and of the classifiers based on these features was evaluated on two kinds of datasets created from these networks. The *easy* dataset includes false links that connect two randomly chosen vertices, while the *hard* dataset includes only links where the distance between the two foreseen neighbors is two (they already have at least one common friend). The evaluation results presented in Section 5 demonstrate that the *Friends measure* is a very effective feature for link prediction. Furthermore, in Section 4.2, we present a method for evaluating the training set size needed in order to build decision tree link prediction classifiers with near-maximal Area Under the ROC Curve (AUC).

The rest of the article is organized as follows. In Section 2, we give a brief overview of previous studies on different link prediction algorithms. We also describe several algorithms and definitions from graph theory and social networks analysis. In Section 3 we discuss online social networks whose structures were used in this study. Section 4 describes the experimental framework and the methods used for developing and evaluating a link predictor. The topological features used in this study are formally defined in Section 4.1. Section 5 presents our numeric results including the AUC, the contribution of each feature set, and the information gain value for the different features. Lastly, in Section 6 we present our conclusions.

2. RELATED WORK

2.1. Social Network Topology

Social networks have several well-known characteristics, such as power law degree distribution [Barabasi and Albert 1999], the small world phenomenon [Watts and Strogatz 1998], and community structure [Girvan and Newman 2002]. In this study, we make use of the fact that social networks have a community structure. Networks with community structure can be grouped into sets such that each set of vertices is densely and internally connected. There are many algorithms with different properties for finding communities in social networks. In this study, we used the Louvain method, a greedy algorithm that attempts to optimize the modularity of a partition of the network [Blondel et al. 2008]. We use this algorithm in order to split each of the tested social networks into disjoint communities. Consequently, we used these communities in order to calculate the *same-community* feature (see Section 4.1).

2.2. Supervised Learning and Link Prediction

In this study we focus on the common approach for solving the Link Prediction problem by using supervised learning algorithms. This approach was introduced by Liben-Nowell and Kleinberg in 2003 [Liben-Nowell and Kleinberg 2007], who studied the usefulness of graph topological features by testing them on five co-authorship networks datasets, each containing several thousands of authors. In 2006, Hasan et al. [2006] extended their work on the DBLP and BIOBASE co-authorship networks (each containing several hundreds of thousands of papers). Since its publication, the supervised learning approach has been implemented by several other researchers [Doppa et al. 2009; Sa and Prudencio 2010; Leskovec et al. 2010]. Most of the solutions proposed by these researchers were tested on bibliographic or on co-authorship datasets [Hasan et al. 2006; Liben-Nowell and Kleinberg 2007; Doppa et al. 2009; Sa and Prudencio 2010]. In 2009, Song et al. used matrix factorization to estimate the similarity between vertices in real-life social networks, such as Facebook and MySpace [Song et al. 2009]. In 2011, as a result of the issuing of the IJCNN social network challenge [Nachbar 2010], several papers were published. Each of these papers proposed a different method for predicting links in social networks. Narayanan et al. won the challenge by using a method that combined machine learning algorithms with

Table I. Social Networks Datasets

Network	Is Directed	Vertices Number	Links Number	Date
Academia	Yes	200,169	1,398,063	2011
DBLP	No	902,664	3,212,217	2011
Facebook	No	63,731	817,090	2009
Flickr	Yes	1,133,54	7,237,983	2010
Flixster	No	2,523,386	9,197,338	2010
Google+	Yes	211,187	1,506,896	2012
Gowalla	No	196,591	950,327	2010
Twitter	Yes	2,085	491,194	2009
TheMarker	No	65,953	1,572,684	2011
YouTube	Yes	1,138,499	4,945,382	2007

deanonymization [Narayanan et al. 2011]. Cukierski et al. won second place by extracting 94 distinct graph features and using the Random Forest algorithm in order to analyze the training data (consisting of several thousands of links) [Cukierski et al. 2011]. Recently, Fire et al. presented a method for predicting links inside communities; their methods used supervised learning ensemble classifiers that were constructed by only using a small training set which contained several hundreds of examples [Fire et al. 2012a].

In order to build an efficient classifier for link prediction, it is crucial to define and calculate a set of graph structural features. When dealing with large-scale graphs that may include millions of vertices and links, one of the challenges is the computationally intensive extraction of such features. For example, Facebook has more than one billion registered users and each month many new users are added [Facebook-Newsroom 2013]. Moreover, the power law degree distribution in social networks suggests that there are some individuals with a large number of connections (hubs). Computing local topological features on a subgraph consisting only of the friends of these individuals may be computationally intensive. In our previous work [Fire et al. 2011], we presented a method for solving the Link Prediction problem in large-scale online social networks, by using easy-to-compute topological features.

3. ONLINE SOCIAL NETWORKS DATASETS

In this study we apply link prediction classifiers to ten social network datasets (see Table I): Academia.edu, DBLP, Facebook, Flickr, Flixster, Google+, Gowalla, TheMarker, Twitter, and YouTube.

Academia.edu. Academia.edu is a platform for academics to share and follow research underway in a particular field or discipline. Academics upload their papers to share them with other academics in over 100,000 research areas. An Academia social network member may choose to follow any of the other members in this network, hence the directed nature of the links within this network.

DBLP. DBLP (Digital Bibliography and Library Project) is a computer science bibliography Web site that lists over 1.3 million articles on computer science. It tracks various well-known journals, such as ACM Transactions, as well as conference proceedings papers. The publication's co-authorship network is commonly used as a social network for empiric purposes.

Facebook. Facebook is a social networking service and Web site launched in February 2004. As of October 2012, nearly one billion users have active Facebook accounts. Since the friendship link between two members must be reciprocal, the existence of a link between member A and member B induces a mutual connection. We therefore refer to

Facebook's underlying friendship graph as an undirected one. The Facebook data used in this research was obtained from Viswanath et al. [2009].

Flickr. Flickr is an image and video hosting Web site that enables its members to socially interact via comments and follow each other by means of posted videos and images. Links between members do not require mutual approval since one may choose to follow any other visible member. The underlying graph that represents the Flickr social network is therefore regarded as directed. We obtained a subgraph of Flickr users from Nachbar [2010].

Flixster. Flixster is a social movie site which allows users to share movie ratings, discover new movies, learn about movies, and communicate with other. We evaluated our link prediction on a part of the Flixster social network download from the Social Computing Data Repository at Arizona State University [Zafarani and Liu 2009].

Google+. Google+ is a social networking service and Web site offered by Google. A Google+ member can add any other member to his circles, creating a directed social graph. We used a dedicated crawler to obtain this dataset which contained more than 250,000 users.

Gowalla. Gowalla is a location-based social networking Web site where users share their locations by checking-in. The friendship network is undirected and consists of 196,591 vertices and 950,327 links. This dataset was obtained from the work of Cho et al. [2011].

Twitter. Twitter is an online social networking and microblogging service that enables its users to send and read text-based posts. Each member can follow other members, creating a directed social network graph. We used the Twitter graph from the work of Cha et al. [2010]; however, we referred only to the elite of the Twitter subgraph that is, the subgraph of users with more than 30,000 followers [Avin et al. 2011].

TheMarker Cafe. TheMarker Cafe is an Israeli online social network site that allows its members to connect and interact. Since most of its members are Israelis, most interaction and communication among members is done in Hebrew. Due to the geographic and demographic nature of this network, it is smaller in scale compared to the other networks. TheMarker friendship connection is reciprocal, hence its underlying social structure may be represented as an undirected graph.

YouTube. YouTube is a popular video-sharing site that includes a social network. YouTube switched from directed links to a two-phase symmetric link creation process in 2007. In this article we use the dataset published by Mislove et al. [2007], which was collected while YouTube was still a directed graph.

In summary, DBLP, Facebook, Flixster, Gowalla, and TheMarker are undirected networks while Academia.edu, Google+, Flickr, Twitter, and YouTube are directed networks. Details of the datasets are summarized in Table I.

4. METHODS AND EXPERIMENTS

As our goal was to identify and predict a set of hidden links within a social network structure, we chose to use machine learning methods and develop link prediction classifiers which can predict the likelihood of a link existence in the social graph. We collected several social network datasets for this purpose. Some of the datasets (Academia.edu, Google+, and TheMarker) were collected using a dedicated Web crawling code that had been previously developed. The other datasets (from DBLP, Facebook, Flickr, Flixster, Gowalla, Twitter, and YouTube) were gathered from several online resources.

In order to use a machine learning algorithm, we first need to generate a training set that consists of many instances. Every training instance represents a possible candidate link. The target attribute is a binary attribute that indicates the existence or absence of a link. Since we focused on predicting links based on graph topology only, we extracted a set of features from the corresponding graphs of the social networks. These attributes were then fed into WEKA [Hall et al. 2009], a popular suite of machine learning software written in Java and developed at the University of Waikato, New Zealand. In addition to these features sets, the WEKA software also received, for each social network, a set of links included in the graph (also referred to as positive links) as well as a set of links not part of the social graph's original links set (referred to as negative links). Due to the large scale of the tested social network, we were able to create our link prediction classifiers with large quantities of positive and negative links as training examples. However, we wanted to study how different training set sizes affect the performance of our classifiers. Moreover, we also wanted to answer the question, "what is an optimal training set size for our classifiers?" In order to discover the optimal training set size, we constructed decision tree classifiers by using different training set sizes. In order to compare the various classifiers, the ROC curve, which is a standard technique for summarizing classifier performance over a range of trade-offs between True Positive Rates (TPR) and False Positive error Rates (FPR), was used. Each point in the curve corresponds to a particular cut-off, with the x-value as the false positive value (1-specificity) and the y-value as the sensitivity value. Points closer to the upper right corner correspond to lower cut-offs where points closer to the lower left corner correspond to higher cut-offs. The choice of the cut-off thus represents a trade-off between sensitivity and specificity. Ideally, one would want high values of both so that the model can well predict both the existence and the absence of links. Usually, a low cut-off gives a higher sensitivity. Conversely, a high cut-off gives a lower false positive rate, at the price of lower sensitivity. In terms of classifier comparison, the best curve is the one that is the leftmost, the ideal one coinciding with the y-axis. Thus, the AUC is an accepted performance metric for a ROC curve. The AUC range is [0,1]. The area under the diagonal is 0.5. This value represents a random classifier. On the other hand, a value of 1 represents an optimal classifier. The area under the ROC curve (AUC) has become the de facto performance measure for link prediction tasks because, unlike other accuracy measures, AUC is not influenced by the imbalance distribution of the classes [Menon and Elkan 2011].

By using the different classifier's AUC results together with a nonlinear regression model, we successfully constructed a nonlinear prediction model for a training set size that can give near-optimal AUC results. In regards to the goal of developing a preferred classifier, we then performed supervised learning using various machine learning algorithms. The rest of this section describes the set of features extracted from the social network graphs and the methods used to compose the training, find the optimal training set size, and test both sets and the machine learning algorithms examined.

4.1. Feature Extraction

This section describes all the features extracted and used during our experiment. Let $G = \langle V, E \rangle$ be the graph that represents the topological structure of a general social network. Links in the graph are denoted by $e = (u, v) \in E$, where $u, v \in V$. Our aim was to build a simple classifier using machine learning techniques so that for each two vertices (also referred to as vertices) $v, u \in V$ can predict whether or not the connection between u and v has a high probability of existing. Such a classifier could then be used to decide whether $(u, v) \in E$ or $(u, v) \notin E$.

The features for this classifier were extracted from the topological structure of the graph. For each link candidate for classification, we extracted a set of topological

features. These features assist in estimating the chances that a given link indeed exists in the graph. The link features depend on the type of the graph. If the graph is directed, then we can extract more features based on the direction of the links. Next we describe the topological features used to build the classifier.

4.1.1. Vertex Features. Let be $v \in V$, a neighborhood ($\Gamma(v)$) of v is defined as the set of v's friends, namely, vertices that are adjacent to v. In directed graphs the set of users that v follows (i.e., there is a directed link from v to these users) is different from the set of users that follow v (i.e., there is a directed link from them to v). We can therefore define outgoing ($\Gamma_{out}(v)$) and incoming ($\Gamma_{in}(v)$) neighborhoods. A neighborhood of v can also include v or exclude it from the set of vertices. Inclusion and exclusion of v in the neighborhood generate subgraphs that are very different with respect to their topological properties, as shown next. The following are the formal definitions of neighborhoods used to extract topological features.

$$\begin{aligned} \Gamma(v) &:= \{u|(u,v) \in E \text{ or } (v,u) \in E\}. \\ \Gamma_{in}(v) &:= \{u|(u,v) \in E\}. \\ \Gamma_{out}(v) &:= \{u|(v,u) \in E\}. \\ \Gamma^{+}(v) &:= \Gamma(v) \cup \{v\}. \end{aligned} \tag{1}$$

Based on the definition of neighborhoods, we can also define subgraphs induced by these neighborhoods. We defined the *neighborhood subgraphs* of v as

$$\begin{aligned} \textit{nh-subgraph}(v) &= \{(x,y) \in E|x,y \in \Gamma(v)\} \\ \textit{nh-subgraph}^{+}(v) &= \{(x,y) \in E|x,y \in \Gamma^{+}(v)\}. \end{aligned} \tag{2}$$

Using the preceding neighborhood definitions, we can create the following features for vertex v.

Vertex degree features. Using the neighborhoods definition we defined the *degree* of v as

$$d(v) = |\Gamma(v)|. \tag{3}$$

For a directed graph G, we defined the *in-degree*, *out-degree*, and *bi-degree* features as follows: $d_{in}(v) = |\Gamma_{in}(v)|$, $d_{out}(v) = |\Gamma_{out}(v)|$ and $d_{bi}(v) = |\Gamma_{in}(v) \cap \Gamma_{out}(v)|$. Using the degree features, we defined *degree-density* features as

$$\begin{aligned} \textit{in-degree-density}(v) &= \frac{d_{in}(v)}{d(v)}, \\ \textit{out-degree-density}(v) &= \frac{d_{out}(v)}{d(v)}, \\ \textit{bi-degree-density}(v) &= \frac{d_{bi}(v)}{d(v)}. \end{aligned} \tag{4}$$

In social networks, the degree feature represents the number of friends or followers that user v has.

Vertex subgraphs features. Using the *neighbor subgraphs*, we defined the following features that denote the number of links within the *neighbor subgraphs* for each vertex v.

$$\begin{aligned} \textit{Subgraph-Link-Number}(v) &= |\textit{nh-subgraph}(v)|, \\ \textit{Subgraph-Link-Number}(v)^{+} &= |\textit{nh-subgraph}^{+}(v)|. \end{aligned} \tag{5}$$

We can also define the density of each subgraph.

$$\begin{aligned} \textit{Density-nh-subgraph}(v) &= \frac{d(v)}{|\textit{nh-subgraph}(v)|}, \\ \textit{Density-nh-subgraph}^{+}(v) &= \frac{d(v)}{|\textit{nh-subgraph}^{+}(v)|}. \end{aligned} \tag{6}$$

If the G is directed, we can also calculate both the number of Strongly Connected Components (*SCC*) and the number of Weakly Connected Components (*WCC*) [Gibbons 1985] within the vertex subgraphs. The average number of vertices in these components can also be useful for classifying links (later shown in Section 5).

$$\begin{aligned} \textit{avg-scc}(v) &= \frac{d(v)}{\textit{scc}(\textit{nh-subgraph}(v))}, \\ \textit{avg-wcc}(v) &= \frac{d(v)}{\textit{wcc}(\textit{nh-subgraph}(v))}, \\ \textit{avg-scc}^{+}(v) &= \frac{d(v)}{\textit{scc}(\textit{nh-subgraph}^{+}(v))}. \end{aligned} \tag{7}$$

The number of weakly or strongly connected components in v's subgraph may provide an indication as to the number of different social groups to which v belongs.

4.1.2. Link Features. Let be $u, v \in V$ where $e = (u, v) \notin E$. Using the neighborhoods of u and v , one can extract several feature sets. These features include the number of Common friends u and v have (*Common friends(u,v)*), the number of distinct friends u and v have (*Total friends(u,v)*), the number of connections between u and v neighborhoods (*Friends measure(u,v)*) , and many other features which we define shortly. These features help us determine the likelihood that a connection between u and v exists.

Common friends. The Common friends of $u, v \in V$ refers to the size of the common set of friends that both u and v possess. The formal Common-friends definition for an undirected graph G is

$$\textit{common-friends}(u, v) = |\Gamma(v) \cap \Gamma(u)|. \tag{8}$$

For a directed graph G, we can also define *Common friends* based on the link direction: $\textit{common friends}_{in}(u, v) = |\Gamma_{in}(v) \cap \Gamma_{in}(u)|$, $\textit{common friends}_{out}(u, v) = |\Gamma_{out}(v) \cap \Gamma_{out}(u)|$, and $\textit{common friends}_{bi}(u, v) = |\Gamma_{bi}(v) \cap \Gamma_{bi}(u)|$. The relevance of the *Common friends* feature is very intuitive. It is expected that the larger the size of the common neighborhood, the higher the chances are that both vertices will be connected. The *Common friends* feature was widely used in past work on link prediction on several datasets and found to be very helpful [Huang et al. 2005; Liben-Nowell and Kleinberg 2007; Sa and Prudencio 2010; Song et al. 2009; Cukierski et al. 2011].

Total Friends. For two vertices u, v, we can define the number of distinct friends that u and v have together, namely: Let be $u, v \in V$, we define the *Total friends* of u, v to be the number of distinct neighbors u, v has.

$$\textit{total-friends}(u, v) = |\Gamma(u) \cup \Gamma(v)|. \tag{9}$$

Jaccard's coefficient: Jaccard's coefficient is a well-known feature for link prediction [Huang et al. 2005; Liben-Nowell and Kleinberg 2007; Sa and Prudencio 2010; Song et al. 2009; Cukierski et al. 2011]. The Jaccard's coefficient, which measures the similarity between sample sets, is defined as the size of the intersection divided by the size of the union of the sample sets. In our approach it indicates whether two social

network members (vertices in the corresponding graph) have a significant amount of Common friends, regardless of their *Total-friends* set size. A higher value of *Jaccard's coefficient* denotes a stronger tie between two friends. The coefficient defines the ratio between the number of *Common friends* and the number of *Total friends*, namely

$$jaccard's\text{-}coefficient(u, v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|}. \tag{10}$$

Transitive Friends. If G is a directed graph, we can calculate the number of *transitive friends* of u, v and v, u.

$$transitive\text{-}friends(u, v) = |\Gamma_{out}(u) \cap \Gamma_{in}(v)|. \tag{11}$$

Preferential attachment score. One well-known concept in social networks is that users with many friends tend to create more connections in the future. This is due to the fact that in some social networks, like in the finance world, the *rich get richer*. We estimate how "rich" our two vertices are by calculating the multiplication between the number of friends or followers each vertex has, namely,

$$preferential\text{-}attachment\text{-}score(u, v) = |\Gamma(u)| \cdot |\Gamma(v)|. \tag{12}$$

In several previous works on link prediction, this attribute was found to be a very significant feature [Hasan et al. 2006].

Katz measure. In 1953, Katz proposed a path-ensemble-based proximity measure [Katz 1953]. The *Katz measure* is a variant of the *shortest-path measure* (see Section 4.1.4). The idea behind the *Katz measure* is that the more paths there are between two vertices, and the shorter these paths are, the stronger the connection. The *Katz measure* is defined as

$$Katz(u, v) = \sum_{l_{min}=1}^{l_{max}=\infty} \beta^l \left| path^l_{u,v} \right|, \tag{13}$$

where $|path^l_{u,v}|$ is the number of paths between u and v with a length of l. The problem with the *Katz measure* is that it has cubic complexity. This complexity makes it unfeasible for use in large social networks. Consequently, we did not use the *Katz measure* feature when building our classifier. Instead, we used the *Friends measure*, which can be regarded as an approximation of the *Katz measure*.

Friends measure. When looking at two vertices in a social network, we can assume that the more connections their neighborhoods have with each other, the higher the chances are that the two vertices are connected. We accept the logic of this statement and define the *Friends measure* as the number of connections between u and v neighborhoods. The formal definitions of *Friends measure* is: Let be $G = \langle V, E \rangle$ and $u, v \in V$

$$Friends\text{-}measure(u, v) = \sum_{x \in \Gamma(u)} \sum_{y \in \Gamma(v)} \delta(x, y), \tag{14}$$

where we define the function $\delta(x, y)$ as

$$\delta(x, y) = \begin{cases} 1 & \text{if } x = y \text{ or } (x, y) \in E \text{ or } (y, x) \in E \\ 0 & \text{otherwise} \end{cases}.$$

One can notice that in undirected networks, the *Friends measure* is a private case of the *Katz measure* where $\beta = 1$, $l_{min} = 2$, and $l_{max} = 3$.

Opposite direction friends. For a directed graph G, we can create a specific measure that indicates whether reciprocal connections exist between the vertices.

$$Opposite\text{-}direction\text{-}friends(u,v) = \begin{cases} 1 & \text{if } (v,u) \in E \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Same community. Let be $V = \coprod_{C' \in C} \coprod_{u \in C'} u$, where C is the set of all disjoint communities created from G by the Louvain method [Blondel et al. 2008]. We say that $u, v \in V$ are in the same community if $\exists C' \in C$ where $u, v \in C'$. The formal definition of *same-community* feature is

$$same\text{-}community(u,v) = \begin{cases} 1 & \text{if } \exists C' \in C \text{ where } u, v \in C'. \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

In case u and v were chosen at random, the Same-community feature can be used as an easy-to-compute rule of thumb for predicting existence of link between u and v[11].

4.1.3. Link Subgraph Features. Let be $u, v \in V$ using the neighborhoods definitions from Section 4.1.1, we can define the following subgraphs.

$$\begin{aligned} nh\text{-}subgraph(u,v) &= \{(x,y) \in E | x, y \in \Gamma(u) \cup \Gamma(v)\}, \\ nh\text{-}subgraph^{+}(u,v) &= \{(x,y) \in | x, y \in \Gamma^{+}(u) \cup \Gamma^{+}(v)\}. \end{aligned} \tag{17}$$

These subgraphs contain data about the number of connections between the links of u, v, including the inner connections between each vertex neighborhood. These types of graphs were also used to extract features for link prediction by Cukierski et al. [2011]. Another subgraph from which we can create features is the *inner-connection subgraph*.

$$\begin{aligned} inner\ subgraph(u,v) = \{&(x,y) \in E|, \\ &(x \in \Gamma(u) \text{ and } y \in \Gamma(v)) \text{ or} \\ &(x \in \Gamma(v) \text{ and } y \in \Gamma(u))\}. \end{aligned} \tag{18}$$

In social networks, the *inner subgraph* represents the number of connections between the friends of each user.

Link subgraphs links number. Using the previous definitions we can create features by counting the number of links in each subgraph.

$$\begin{aligned} &|nh\text{-}subgraph(u,v)| \\ &|nh\text{-}subgraph^{+}(u,v)| \\ &|inner\text{-}subgraph(u,v)| \end{aligned} \tag{19}$$

Link subgraphs components number. We can also count the number of strong and weak components for each subgraph.

$$\begin{aligned} &scc(nh\text{-}subgraph(u,v)),\ wcc(nh\text{-}subgraph(u,v)) \\ &scc(nh\text{-}subgraph^{+}(u,v)) \\ &scc(inner\text{-}subgraph(u,v)),\ wcc(inner\text{-}subgraph(u,v)) \end{aligned} \tag{20}$$

4.1.4. Path Features. Let be $u, v \in V$. We defined the following feature based on the length of the path between u and v.

[11]Due to the fact that the evaluated social networks in this study are large-scale social networks with hundreds of thousands of links, we precompute the community structure of each social network in order to optimize our algorithms' performances. However, in the case of predicting links in small communities, it is advised to avoid precomputing the *Same-community* feature due the risk of overfitting. In order to predict links inside small communities, we recommend using different subsets of features instead [Fire et al. 2012a, 2012b].

Shortest Path. We defined the *Shortest-path(u,v)* feature to be the shortest path length between u and v in G. If G is a directed graph, we can also define the feature *shortest path(v,u)*. The shortest-path feature has been explored in several papers and found to be one of the most significant features in link prediction [Hasan et al. 2006].

4.2. Training Set Size

Due to the large scale of the tested social network it is possible to train the link prediction classifiers on hundreds of thousands of positive and negative examples. Since the training set size directly affects the time required for building a classifier, we wanted to study how the training set size affects the performance of the classifiers. In addition, the ability to train an accurate classifier using a small dataset may enable building link prediction models in cases where only a small part of the social network is known.

In this study, in order to find the minimal balanced training size required, we first created two datasets ("easy" and "hard") each with 100,000 positive examples and 100,000 negative examples for every investigated social network. The "easy" dataset contained foreseen links where the negative examples were uniformly chosen at random from all unconnected pairs of vertices. In contrast to chosen false links in the "easy" dataset, in the "hard" dataset, the endpoints of the false links are two hops apart from each other (in the absence of the foreseen link). Previous studies have shown that in many cases, the existence of common friends is a very good predictor for the existence of a link between the two vertices [Liben-Nowell and Kleinberg 2007; Cukierski et al. 2011]. Such links represent a much more difficult classification problem, hence we refer to this dataset as a "hard dataset". Using the hard dataset enables simulation of difficult cases of missing link prediction scenarios between two vertices that may share many similar features. Moreover, using the hard dataset may assist in simulating link prediction inside small group and closed communities.

Subsequently, we created several training sets, each with a different size but the same number of positive and negative examples. The sizes of the training sets ranged from 2,000 to 200,000 examples, with at least 18 training sets of different sizes for each social network. During the evaluation process, we constructed and evaluated a chosen classifier on more than 360 different training sets of different scales. Moreover, the chosen classifier needed to be evaluated by using the tenfold cross-validation method, which is a very time-consuming evaluation method. To cope with these challenges, we chose to use the J48 decision tree algorithm to construct a classifier on each training set. We chose the J48 classifier due to its low run time and relatively high performance. Moreover, in our previous study [Fire et al. 2011], we demonstrated that using the J48 classifier for solving similar link prediction problems returns near-optimal results in many cases.

After constructing and evaluating the J48 classifier on each training set, for each social network, we presented the relations between different training set sizes and their received AUC results on a graph (see Figures 1 and 2). By observing these graphs, we noticed correlations between the different training set sizes and their matched AUC results in each social network. The observed correlations pattern resembled the behavior of the Gompertz function, therefore, we used Gompertz regression [Erickson et al. 2006; d'Onofrio 2005] in order to find the correlation between the size of the training set and the classifiers' AUC (see Figures 1 and 2). We created two Gompertz regression models for each social network: one for the "hard" datasets and one for the "easy" datasets. Gompertz regression is a nonlinear regression based on the Gompertz function $f(x) = ae^{be^{cx}}$. Due to its flexibility, the Gompertz function is frequently used for modeling a variety of processes, such as population expansion in a confined space [Erickson et al. 2006] and growth of tumors [d'Onofrio 2005]. We calculated the Gompertz regression for each one of the social networks (see Table II).

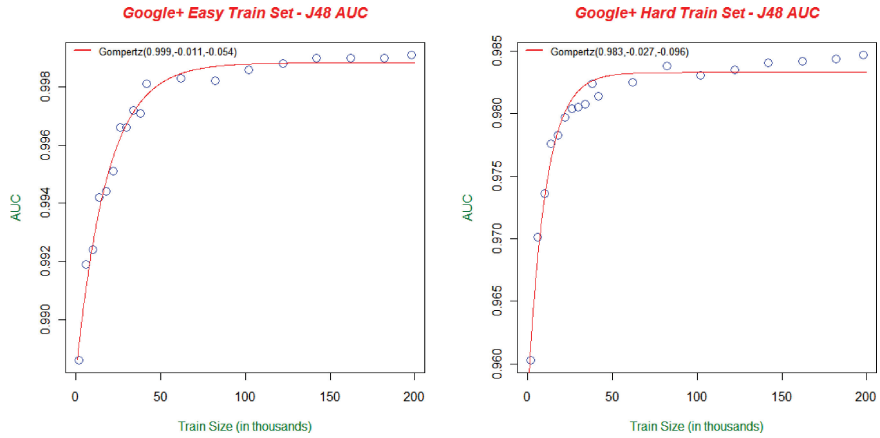

Fig. 1. Google+: Predicting decision tree classifier AUC using different training set sizes.

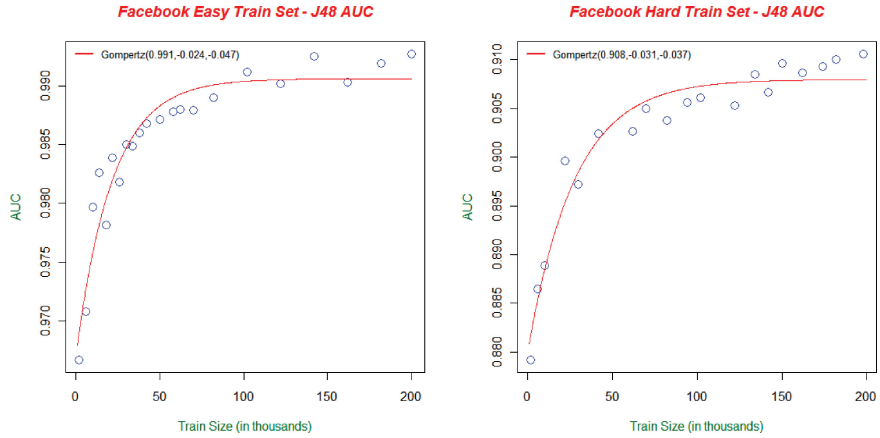

Fig. 2. Facebook: predicting decision tree classifier AUC using different training set sizes.

Table II. Gompertz Regressions Results and AUC Predictions

Network	"Easy" Gompertz	AUC (50K)	AUC ($x \to \infty$)	"Hard" Gompertz	AUC (50K)	AUC ($x \to \infty$)
Academia	$0.996e^{-0.019e^{-0.062x}}$	0.995	0.996	$0.976e^{-0.04e^{-0.177x}}$	0.976	0.976
DBLP	$e^{-0.009e^{-0.104x}}$	0.9999	1	$0.982e^{-0.048e^{-0.058x}}$	0.979	0.982
Facebook	$0.99e^{-0.024e^{-0.046x}}$	0.9876	0.99	$0.908e^{-0.031e^{-0.037x}}$	0.9036	0.908
Flickr	$0.998e^{-0.042e^{-0.812x}}$	0.998	0.998	$0.954e^{-0.026e^{-0.058x}}$	0.9526	0.954
Flixster	$0.998e^{-0.007e^{-0.044x}}$	0.998	0.997	$0.994e^{-0.012e^{-0.171x}}$	0.994	0.994
Google+	$0.999e^{-0.011e^{-0.054x}}$	0.998	0.999	$0.983e^{-0.027e^{-0.096x}}$	0.983	0.983
Gowalla	$0.997e^{-0.015e^{-0.042x}}$	0.995	0.997	$0.957e^{-0.036e^{-0.175x}}$	0.957	0.957
Twitter	$0.987e^{-0.018e^{-0.239x}}$	0.987	0.987	$0.977e^{-0.022e^{-0.084x}}$	0.9767	0.977
TheMarker	$0.975e^{-0.037e^{-0.188x}}$	0.975	0.975	$0.932e^{-0.035e^{-0.041x}}$	0.9278	0.932
YouTube	$0.999e^{-0.015e^{-0.204x}}$	0.999	0.999	$0.999e^{-0.012e^{-0.097x}}$	0.999	0.999

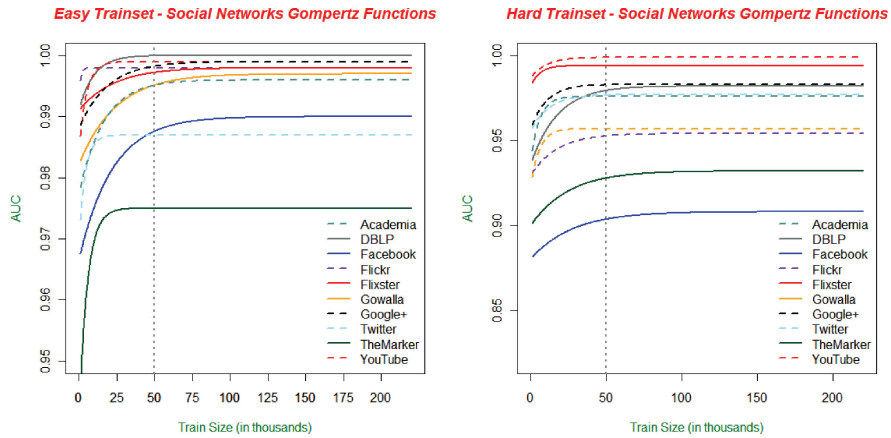

Fig. 3. Extrapolation of the link prediction classifiers for different social networks based on the Gompertz regression.

The results of the regression models are presented in Table II and in Figure 3. In all the regression models, the maximal residual standard error was 0.0038, indicating a good adequacy of our regression models.

According to the regression results, we discovered that on the one hand, many link prediction classifiers reach their near a maximal AUC with training sets that contain several thousands of examples only. On the other hand, in some social networks, in order to get near-maximal AUC, a training set with tens of thousands of examples is required. By using the Gompertz regression results, we can estimate the maximal AUC that can be obtained using a decision tree classifier. Moreover, we can see that in almost all the social networks, a near-maximal AUC can be obtained by using training sets with up to 50,000 examples (25,000 positive links and 25,000 negative links). Therefore, in the rest of this study, we constructed and evaluated our classifiers using training sets with 50,000 examples.

4.3. Experimental Setup

As part of our experiment we built a classifier for each social network graph G by randomly choosing 25,000 positive links that exist in the graph and 25,000 negative links that do not exist in the graph. Additionally, we randomly chose 25,000 negative links, where the shortest path between each link vertices was two. By using these randomly selected links, we created two datasets: the first dataset (i.e., the "easy" dataset) was built by taking 25,000 positive links and 25,000 negative links and the second dataset (i.e., the "hard" dataset) was built by taking 25,000 positive links and 25,000 negative links in the distance of two hops. These sets were formed in order to create datasets to train and evaluate our classifiers.

Next, in order to extract features, we developed a Python code using the Networkx graph package [Hagberg et al. 2008]. This code extracted the vector of features for each link (u, v) in the training sets. For each vertices u, v, we extracted all the vertex features (see Section 4.1.1) and all the link features for the link (u, v) (see Section 4.1.2). In total, we extracted 54 features for directed graphs and 34 features for undirected graphs. For each social network in our datasets, we created several feature subsets according to different characteristics of the features. Specifically, we created the following feature subsets.

—*All-features subset* contains all the extracted features: 54 features for directed networks and 34 for undirected networks.

—*Friends-features subset* contains the following features: vertices *degree* features *Common friends*; *Total friends Preferential attachment score Same community*, and *Friends measure*. A total of 9 features for undirected networks and 16 features for directed networks were created.
—*Friends measure and Same community (FM & SM)* contains the *Friends-measure* and the *Same-community* features.
—*Common-friends subset* contains only the *Common-friends* feature.
—*Friends-measure subset* contains only the *Friends-measure* feature.
—*Jaccard's coefficient* contains only the *Jaccard's coefficient* feature.
—*Same community* contains only the *same-community* feature.

Based on the results of our previous experiments [Fire et al. 2011], we used the J48 decision trees classifier and two ensemble learning methods, namely, Bagging and Random Forest. Ensemble learning, also known as "committee machines" or a mixture of experts, is a well-known technique in machine learning in which the outputs of several classifiers (experts) are combined. Each of the classifiers solves the same original task. Combining these classifiers usually results in a better composite global model, with more accurate and reliable estimates or decisions than can be obtained from using a single model. This idea imitates a common human characteristic: the desire to obtain several opinions before making any crucial decision. It is known that combining different types of classifiers can improve predictive performance, mainly due to the phenomenon that various types of classifiers have different "inductive biases". In particular, it has been shown that combining diverse classifiers can be used to reduce the variance error (i.e., error due to sampling variation) without increasing the bias error (i.e., error due to an inadequate model). Additionally, many participants in prediction contests combine various models in order to achieve the best results (see, for example, Koren [2009]).

As has been shown in our previous work [Fire et al. 2011], the ensemble schemas achieve the highest predictive performance; however, their running times are very long. Among the single models, the J48 decision tree and the Artificial Neural Networks (ANNs) also achieved a relatively high performance. However, ANN models are difficult to interpret and have longer construction times than J48. Consequently, we found that the J48 provides the best trade-off between computational costs and accuracy since its AUC results are, in most cases, slightly lower than those of the ensemble methods. Conversely, its computational times are much faster. Another advantage of the J48 is that its models are easy to understand and analyze. Obviously, when the predictive performance is of the highest importance, more time-consuming and sophisticated methods can be applied. The performance observed for the four evaluated ensemble methods, Bagging, Adaboost, Rotation Forest, and Random Forest, was very similar for all these methods in most cases. Therefore, our selection of the ensemble algorithms was mainly motivated by their time performance differences, and the two quickest, namely Bagging and Random Forest, were chosen. We used WEKA's [Hall et al. 2009] J48 (a variant of the well-known C4.5), Bagging, and Random Forest implementations of the corresponding algorithms. For each of these algorithms, most of the configurable parameters were set to their default values except for the following. For the J48 classifier, the minimum number of instances per leaf parameter was set to 10. The number of iterations for all ensemble methods was set to 100. The Bagging algorithm was evaluated using J48 as the base classifier with the number of instances per leaf set to 10.

5. RESULTS

For each social network and subset of features we evaluated the specified machine learning algorithms (see Section 4.3) using a 10-fold cross-validation approach. For

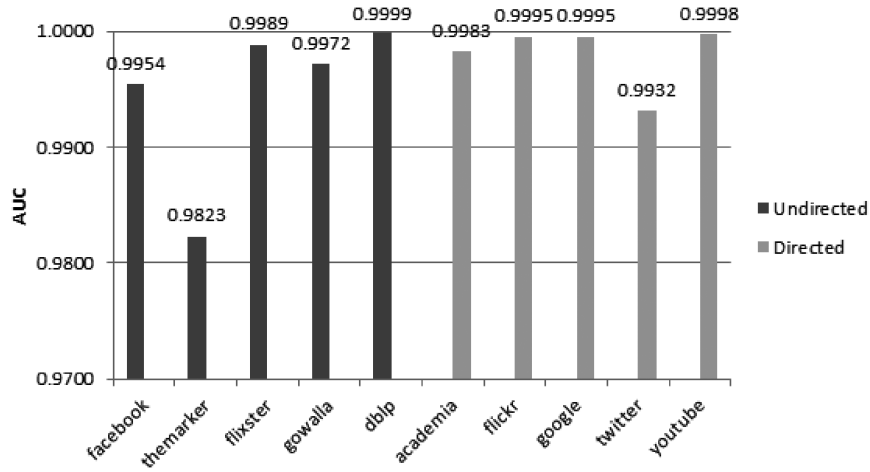

Fig. 4. AUC results: Bagging algorithm using all the features on easy datasets.

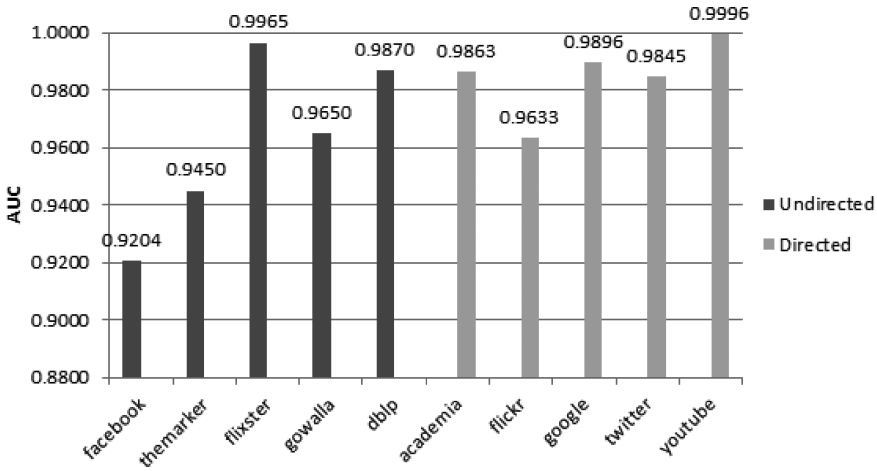

Fig. 5. AUC results: Bagging algorithm using all the features on hard datasets.

example, the AUC results of the Bagging algorithm on all the networks using a set of all features on easy and hard datasets are presented in Figures 4 and 5.

It can be seen that on the easy datasets, the Bagging ensemble achieves very high predictive performance. On most of the evaluated networks the AUC values approximate to 1. The more difficult social networks for link predictions are TheMarker Cafe among the undirected and Twitter among the directed networks. The reason for the relatively low (yet the AUC is above 98 and 99% correspondingly) prediction performance on these networks is due to the nature of these social networks. TheMarker Cafe social network contains many random connections between its users, while the Elite subgraph of Twitter is very dense [Avin et al. 2011], making link prediction a more difficult task.

However, on the hard datasets, the picture is somewhat different. The AUC results are lower than on the easy datasets, especially for undirected networks. This is explained by the short distance of two between the negative links. The more difficult network, in this case, is Facebook, with an AUC of 0.923. We expect that the predictive

Table III. Easy and Hard Datasets - Random Forest AUC Results

Network	Training	All-features	Friends-features	FM&SM	Friends-Measure	Common-Friends	Jaccard's-coefficient	Same-community
Academia	Easy	0.998	0.996	0.978	0.962	0.865	0.864	0.834
	Hard	0.985	0.981	0.85	0.849	0.841	0.807	0.563
DBLP	Easy	0.9998	0.999	0.994	0.973	0.971	0.971	0.89
	Hard	0.989	0.989	0.812	0.811	0.874	0.801	0.568
Facebook	Easy	0.993	0.989	0.974	0.964	0.946	0.943	0.827
	Hard	0.923	0.915	0.803	0.817	0.88	0.812	0.58
Flickr	Easy	0.998	0.999	0.967	0.921	0.832	0.831	0.82
	Hard	0.96	0.95	0.803	0.804	0.864	0.851	0.59
Flixster	Easy	0.998	0.998	0.969	0.865	0.711	0.711	0.81
	Hard	0.996	0.995	0.924	0.924	0.922	0.961	0.537
Google+	Easy	0.9996	0.995	0.99	0.9586	0.899	0.89	0.944
	Hard	0.989	0.98	0.89	0.888	0.882	0.898	0.5355
Gowalla	Easy	0.996	0.995	0.976	0.922	0.885	0.883	0.87
	Hard	0.966	0.968	0.837	0.837	0.874	0.817	0.545
TheMarker	Easy	0.983	0.978	0.958	0.956	0.941	0.922	0.65
	Hard	0.945	0.937	0.831	0.838	0.86	0.759	0.534
Twitter	Easy	0.987	0.9917	0.938	0.915	0.967	0.941	0.695
	Hard	0.983	0.981	0.813	0.798	0.88	0.813	0.598
YouTube	Easy	0.999	0.9996	0.988	0.925	0.761	0.761	0.868
	Hard	0.999	0.999	0.942	0.941	0.887	0.9	0.569

performance on these social networks as depicted by our evaluation is lower than would be in practice. The reason for that is a relatively high number of missing links in the evaluation dataset itself. There might be social links that are correctly classified by machine learning classifiers as existing links, but due to the evaluation procedure are incorrectly treated as false positive prediction. We assume that this phenomenon of relatively high number of missing links in the Facebook social networks could be the result of user privacy settings which limit the access of our crawlers to the users' existing connections in these networks. This limitation causes the collected dataset to be more biased than the other crawled social networks. Facebook provides its users not only the option of hiding their friends list, but also the option to limit receiving friend requests from friends-of-friends[12] only. This type of privacy setting not only limits our crawler's access, but can also change the social network topology to be different from other more public social networks. We assume that these differences cause our classifiers' performance to decline. The decline in the AUC for Facebook is less stressed in the easy dataset because there is only a small chance for a link between two profiles to have no common friends. The chance of such a link to be hidden by privacy settings or not picked up by a crawler is even smaller. Table III presents the AUC results of the Random Forest method for each social network on all evaluated feature subsets. The results of the J48, Random forest, and Bagging algorithms on the Google+ network using all evaluated feature subsets are presented in Figure 6.

It can be seen that the best performance is achieved by using the set of all features. However, training classifiers on a large number of features is typically very time consuming and computationally intensive. The classifiers' performance using the subset of the friends features only is only slightly lower in most cases, and sometimes is the same or even higher than using all the features. On the other hand, the computational times required for learning and classification using a small number of features only are

[12]https://www.facebook.com/help/privacy.

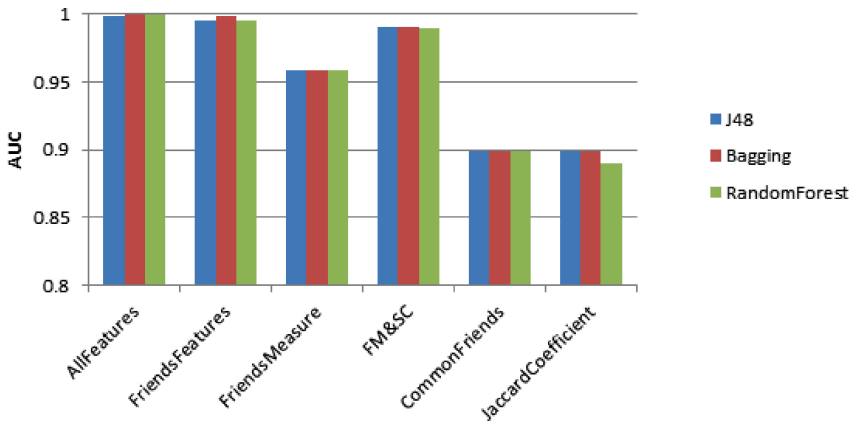

Fig. 6. AUC results: J48, Bagging, and Random Forest algorithms on Google+ network using various subsets of features (easy dataset).

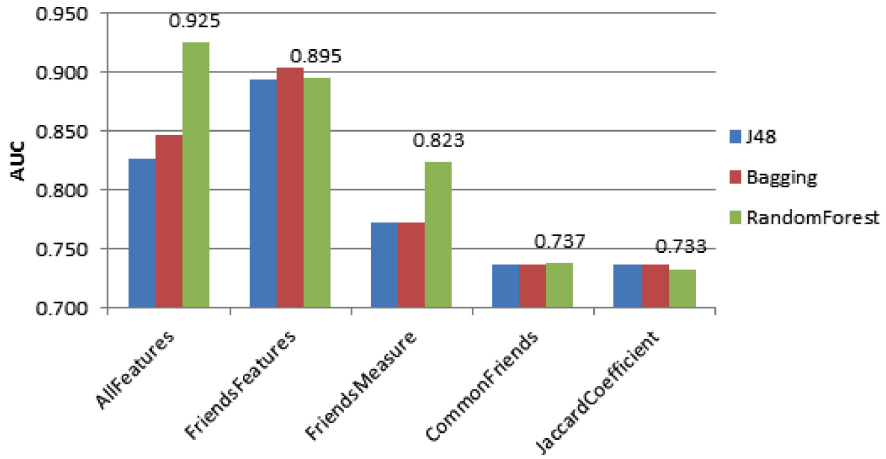

Fig. 7. AUC results: J48, Bagging, and Random Forest algorithms on IJCNN Flickr social network using various subsets of features.

much faster. Additionally, we can see that the new *Friends measure* provides a better predictive performance than the well-known *Common-friends* and *Jaccard's-coefficient* features. It can also be seen that another new feature, *Same community*, improves the results of the Friends measure on the easy datasets (see Table III).

We used the Friedman test proposed by Demsar [2006] for validating the statistical significance of the differences between the evaluated sets of features. For this test we focused on the results of the Random Forest method. With a confidence level of above 0.99, the results reveal that there are significant differences between the evaluated sets of features. We then proceeded with the Nemenyi post hoc test [Demsar 2006] to compare the feature sets to each other. Interestingly, the following groups of features subsets were found not to be significantly different: (A) *All features* (average rank 1.1) versus *Friends features* (average rank 1.9), (B) *Common friends* (average rank 4.5) versus. *Friends measure* (average rank 4.5) versus *Jaccard's coefficient* (average rank 5.5) and versus *Friends measure and Same community* (average rank 3.9), and (C) *Jaccard's coefficient* (average rank 5.5) versus *Same community* (average rank 6.7).

We also tested a version of our classifiers[13] against the IJCNN social network challenge test set. The IJCNN test set was created using the different random edge chooser algorithm. The IJCNN random algorithm was written in order to make link prediction more difficult and can be somewhat biased. Using the IJCNN random algorithm we created a training set of 25,000 positive edges and 25,000 negative edges, and built a classifier using the features presented in Section 4.1. The AUC results of our classifier using the J48, Bagging, and Random Forest algorithms on all feature subsets are presented in Figure 7.

It can be seen that on this test data the Random Forest method achieves the highest AUC result using all the features. The results of other methods and of using various features subsets are much lower. Interestingly, the Bagging and J48 algorithms demonstrate better performance using only the friends-features subset than on all the features. This can be caused by the tendency of J48 method to overfit the training data in the presence of a large number of features. However, the Random Forest method, which constructs an ensemble of the J48 decision trees, randomly selects just a subset of features for each tree. Thus, it does not suffer from the aforementioned problem and achieves the highest AUC performance. A variant of our classifier that used most of the features in the All-features subset and the Rotation Forest method participated in the IJCNN social network challenge and received a result of 0.9244 in AUC on the challenge test set The average time of extracting features for each edge in the IJCNN test set was 0.64[14] seconds, as opposed to the 10 seconds per edge that was obtained by Cukierski et al. [2011].

To obtain an indication of the usefulness of the various features, we analyzed their importance using Weka's information gain attribute selection algorithm. The top attributes with the highest rank for each one of the training sets for all social networks are presented in Tables IV and V.

It can be seen that different attributes are the most influential for different social networks and many features turn out to be useless for predicting links in the hard dataset. However, the Friends measure is among the most influential features on almost all of the networks. In addition, the average information gain of this feature is the highest among all the evaluated features. Furthermore, it can be concluded by Table V that features which only exist in directed social networks, such as the *Opposite-direction-friends* feature and *WCC* features, are very useful for predicting directed social networks links.

5.1. Ranking Performance Evaluation

We used the Average Precision (AP) measure in order to verify the ranking performance of our algorithm. The AP is a frequently used measure in the information retrieval and machine learning domains. The AP is formulated as

$$\frac{1}{N}\sum_{k=1}^{N} P(k),$$

where $P(k)$ is the Precision at top k (*precision @ k*) which measures how many of top-k positive predictions of an algorithm are actually correct, and N is the total number of considered cut-off points. Usually, *precision @ k* is calculated per social network user in order to evaluate how well the link recommender algorithm performs in recommending

[13]The IJCNN link prediction classifier was constructed by using all features except the *Same-community* feature.

[14]We ran our algorithm using Python 2.7 on a regular Dell Latitude E6420 laptop with i7 core, and 8GB RAM.

Table IV. InfoGain Values of Different Features for Directed and Undirected Networks

			Friends-measure(u,v)	Jaccard's-coefficient(u,v)	|Inner-subgraph(u,v)|	Preferential-attachment-score(u,v)	Common-Friends_bi(u,v)	|nh-subgraph+(u)|	avg-scc+(u)	Total-Friends(u,v)	|nh-subgraph(u)|	|nh-subgraph+(u, v)|	Density-nh-subgraph+(u)	|nh-subgraph(u, v)|	avg-scc(u)	scc(nh-subgraph+(u,v))	shortest-path(u,v)	scc(nh-subgraph(u,v))	scc(Inner-subgraph(u,v))	same-community(u,v)	|nh-subgraph+(v)|	|nh-subgraph(v)|	avg-scc+(v)	Density-nh-subgraph+(v)	avg-scc(v)	is-shortest-path(u,v)
Undirected	Facebook	easy	0.7	0.7	0.7	0.4	0.7	0.2	0.3	0.3	0.2	0.2	0.2	0.2	0.2	0.5	0.7	0.0	0.0	0.4	0.2	0.2	0.3	0.2	0.2	0.0
		hard	0.3	0.3	0.2	0.2	0.4	0.2	0.2	0.1	0.2	0.1	0.2	0.1	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	TheMarker	easy	0.7	0.5	0.7	0.7	0.6	0.4	0.4	0.6	0.4	0.6	0.4	0.6	0.4	0.1	0.5	0.2	0.2	0.1	0.4	0.4	0.4	0.4	0.4	0.0
		hard	0.4	0.1	0.4	0.4	0.3	0.4	0.4	0.2	0.4	0.2	0.4	0.2	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Flixter	easy	0.5	0.3	0.5	0.9	0.3	0.4	0.4	0.8	0.4	0.8	0.3	0.6	0.3	0.5	0.6	0.8	0.8	0.3	0.4	0.3	0.4	0.3	0.3	0.1
		hard	0.5	0.6	0.2	0.6	0.6	0.4	0.4	0.5	0.4	0.4	0.3	0.3	0.4	0.1	0.3	0.5	0.4	0.0	0.1	0.1	0.1	0.0	0.1	0.1
	Gowalla	easy	0.6	0.5	0.5	0.5	0.6	0.3	0.3	0.3	0.3	0.3	0.2	0.3	0.2	0.4	0.6	0.2	0.2	0.5	0.3	0.3	0.3	0.2	0.2	0.0
		hard	0.3	0.3	0.2	0.2	0.4	0.3	0.3	0.1	0.3	0.1	0.3	0.1	0.2	0.1	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	DBLP	easy	0.8	0.8	0.7	0.3	0.8	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.8	0.6	0.3	0.2	0.6	0.2	0.2	0.2	0.2	0.2	0.1
		hard	0.2	0.2	0.2	0.1	0.4	0.2	0.2	0.0	0.2	0.1	0.1	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Directed	Academia	easy	0.7	0.5	0.6	0.4	0.2	0.3	0.2	0.3	0.2	0.3	0.2	0.3	0.2	0.2	0.4	0.2	0.2	0.4	0.2	0.2	0.1	0.1	0.1	0.0
		hard	0.3	0.3	0.1	0.1	0.1	0.2	0.2	0.1	0.2	0.1	0.1	0.1	0.2	0.1	0.2	0.1	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Flickr	easy	0.6	0.4	0.7	0.9	0.0	0.8	0.8	0.8	0.7	0.7	0.6	0.7	0.7	0.8	0.0	0.9	0.8	0.3	0.3	0.3	0.3	0.3	0.3	0.7
		hard	0.3	0.3	0.2	0.2	0.0	0.3	0.1	0.3	0.3	0.2	0.2	0.2	0.2	0.3	0.2	0.3	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	YouTube	easy	0.7	0.3	0.5	0.7	0.3	0.5	0.5	0.6	0.4	0.5	0.4	0.5	0.4	0.4	0.2	0.5	0.5	0.5	0.3	0.3	0.3	0.3	0.3	0.1
		hard	0.6	0.5	0.2	0.3	0.2	0.3	0.3	0.2	0.3	0.2	0.3	0.1	0.3	0.2	0.2	0.2	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.1
	Google+	easy	0.8	0.6	0.6	0.5	0.3	0.4	0.3	0.4	0.3	0.4	0.3	0.3	0.3	0.3	0.2	0.4	0.3	0.7	0.4	0.3	0.3	0.3	0.3	0.2
		hard	0.5	0.5	0.3	0.3	0.2	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.2	0.1	0.2	0.2	0.0	0.2	0.2	0.2	0.2	0.2	0.1
	Twitter	easy	0.6	0.7	0.6	0.6	0.6	0.3	0.3	0.3	0.3	0.4	0.3	0.4	0.3	0.0	0.2	0.0	0.3	0.1	0.3	0.3	0.3	0.3	0.3	0.0
		hard	0.4	0.4	0.4	0.3	0.4	0.3	0.2	0.2	0.3	0.2	0.3	0.2	0.2	0.0	0.1	0.0	0.1	0.0	0.1	0.1	0.1	0.2	0.1	0.0
Average			**0.52**	0.44	0.43	0.43	0.37	0.34	0.32	0.32	0.32	0.32	0.29	0.29	0.28	0.26	0.26	0.24	0.23	0.20	0.18	0.17	0.17	0.16	0.15	0.09
StDev			0.19	0.19	0.21	0.23	0.23	0.14	0.16	0.22	0.12	0.21	0.12	0.19	0.13	0.26	0.23	0.24	0.23	0.23	0.15	0.14	0.14	0.13	0.12	0.15

Table V. InfoGain Values of Different Features for Directed Networks

			Common-Friends(u,v)	Transitive-Friends(u,v)	dout(u)	d(u)	avg-wcc+(u)	dbi(u)	bi-degree-density(u)	out-degree-density(u)	avg-wcc(u)	in-degree-density(u)	shortest-path(v,u)	din(u)	transitive-friends(v,u)	wcc(inner-subgraph(u, v))	wcc(nh-subgraph(u,v))	din(v)	Opposite-direction-friends(u,v)	avg-wcc+(v)	d(v)	avg-wcc(v)	dbi(v)	dout(v)	bi-degree-density(v)	out-degree-density(v)	in-degree-density(v)	is-shortest-path(v,u)
Directed	Academia	easy	0.5	0.4	0.3	0.3	0.3	0.3	0.2	0.3	0.2	0.2	0.4	0.2	0.3	0.1	0.1	0.2	0.3	0.2	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.0
		hard	0.3	0.4	0.3	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.1	0.0	0.0	0.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Flickr	easy	0.4	0.4	0.9	0.8	0.8	0.9	0.9	0.9	0.7	0.9	0.0	0.4	0.0	0.7	0.7	0.3	0.0	0.3	0.3	0.3	0.0	0.0	0.0	0.0	0.0	0.0
		hard	0.4	0.4	0.3	0.3	0.3	0.3	0.1	0.1	0.2	0.1	0.0	0.2	0.0	0.2	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	YouTube	easy	0.3	0.3	0.5	0.5	0.5	0.5	0.5	0.5	0.4	0.4	0.3	0.5	0.3	0.5	0.5	0.3	0.6	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.2	0.2
		hard	0.5	0.5	0.3	0.3	0.3	0.3	0.2	0.2	0.3	0.3	0.5	0.3	0.2	0.1	0.1	0.0	0.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Google+	easy	0.6	0.6	0.4	0.4	0.4	0.3	0.3	0.3	0.3	0.3	0.1	0.3	0.3	0.1	0.3	0.4	0.3	0.4	0.4	0.3	0.2	0.3	0.2	0.3	0.3	0.1
		hard	0.5	0.6	0.3	0.3	0.3	0.3	0.2	0.2	0.3	0.2	0.2	0.2	0.2	0.1	0.1	0.2	0.3	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.1	0.0
	Twitter	easy	0.7	0.7	0.4	0.3	0.3	0.3	0.3	0.3	0.3	0.2	0.6	0.3	0.6	0.4	0.0	0.3	0.7	0.3	0.3	0.3	0.3	0.3	0.3	0.2	0.2	0.0
		hard	0.4	0.4	0.3	0.3	0.3	0.3	0.2	0.2	0.3	0.2	0.7	0.3	0.4	0.2	0.0	0.2	0.7	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.0
Average			0.47	0.47	0.41	0.39	0.39	0.36	0.32	0.32	0.31	0.30	0.30	0.29	0.24	0.24	0.21	0.20	0.19	0.19	0.19	0.17	0.13	0.13	0.12	0.12	0.10	0.04
StDev			0.12	0.11	0.20	0.18	0.18	0.20	0.21	0.23	0.15	0.23	0.24	0.10	0.18	0.21	0.24	0.15	0.26	0.14	0.14	0.12	0.12	0.12	0.11	0.11	0.10	0.06

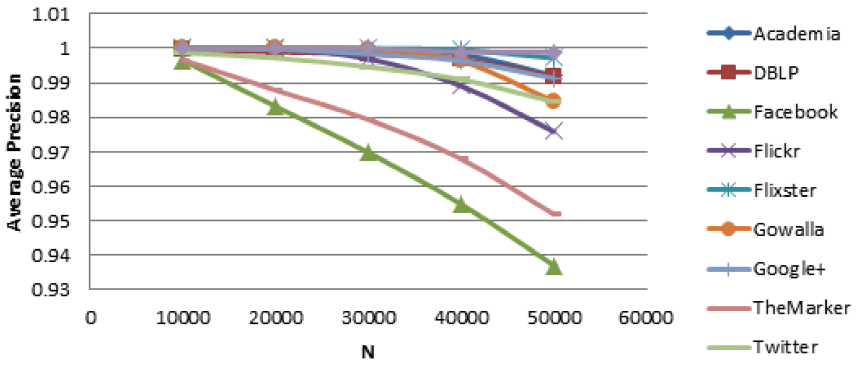

Fig. 8. Bagging over all features AP results for different N values on balanced hard datasets.

the user new links. However, in our study we wanted to optimize the recovery of missing links for each social network. Therefore, our *precision @ k* estimated the precision in recovery of missing links from each social network point of view instead of the user's point of view. This measure is particularly appropriate in the context of missing link prediction where we aim to discover the unknown existing links in the network. Thus, for our AP calculation, we order all the link predictions in descending order of their probability to exist in the network and measure how many of top-k links actually exist in the network. The AP is then calculated as an average of the *precision @ k* values for all k's from 1 to N.

The experiment was performed on balanced test sets consisting of 50,000 positive and 50,000 negative examples. Figure 8 presents the the APs for various N values using the Bagging algorithm over all features in the hard datasets. It can be seen that the higher the N the lower is the AP. This means that, as expected, there are more false positive predictions for lower probability cut-off points. However, even for the highest evaluated $N = 50{,}000$ (equal to the total number of positive examples in the test set), the AP is very high for all datasets, and above 97% for most of them.

5.2. The Effect of Imbalanced Data on Predictive Performance

In many link prediction scenarios, there are many more negative links than positive links. This phenomenon is referred to in the literature as the *class imbalance problem* [Chawla et al. 2004]. In particular, class imbalance usually occurs when, in a classification problem, there are many more examples of a certain class than of another class. Imbalanced datasets pose difficulties for induction algorithms. In particular, standard machine learning techniques may be "overwhelmed" by the majority class and ignore the minority class. A well-known method in machine learning for overcoming the class imbalance problem is undersampling the majority class (in our case, the negative links), as described in Chawla et al. [2004]. After training the classifier from a balanced dataset, the classifier can still be used for addressing imbalanced datasets. For this purpose, one should use the positive class probability which is provided by the classifier to rank the candidate links in descending order. Moreover, certain link prediction challenges [Cukierski et al. 2011; Narayanan et al. 2011] and works [Guha et al. 2004; Hasan et al. 2006; Leskovec et al. 2010] used balanced training and test sets for evaluating the performance of machine learning algorithms despite the fact that the distribution of the resulting testing data does not represent the same challenges as the real-world distribution. While in the evolution scenario frequently used in previous link prediction works the imbalanced assumption holds, in the scenario that motivates this work, the imbalanced setting is partially relaxed due to the fact that we aim to

Table VI. AUC for Various Imbalance Levels (Hard and Easy Dataset, All Features, Bagging)

Network	Training set	0.1%	0.2%	1%	2%	5%	10%	50%
Academia	Easy	0.992	0.989	0.989	0.989	0.989	0.989	0.989
	Hard	0.983	0.986	0.988	0.987	0.987	0.986	0.987
DBLP	Easy	0.999	0.999	0.999	0.999	0.999	0.999	1.000
	Hard	0.984	0.986	0.985	0.986	0.987	0.987	0.987
Facebook	Easy	0.996	0.995	0.995	0.995	0.996	0.995	0.996
	Hard	0.913	0.918	0.919	0.916	0.919	0.918	0.918
Flickr	Easy	0.997	0.997	0.997	0.997	0.997	0.997	0.997
	Hard	0.974	0.96	0.964	0.96	0.962	0.962	0.963
Flixster	Easy	0.988	0.988	0.990	0.989	0.989	0.988	0.988
	Hard	0.998	0.996	0.996	0.996	0.996	0.996	0.996
Google+	Easy	0.999	0.999	0.999	0.999	0.999	0.999	1
	Hard	0.992	0.992	0.99	0.99	0.989	0.989	0.99
Gowalla	Easy	0.99	0.998	0.997	0.998	0.998	0.998	0.998
	Hard	0.963	0.962	0.969	0.968	0.965	0.965	0.965
TheMarker	Easy	0.990	0.983	0.984	0.983	0.983	0.983	0.983
	Hard	0.931	0.935	0.933	0.938	0.941	0.94	0.937
Twitter	Easy	0.984	0.990	0.990	0.991	0.990	0.990	0.990
	Hard	0.978	0.988	0.987	0.988	0.987	0.987	0.985
YouTube	Easy	0.999	0.999	0.999	0.999	0.999	0.999	0.999
	Hard	0.999	0.999	0.999	0.999	0.999	0.999	0.999

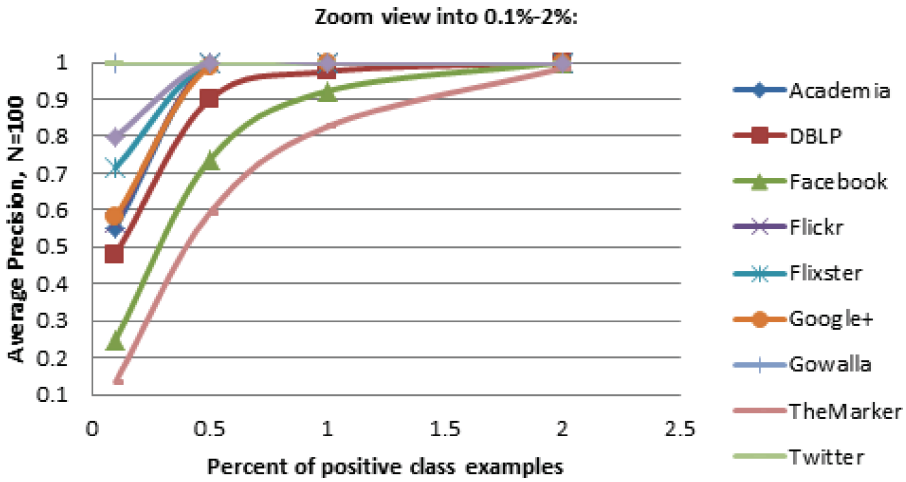

Fig. 9. Bagging average precision results for different imbalance levels (using all features on hard datasets).

solve the problem of uncovering hidden or missing links. In previous work [Fire et al. 2012b], we demonstrated that similar methods of using balanced training sets can help predict hidden links in high fractional datasets. Obviously there are other methods for overcoming the imbalance challenge and these methods have been evaluated with previous link prediction methods [Lichtenwalter et al. 2010].

Previous works show that the imbalance level of the training set affects the AUC performance of the test set (see Mazurowski et al. [2008]; Lichtenwalter et al. [2010]). However, while a balanced training set can always be generated by undersampling the overrepresented class, as described in Section 4.2, we do not have control of the imbalance level of the test set and in some cases we cannot even estimate the imbalance

level. In this work, we examined a different scenario in which a balanced training set is generated and used to train a classifier which is then tested on an imbalanced test set. For these experiments, two nonoverlapping subsets of data were selected from the original datasets. The first subset of data consisting of 25,000 positive and 25,000 negative examples was used as balanced training set. The second subset consisted of 50,000 positive and 50,000 negative examples and was used to generate test sets for experiments with various data imbalance levels. For each test set all the negative examples (i.e., 50,000) and the number of positive examples corresponded to the specific imbalance level selected. The evaluated imbalance levels and the reason for their selection are described next in this section.

Table VI presents the AUC performance for different imbalance levels using the Bagging algorithm over all features in the hard and easy datasets. The selection of the examined imbalance levels is motivated by the work of Liben-Nowell and Kleinberg [2007] showing that the imbalance level on several real link prediction problem datasets varies from 0.1% to almost 0.5%; and by the work of Mazurowski et al. [2008], who evaluated the effects of 1%, 2%, 5%, 10%, 20%, and 50% imbalance levels. Similar to Mazurowski et al. [2008], we define the positive class prevalence index c as $c = \frac{N_{pos}}{N_{tot}}$, where N_{pos} is the number of positive examples and N_{tot} is the total number of examples in the test set. For the experiments the number of negative examples was kept constant and equal to 50,000. The number of positive examples varies correspondingly for each prevalence index c. For the case of 0.1% imbalance level ($c = 0.001$) the number of positive examples in the test set is 50; and for 0.5% imbalance level the number of positive examples in the test set is 250, and so on. In total, we evaluated the following seven imbalance levels: 0.1%, 0.5%, 1%, 2%, 5%, 10%, and 50% where the last one corresponds to the equal prevalence of both classes. As mentioned before, the AUC results for all these imbalance levels are presented on Table VI. It can be seen that the predictive performance in terms of AUC remains stable over the different imbalance levels varying from 0.1% to 50%.

The results in Table VI indicate that if a balanced training set is used, the imbalance level of the test set does not affect the AUC performance measured on the test set. This conclusion corresponds with previous results showing that "AUC, unlike error rate, is unaffected by the class distribution of the test set" [Weiss and Provost 2003]. It should be noted that we examined only balanced training sets. We selected this balanced strategy because Weiss and Provost [2003] show that in most cases "the balanced distribution is within the optimal range" when using AUC as the performance measure. Nevertheless, in some cases we might benefit from using a slightly imbalanced training set. We leave this for future research.

Unlike AUC, the imbalance level of the test set does affect the values of other performance measures like the AP. The results for this measure of the Bagging algorithm on test sets with various imbalance levels and $N = 100$ are presented in Figure 9. As expected, the performance of the AP drops as the imbalance level increases. Still, near-optimal predictive performance is observed for test sets with a moderate low percentage (1% and above) of positive class examples.

6. CONCLUSION

This article presents methods for constructing efficient and effective classifiers based on a set of features that are easy and fast to calculate. We achieved this by defining a set of computationally efficient features and extracting them from ten real social network datasets. We created several feature subsets according to their characteristics and evaluated the classifier performance for each one of these subsets with several machine learning algorithms. The evaluation demonstrated that our models performed

well in terms of AUC measures (also referred to as ROC area) for all the tested datasets (see Table III and Section 5). The best results were obtained using all the features. We furthermore demonstrated that it would be sufficient to obtain good results with a relatively high AUC, even with a smaller subset of the features, such as the *Friends subset*, which contains 9 features only.

Another contribution of this article is the introduction of a topological feature, the *Friends measure*, which is simple to calculate. The experimental results demonstrated that using the *Friends measure* for link prediction produces better results when compared to the use of the well-known *Common-friends* and *Jacquard's-coefficient* features. Futhermore, using attribute information gain analysis, it was found that the Friends measure is among the most influential features in all of the evaluated networks and has the highest average information gain (see Table IV). We demonstrated as well that our models provide good results even when tested on links with end vertices that are two hops away from each other. Such results demonstrate the ability to predict link creation within tightly coupled social communities and shows that the obtained classifiers can distinguish between friends and nonfriends, even if the two vertices have at least one common friend (i.e., they are two hops from each other).

Our research currently considers link prediction using graph topology features only. A possible future research direction is to analyze other types of social network features and to examine their impact on link prediction. Examples of other types of features are: content-based features such as posted messages, demographic features (e.g., gender, age, etc.), and affiliation-related features. Additionally, in future work, we plan to evaluate the presented link prediction methods on different link prediction scenarios, such as social network evolution and a friends recommender system where friend suggestions are based on measuring precision@k for a given user. Another future direction would be to examine our algorithms in relation to different domains, such as bioinformatics networks.

7. AVAILABILITY

Anonymous version of Academia.edu, Google+, and TheMarker Cafe social networks topologies are available for other researchers to use on our research group Web site http://proj.ise.bgu.ac.il/ sns/.

ACKNOWLEDGMENTS

We would want to thank Jennifer Brill for repeated readings and markups on our grammar and spelling. We also want to thank the anonymous reviewers for their valuable comments and suggestions to improve the manuscript.

REFERENCES

Airoldi, E., Blei, D., Fienberg, S., Xing, E., and Jaakkola, T. 2006. Mixed membership stochastic block models for relational data with application to protein-protein interactions. In *Proceedings of the International Biometrics Society Annual Meeting.*

Avin, C., Lotker, Z., and Pignolet, Y. 2011. On the elite of social networks. Arxiv preprint. http://arxiv.org/abs/1111.3374.

Barabasi, A.-L. and Albert, R. 1999. Emergence of scaling in random networks. *Sci. 286*, 509–512.

Blondel, V., Guillaume, J., Lambiotte, R., and Lefebvre, E. 2008. Fast unfolding of communities in large networks. *J. Statist. Mech. Theory Exper. 2008*, P10008.

Cha, M., Haddadi, H., Benevenuto, F., and Gummadi, K. P. 2010. Measuring user influence in twitter: The million follower fallacy. In *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media (ICWSM'10).*

Chawla, N., Japkowicz, N., and Kotcz, A. 2004. Editorial: Special issue on learning from imbalanced data sets. *ACM SIGKDD Explor. Newslett. 6*, 1, 1–6.

CHO, E., MYERS, S., AND LESKOVEC, J. 2011. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Press, New York, 1082–1090.

CUKIERSKI, W. J., HAMNER, B., AND YANG, B. 2011. Graph-based features for supervised link prediction. In *Proceedings of the International Joint Conference on Neural Networks*.

DEMSAR, J. 2006. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res. 7*, 1–30.

D'ONOFRIO, A. 2005. A general framework for modeling tumor-immune system competition and immunotherapy: Mathematical analysis and biomedical inferences. *Phys. D: Nonlinear Phenomena 208*, 3–4, 220–235.

DOPPA, J. R., YU, J., TADEPALLI, P., AND GETOOR, L. 2009. Chance-constrained programs for link prediction. In *Proceedings of the Workshop on Analyzing Networks and Learning with Graphs at NIPS Conference*.

ERICKSON, G., CURRIE, P., INOUYE, B., AND WINN, A. 2006. Tyrannosaur life tables: An example of nonavian dinosaur population biology. *Sci. 313*, 5784, 213.

FACEBOOK-NEWSROOM. 2013. http://www.facebook.com.

FIRE, M., KATZ, G., ROKACH, L., AND ELOVICI, Y. 2012a. Links reconstruction attack: Using link prediction algorithms to compromise social networks privacy. In *Security and Privacy in Social Networks*, Springer, 181–196. http://proj.ise.bgu.ac.il/sns/papers%5Clink_reconstruction.pdf.

FIRE, M., PUZIS, R., AND ELOVICI, Y. 2012b. Link prediction in highly fractional data sets. In *Handbook of Computational Approaches to Counterterrorism*, Springer, 283–300.

FIRE, M., TENENBOIM, L., LESSER, O., PUZIS, R., ROKACH, L., AND ELOVICI, Y. 2011. Link prediction in social networks using computationally efficient topological features. In *Proceedings of the 3rd IEEE International Conference on Privacy, Security, Risk and Trust (PASSAT'11) and the 3rd IEEE International Conference on Social Computing (SocialCom'11)*. 73–80.

GIBBONS, A. 1985. *Algorithmic Graph Theory*. Cambridge University Press.

GIRVAN, M. AND NEWMAN, M. 2002. Community structure in social and biological networks. *Proc. National Acad. Sci. 99*, 12, 7821.

GUHA, R., KUMAR, R., RAGHAVAN, P., AND TOMKINS, A. 2004. Propagation of trust and distrust. In *Proceedings of the 13th International Conference on World Wide Web*. ACM Press, New York, 403–412.

HAGBERG, A. A., SCHULT, D. A., AND SWART, P. J. 2008. Exploring network structure, dynamics, and function using networkx. In *Proceedings of the 7th Python in Science Conference (SciPy'08)*.

HALL, M., FRANK, E., HOLMES, G., PFAHRINGER, B., REUTEMANN, P., AND WITTEN, I. H. 2009. The weka data mining software: An update. *SIGKDD Explor. Newslett. 11*, 10–18.

HASAN, M. A., CHAOJI, V., SALEM, S., AND ZAKI, M. 2006. Link prediction using supervised learning. In *Proceedings of the SDM Workshop of Link Analysis, Counterterrorism and Security*.

HASAN, M. A. AND ZAKI, M. J. 2011. *Social Network Data Analytics*. Springer.

HUANG, Z., LI, X., AND CHEN, H. 2005. Link prediction approach to collaborative filtering. In *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*.

KATZ, L. 1953. A new status index derived from sociometric analysis. *Psychometrika 18*, 1, 39–43.

KOREN, Y. 2009. The bellkor solution to the netflix grand prize. http://www.stat.osu.edu/~dmsl/GrandPrize2009_BPC_BellKor.pdf.

LESKOVEC, J., HUTTENLOCHER, D., AND KLEINBERG, J. 2010. Predicting positive and negative links in online social networks. In *Proceedings of the 19th International Conference on World Wide Web*. ACM Press, New York, 641–650.

LIBEN-NOWELL, D. AND KLEINBERG, J. 2007. The link-prediction problem for social networks. *J. Amer. Soc. Inf. Sci. Technol. 58*, 7, 1019–1031.

LICHTENWALTER, R., LUSSIER, J., AND CHAWLA, N. 2010. New perspectives and methods in link prediction. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Press, New York, 243–252.

MAZUROWSKI, M., HABAS, P., ZURADA, J., LO, J., BAKER, J., AND TOURASSI, G. 2008. Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural Netw. 21*, 2, 427–436.

MENON, A. AND ELKAN, C. 2011. Link prediction via matrix factorization. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD'11)*. 437–452.

MISLOVE, A., MARCON, M., GUMMADI, K. P., DRUSCHEL, P., AND BHATTACHARJEE, B. 2007. Measurement and analysis of online social networks. In *Proceedings of the 5th ACM/Usenix Internet Measurement Conference (IMC'07)*.

NACHBAR, D. 2010. IJCNN social network challenge. http://www.kaggle.com/c/socialNetwork/Data.

Narayanan, A., Shi, E., and Rubinstein, B. 2011. Link prediction by de-anonymization: How we won the kaggle social network challenge. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN'11)*. 1825–1834.

Sa, H. R. and Prudencio, R. B. C. 2010. Supervised learning for link prediction in weighted networks. In *Proceedings of the 3rd International Workshop on Web and Text Intelligence*.

Song, H. H., Cho, T. W., Dave, V., Zhang, Y., and Qiu, L. 2009. Scalable proximity estimation and link prediction in online social networks. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference (IMC'09)*. ACM Press, New York, 322–335.

Tan, P. N., Steinbach, M., and Kumar, V. 2005. *Introduction to Data Mining*. Addison Wesley.

Viswanath, B., Mislove, A., Cha, M., and Gummadi, K. P. 2009. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Social Networks (WOSN'09)*.

Watts, D. J. and Strogatz, S. H. 1998. Collective dynamics of 'small-world' networks. *Nature 393*, 440–442.

Weiss, G. and Provost, F. 2003. Learning when training data are costly: The effect of class distribution on tree induction. *J. Artif. Intell. Res. 19*, 315–354.

Zafarani, R. and Liu, H. 2009. Social computing data repository at asu. http://socialcomputing.asu.edu.

Received February 2012; revised October 2012; accepted November 2012