JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 30, 587-603 (2014)

DOI:10.1688/JISE.2014.30.3.3

Bipartite Editing Prediction in Wikipedia

YANG-JUI CHANG, YU-CHUAN TSAI AND HUNG-YU KAO
Department of Computer Science and Information Engineering
National Cheng Kung University
Tainan, 701 Taiwan
E-mail: cjchang@ikmlab.csie.ncku.edu.tw; {p7894131; hykao}@mail.ncku.edu.tw

Link prediction problems aim to project future interactions among members in a social network that have not communicated with each other in the past. Classical approaches for link prediction usually use local information, which considers the similarity of two nodes, or structural information such as the immediate neighborhood. However, when using a bipartite graph to represent activity, there is no straightforward similarity measurement between two linking nodes. However, when a bipartite graph shows two nodes of different types, they will not have any common neighbors, so the local model will need to be adjusted if the users' goal is to predict bipartite relations. In addition to local information regarding similarity, when dealing with link predictions in a social network, it is natural to employ community information to improve the prediction accuracy. In this paper, we address the link prediction problem in the bipartite editing graph used in Wikipedia and also examine the structure of community in this edit graph. As Wikipedia is one of the successful member-maintained online communities, extracting the community information and solving its bipartite link prediction problem will shed light on the process of content creation. In addition, to the best of our knowledge, the problem of using community information in bipartite for predicting the link occurrence has not been clearly addressed. Hence we have designed and integrated two bipartite-specific approaches to predict the link occurrence: First, the supervised learning approach, which is built around the adjusted features of a local model and, second, the community-awareness approach, which utilizes community information. Experiments conducted on the Wikipedia collection show that in terms of *F*1-measure, our approaches generates an 11% improvement over the general methods based on the *K*-Nearest Neighbor. In addition to this, we also investigate the structure of communities in the editing network and suggest a different approach to examining the communities involved in Wikipedia.

Keywords: Wikipedia, link prediction, bipartite graph, social network, edit-network

1. INTRODUCTION

Given a snapshot of a social network, projecting which new interactions among its members are likely to occur in the near future is formally known as "the link prediction problem." The goal of link prediction is to understand which measures of "proximity" in a network lead to most accurate prediction [16]. Common approaches often consider the network as a general graph but not as a bipartite.

In this paper, we focus on link predictions for bipartite networks for Wikipedia. Bipartite refers to an important class in networks, a class which contains edges between two types of entities, such as instant item rating graphs, authorship graphs and document-feature networks. While bipartite graphs are a special case within the category of general graphs, popular local link prediction methods cannot be generalized to these

Received February 28, 2013; accepted June 15, 2013.
Communicated by Hung-Yu Kao, Tzung-Pei Hong, Takahira Yamaguchi, Yau-Hwang Kuo, and Vincent Shin-Mu Tseng.

graphs. Link prediction problems are usually defined in relation to unipartite graphs, and local link prediction functions are limited by their dependence on the immediate neighborhood of the two nodes being considered. In a bipartite network, two nodes belonging to different clusters will not have any common neighbors, and therefore the local methods cannot predict the edge occurrence without any specialization.

Many unipartite networks can be reinterpreted as bipartite networks when edges are modeled as vertices, such as in co-authorship networks. For these bipartite cases, special link prediction algorithms are necessary. In earlier works, Benchettara *et al.* [4] created a unimodal graph from bipartite by projecting the graph over one type of its entities. This leads to two variations of link prediction problems: predicting links in a bipartite graph and predicting links in a unimodal graph. In experiments on the original graph and its unimodal graph, analysis reveals that taking into account the bipartite nature of the graph can substantially enhance the accuracy of the prediction model.

Studies of link prediction problems have played an important role in social network evolution, serving as fundamental questions [6, 7, 13]. In fact, a number of areas can benefit from promising interactions or collaborations that have not yet been utilized within social network. The Wikipedia network is similar Web linked structure [9]. In this paper, we focus on the domain of the edit-network in Wikipedia. There are three main reasons this is worthy of study.

First, Kittur and Kraut [11] have examined the development of and interactions between coordination and conflict in samples taken from several wiki production groups, and they find that the coordination mechanisms of Wikipedia, such as article talk and user talk, may be a social benefit to communication between editors, and as such reduces the likelihood of conflict between them. Thus, among the editors of Wikipedia, there are fruitful interactions (such as co-authorship) which are meaningful and can help improve the performance of prediction models.

Second, in regards to member-maintained online communities, social science theory suggests that reducing the cost of contribution will increase members' motivation to participate [3]. With this in mind, it should be noted that the link prediction approach, particularly for Wikipedia, is similar to task recommendation. Thus, it can reduce the cost of finding articles that align with editors' interests. We can therefore improve the quality and quantity of the articles on Wikipedia.

Third, over the past decade Wikipedia has become the largest online collaborative encyclopedia, even more remarkable because of its egalitarian, democratic, and transparent nature: it can be edited by anyone on the Internet, and its entire editing history has been made publicly available as well. In addition, Wikipedia contains a wealth of information and data, such as general terms, domain-specific lexicons, and named entities which belong to different fields. Among these fields, there are many kinds of relations, such as redirection, categories, disambiguation, and internal links for words that are semantically relevant to a given context. This enormous scalability and its complete link structure have made Wikipedia an invaluable and convenient research resource. These are reasons why we have chosen it as our edit-network construction.

For the above reasons we have constructed the edit graph using the history revisions of Wikipedia, which we have reformulated as a link prediction problem expressed as a two-class discrimination problem. We then extracted several features from the edit graph and fed them into a machine learning algorithm. The aim of this work is to build a super-

vised machine learning approach for bipartite link prediction.

The rest of this paper is organized as follows. In Section 2, we offer a brief overview of the related research; in Section 3, we describe the details of our approach, the system flow chart, and other experimental setup; in Section 4, we give the description of the dataset and how we obtain our samples; and in Section 5 we show the results we obtained. Finally in Section 6 we discuss conclusions and future directions for useful related research work.

2. RELATED WORK

Various link prediction approaches have been proposed in scientific literature related to this topic. A brief overview of various approaches to link prediction is given as follows.

2.1 Neighborhood based Methods

For a node x, let $\Gamma(x)$ denote the set of neighbors of x in graph G. A number of approaches are based on the idea that two nodes x and y are more likely to form a link in the future if their sets of neighbors $\Gamma(x)$ and $\Gamma(y)$ have large overlap; this follows the natural intuition that if nodes x and y represent authors with many colleagues in common, they are more likely to come into contact themselves. These methods are also classified as triangle-closing models in [15]. The most frequently used neighborhood based attributes are the following:

Common Neighbors This is the number of neighbors that x and y have in common. Newman [18] has computed this quantity in the context of collaboration networks, verifying a correlation between the number of common neighbors of x and y at time t, and the probability that they will collaborate in the future. This measure is defined as:

$$score(x, y) = |\Gamma(x) \cap \Gamma(y)|. \quad (1)$$

Jaccard's Coefficient This is a commonly used similarity metric in information retrieval [20]. It measures the probability that both x and y have a feature f, for a randomly selected feature f that either x or y has. If we take features here to be neighbors in G, the measure may be defined as:

$$score(x, y) = |\Gamma(x) \cap \Gamma(y)| / |\Gamma(x) \cup \Gamma(y)|. \quad (2)$$

Adamic/Adar Adamic and Adar [1] propose that friendship between two persons can be predicted by measuring their similarity to each other. The simple similarity can be measured by shared items, which are weighted towards uniqueness, *i.e.* attributes shared only by these two people (and not by others) are weighted more heavily than attributes shared among many people. Therefore the measure is defined as follows:

$$score(x, y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log |\Gamma(z)|}. \quad (3)$$

Preferential Attachment Preferential attachment has received considerable attention as a model of the growth model of networks [17]. The basic idea is that the probability that a new edge will be incident on a node is proportional to the size of that node's current neighborhood. Barabasi *et al.* [2] have verified this idea through empirical analysis. This measure is defined as:

$$score(x, y) = |\Gamma(x)| \bullet |\Gamma(y)|. \quad (4)$$

However, all the above neighborhood based methods, except the preferential attachment model, are not applicable to bipartite graphs because they make some assumptions based on the triangle-closing model: *triangle closing* and c*lustering*. The former asserts that new edges tend to form triangles and the latter indicates that nodes tend to form well-connected clusters in the graph.

In bipartite graphs these assumptions are not true, since neither triangles nor larger cliques can appear. Because two vertices from different clusters which connect do not have any common neighbor, methods based on common neighbors are irrelevant.

2.2 Distance Based Methods

A number of methods refine the notion of shortest-path distance by implicitly considering the ensemble of all paths between two nodes.

Shortest Distance This is the shortest distance between two nodes. It is a basic approach that ranks node-pairs $<x, y>$ by the length of their shortest path in G. Such a measure follows the idea that collaboration networks are "small worlds," in which members are related through a short chain with a small number of links [19].

Katz Another distance based measure frequently used in affiliation analyses between nodes in a graph is the measure proposed in [10]: it is generated by computing a weighted sum of all paths between x and y.

In a bipartite system, although the distance based measure of two vertices from different clusters can be computed with edges that cross the cluster, this is still unable to account for all cases, especially those with an unreachable side in the node-pair.

2.3 Methods Based on Random Walk

These approaches often predict edge occurrence using random walk. Random walk is a Markov chain describing the sequence of nodes visited by a random walker [21]. This process can be described by a transition probability matrix P.

2.4 Methods Based on Similarity

Commonly, two nodes are more likely to be connected if they are more similar, where a latent assumption is that the link itself indicates a similarity between the two endpoints and this similarity can be transferred through the links.

2.5 Methods Based on Temporal Information

The number of approaches based on temporal information is relatively small. In other words, links established at different times are considered to have the same likelihood of connection, when analyzed from most approaches. Recently, the temporal issue is taken into consideration in several studies. Examples are given in [22].

2.6 Higher-Level Approaches

A number of other methods can be used in conjunction with any of the methods discussed above.

Unseen Bigrams Link prediction is similar to the problem of estimating frequencies for unseen bigrams in language modeling: pairs of words that co-occur in a test corpus, but not in the corresponding training corpus (*e.g.* [5]). Related ideas are given in [14].

Clustering The performance of a predictor might improve through a clustering procedure, which means running the predictor on a "cleaned-up" graph after deleting some "tenuous" edges in G. In the clustering procedure, one first computes the score(u, v) for all edges in G, and then deletes the ρ fraction of the edges with the lowest score. Once the graph has been cleaned up, the score(x, y) for the remaining edges can be recomputed; in this way the similarity of node-pairs is determined using only the edges. This approach gives more trustworthy results through the considered measure.

2.6 Network Community Profile Plot

Leskovec *et al.* [15] explored several questions related to identifying meaningful communities in social and information networks. Most networks may be viewed as having a "core," with no obvious underlying geometry and which contains a constant fraction of the nodes, and then there are a large number of relatively small "whiskers" that only tenuously connected to the core. Using the Network Community Profile Plot, they have observed that as a function of increasing size, the best possible communities become more and more "blended into" the remainder of the network.

3. APPROACH

For predicting whether an edge is likely to form between the particular node-pair in the future, we have built a classifier and trained it with the features extracted from the past snapshot of the edit-network, and then evaluated the trained models on the testing samples in the time period following the training period. Let G_{obs} be the observed graph that summarizes in some way the temporal sequence $G = \langle G_{t1}, \ldots, G_{tn} \rangle$. G_{tn+1} is referred as the labeling graph. As in many other works, G_{obs} is computed as the union of all snapshots in the sequence G. Two examples will be generated for each couple of nodes $<x, y>$ such that x and y belong to both G_{obs} and G_{tn+1}. Let the time sequence be $\{t_1, t_2, \ldots, t_n, t_{n+1}\}$, and partition the data into two sub-ranges. The training samples, which are the

observed snapshots of the graph, are obtained from the first sub-range, $[t_1, t_n]$, and the testing samples are from the second sub-range $[t_n, t_{n+1}]$. The positive examples are the editor-article pairs that do not have an edge between them in t_1, but have an edge by t_n, meaning that the editor edited that particular article during this time frame. The negative examples are those that do not have an edge between the pair, both in t_1 and t_n, representing that the editor did not edit that article. We train our models with samples from $[t_1, t_n]$. Then, we make predictions with our trained models on editor-article pairs in t_n. Finally we evaluate our predictions by examining t_{n+1}. We have chosen SVM to be our classifier.

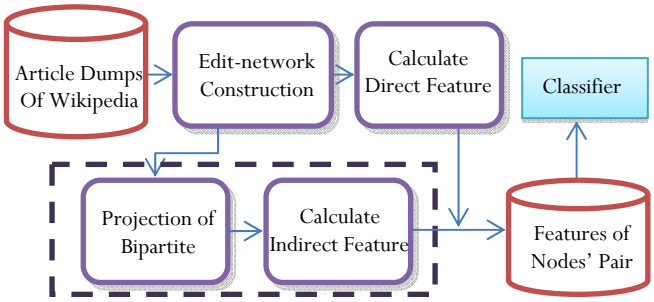

Fig. 1. Flowchart of our approach.

3.1 Baseline

(1) Association Rule (AR)

We apply the association rule algorithm of the general version to the edit graph. In the edit network, the editors are modeled as items, and the set of all the revisions of an article are modeled as a transaction. The rules derive from frequent itemset can be considered as the sets of editors that edit the same set of articles. Given an editor-article pair $<E_i, A_j>$, we want to predict if a new link will occur in the future. We check the association rules that contain E_i, and count the number of editors in the association rule that have edited A_j. Using this baseline method, the greater the number of editors, the more likely an editor will edit that article.

(2) *K*-Nearest Neighbor (KNN)

We set the editing record as the main feature and classify each editor-article pair as positive (linked) or negative (non-linked) by finding the nearest editors. Since the adjacency matrix of the edit network is quite sparse, the majority of the editors are biased and we cannot use the majority of the editors to classify the pairs. Instead, we count the number of editors that have edited the article being tested. When we extract a list of nearest neighbors for each editor, which has a size *K*, we find it is likely that an editor will edit the articles edited by his nearest neighbors. Just because two editors have similar editing behavior does not always mean that they are interested in the same topics.

(3) Graph Partitioning (GP)

In an edit network, we consider the articles as the tasks. To predict the links occurring between editor-article pairs is similar to recommending articles for editors. The

number of edges between partitions in the edit network should be minimized based on the graph partitioning algorithm. Unlike the community in a social network, the size of each partition is regularized. However, the partition can still be seen as a sort of community if we take different p.

3.2 Supervised Leaning Approach

We now describe our two main approaches in detail. The first approach is summarized in Fig. 2. For predicting whether an edge is likely to form between a particular node-pair, we build a classifier and train it with the features extracted from the past snapshot of the edit-network and then evaluate the trained models on the testing samples in the time period following the training period. We train our models with samples from $[t_1, t_n]$ and then make predictions with our trained models on editor-article pairs in t_n. Finally we evaluate our predictions by examining t_{n+1}.

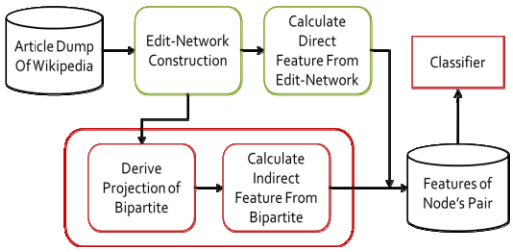

Fig. 2. Supervised learning approach.

Fig. 3. Community-awareness approach.

3.3 Community-Awareness Approach

The second approach is illustrated in Fig. 3. In the community detection module, we employ the Stanford Network Analysis Project (SNAP). Using the local spectral algorithm [3] and each editor as a seed, we cut a set of nodes for each seed-editor in the network.

In the unipartite graph, all nodes are of the same type; the cut-set can be considered as a community straightforwardly. However, since we apply this approach to bipartite, the set of nodes will contain two types of entities: editor and article. Because the cut-set with smaller conductance means that the number of edges inside the set is relatively greater than the number of cut-edges, we can reinterpret the cut-set of bipartite as a community of editors together with the articles that they are more interested in.

If we project the bipartite to the editor-side, we will get a co-edit graph. In the co-edit graph, the weight of the link between editor-nodes represents the amount of articles edited by them directly. Intuitively, we can seek the relevance between editors through the co-edit links. However, in this way we will not be able to find the relevance between two editors if they do not edit the same article, even though there may be relevance between these editors. Our goal here is to find the relevance for these cases. Therefore, in addition to the cut-set for each seed-editor, we also introduce a weight ω determined by the notion of intersection between two cut-sets:

$\varpi = \frac{|S_i \cap S_j|}{|S_i \cup S_j|}$, where S_i and S_j are two cut-sets derived from editor e_i and e_j.

With the intersection score, we can expand the original cut-set by jumping across the different cut-sets. We first set up a threshold Ω and examine the intersection of two cut-sets for each editor-pair (e_i, e_j). If the weight ω reaches the score Ω, we then create an edge between these two editors since the article-nodes in the intersection are the potential articles that will interest the editors in both sets to edit. The edge can be considered as an indicator of the relevance of topics. Using these edges, we can build an editor-intersection network and extract a list of editors and articles, which is similar to the list of nearest-neighbors. We use the integer level C to crawl across the network. If the target in editor-article pair is e_i and C = 1, we expand cut-set of e_i with cut-sets of the first neighbor of e_i. If C = 2, we further expand the set with the neighbor of the next level. In Fig. 4, if editor-node (id 2451) is the target editor and C = 1, we will expand the set with its neighbors (yellow nodes). In other words, we will expand the community of id 2451 with the three communities of id 1036, 1268 and 1269.

3.4 Edit-Network Construction

For the edit-network construction, we consider Wikipedia as an undirected bipartite graph, where articles and editors are nodes in the graph, and an edge between a particular editor-article pair represents an editor editing that article at some point in the past. To predict the edge occurrence between an editor-article pair is similar to deciding whether a particular article is a good candidate to recommend to some editors.

The formal definition of the edit-graph is as follows, G is the bipartite graph, with two sets of nodes, E and A, and a set of edges L, where an edge exists between some $e \in E$ and $a \in A$ if editor e has edited article a at some time point in the past.

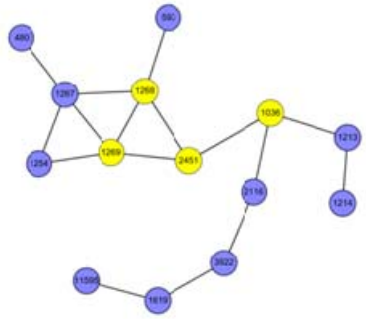

Fig. 4. Example of an editor network built by intersection.

3.5 Features of a Node Pair

The main component of our approach is to come up with a list of features that we believe are informative to feed into the machine learning algorithms. Since the edit-graph is a bipartite, we adapted the measures mentioned above, which are commonly used in

unipartite graphs. First, however, we must adopt the following notation: for a node x, we define $\Gamma(x)$ to be the set of x's neighbors, and $\Gamma'(x) = \bigcap_{y \in \Gamma(x)} \Gamma(y)$, which is the set of x's neighbors' neighbors.

(a) Direct Feature

We define the following measures as direct features based on the equations mentioned above, since they can be derived from the original bipartite graph directly without any transformation of the graph:

SN (Sum of Neighbors) For an article, this is the number of editors that edited it. For an editor, it is the number of articles he/she has edited. Therefore each pair will have two referents, one for editor, one for article. The sum of neighbors might be meaningful because the more articles an editor has edited, the more likely he/she will edit more articles because it suggests that he/she is more active than those who have edited fewer articles. We denote both as *SN*.

CN (Common Neighbors) For an editor-article pair $<e, a>$, common neighbors is defined to be $|\Gamma(e) \cap \Gamma'(a)|$. This feature is adapted to form the intersection of the articles that editor e has edited, and the articles that edited by who edited the article a. This basically captures the idea of "people who edited this article also edited …" We denote this feature as *CN*.

JC (Jaccard's Coefficient) Similar to the common neighbors, this feature measures the similarities between two sets. In the bipartite graph, this is defined as $|\Gamma(e) \cap \Gamma'(a)| / |\Gamma(e) \cap \Gamma'(a)|$. It is the normalized version of common neighbors, which should be more informative. We denote this feature as *JC*.

AA (Adamic/Adar) This feature uses the frequency of common features to compute the similarity between two nodes. In the bipartite graph, the feature is the neighbors, and this feature is defined as $\sum_{z \in \Gamma(e) \cap \Gamma'(a)} 1/\log|\Gamma(z)|$. This feature is used to capture the notion that if in the intersection of articles the number of editors that edited a particular article is smaller than for other articles, then to some extent this article is more important to be associated with this editor than other editors. We denote this feature as *AA*.

PA (Preferential Attachment) This is similar to "the sum of neighbors" measure above and suggests how active the editor is and how popular the article is. Taking the scalar multiplication of the two features can quantify the neighborhood's size. It is defined as $|\Gamma(e)| \times |\Gamma(a)|$, and we denote it as *PA*.

SD (Shortest Distance) This is the minimum hop count between an editor and an article. We hypothesize that the shorter the distance between an editor and an article, the more likely the editor will be to edit the article. We denote this feature as *SD*.

All the above features, except Sum of Neighbors, can be categorized into a set of topological features which characterizes the roles of nodes in the unipartite network. For a bipartite graph, the meaning of topological features is slightly different than for a unipartite one. The links in a unipartite graph are between nodes of the same type, while the links in a bipartite graph are between nodes from different clusters.

(b) Indirect Feature

In the following approach we apply the same general idea of works as presented in [4]. We formulate the link prediction as a supervised learning problem. The goal is to discriminate between linked classes (positive examples) against not-linked classes (negative examples.) A bipartite graph is defined as follows: $G = <X, Y, E>$ where X and Y are two mutually exclusive sets of nodes. E is a set of edges of G and is a subset of $X \times Y$. A unimodal graph can be obtained from a bipartite graph by projecting the graph over one of its nodes' sets. For example, the projection over the set of X is defined by a unimodal graph where nodes from X are tied if they are linked to at least n common nodes in the initial bipartite graph G. To express this in a more formal way, let $\Gamma_g(x)$ bet the set of neighbors of node x in a graph g. Projections of a bipartite graph G are then defined as follows:

$$G_X^n = (V_X \subseteq X, E = \{(a, b): a, b \in X, |\Gamma_G(a) \cap \Gamma_G(b)| \geq n\}) \quad (5)$$

$$G_Y^m = (V_Y \subseteq X, E = \{(a, b): a, b \in Y, |\Gamma_G(a) \cap \Gamma_G(b)| \geq m\}) \quad (6)$$

With these unimodal graphs, we can define the indirect features. Let $f_G(e \in E, a \in A)$ be a direct feature. In comparison with the direct features, the indirect features cannot be computed from the original bipartite graph. With a unimodal graph, we can introduce the following two indirect features:

$$\varphi_{u \in \Gamma_G(a)} f^i_{G_E^n}(e,u) \quad (7)$$

$$\varphi_{v \in \Gamma_G(e)} f^i_{G_E^m}(v,a) \quad (8)$$

These two features are built from f, and are computed in the projected graph as G_X and G_Y, respectively. Without loss of generality, let us consider the first indirect feature. It computes f between e and all neighbors of a in G_E. An aggregate function φ selects the most appropriate value from the set {min, max}, depending on the feature. For example, if the feature is the number of shared neighbors, φ will be the max function. If the feature is the shortest distance, φ will be the min. function. Notice that both projection parameters n and m can take different values.

4. EXPERIMENT EVALUATION

After choosing editor-article pairs, we computed the aforementioned features on all the pairs for the training and testing sets and obtained 460,569 samples in the training set, and 227,722 samples in the testing set. Both sets are balanced with 50% positive and 50% negative samples. The statistics for the sample collection is shown in Table 1.

We also checked the number of different groups of editors. As the number of editors and pairs are shown in Tables 2 and 3, we can see that only 40% of the editors make revisions in new articles that they have never written before. Furthermore, it is not surprising that the editors identified only by IP-address contribute less (32.4%) than registered editors (49.1%), while they both account for half of the editor-nodes in training graphs. The bot group is much smaller than the others; however the number of pairs contributed by bots is close to the group of IP-address.

Table 1. Number of the sample collection.

	Editors	Articles	Total pairs	Positive pairs	Negative pairs
Training	48,406	10,657	460,569	231,589	228,980
Testing	19,798	10,657	227,722	114,481	113,241

Table 2. Number of editors.

	Editors	Registered	Bot	IP
Training	48,406	24,268	162	23,976
Testing	19,798	11,934	79	7,785

Table 3. The number of pairs of different editor-groups in testing graph.

	Positive class	Negative class	Both classes
Bot	13,618	12,585	26,203
IP	15,024	15,022	30,046
Registered	85,839	85,634	171,473
Total	114,481	113,241	227,722

We use three common metrics in our evaluation: precision, recall and F1-value. F1-value is the aggregation of precision and recall, which considers both. For the link prediction of Wikipedia, since the number of articles in Wikipedia is quite large, we can pay more attention on the precision measure. We randomly divide all the training samples into five groups, each of which is balanced with half positive and half negative. Then we evaluate each group by using a testing set. Since the results for all five groups are similar and we focus on the pros and cons of the features chosen, we take the average to be the precision score of the prediction model. The results are shown in Table 5.

Depending on how the different features are combined, these results give us a sense of how well the models can predict whether links will form in the Wikipedia edit graph. Since we are more interested in recommending articles to authors than in filtering out unwanted articles for one specific editor, we will concentrate on the prediction of the positive samples. When doing this, the most suitable measures to use for comparison are precision and recall for positive classes. The F-value is a harmonic mean of precision and recall, so it takes both values into account. Note that the Neighborhood is the set of features: Common Neighbor (CN), Jaccard's Coefficient (JC), Adamic/Adar (AA) and Preferential Attachment (PA).

The Sum of the Neighbors (SN) model has the highest precision, 0.73, when compared with the other combinations of features. However, when the Shortest Distance (SD) and Neighborhood are taken into account, the precision is reduced to 0.68. If we exclude the SD, the precision will recover to 0.69. When SD is taken into account, the recall of positive class is reduced to 0.40, which is quite low. The reason that SD does not perform well might be because the category of articles dumped is not broad enough. Since all the articles are in the same category, it is easy to link an editor to any article that he/she has never worked on. Thus, using the SD can result in lots of false positive samples. However, if we take the weighted distance, the results might be different.

It is interesting to note that if we combine the indirect Common Neighbor with SN, the recall of positive class will climb to 0.79. In the projection of authors, the indirect common neighbor stands for the number of co-authors. The reason why other features do

not work well might be the characteristics of different classes of editors. We divide the result into three classes, Bot (BOT), IP Address (IP) and Registered User (REG). The results are shown in Table 4. We can see that the F-value of other features has a better performance for REG class. BOTs are programmed and anonymous users seldom contribute to articles. Therefore, we can focus on the editors of the REG class, which shows that the more informative features such as JC and AA still lead us to a better result.

Table 4. F1-measure of different editor-classes.

	BOT	IP
SN	0.6540	0.7878
SN+JC	0.6593	0.7876
SN+AA	0.6442	0.7768

Table 5. Results (Neighborhood = {CN+JC+AA+PA}).

Features used	Accuracy	Class	Precision	Recall	*F*-value
SN only	70.60%	0	0.68	0.75	0.71
		1	**0.73**	0.65	0.69
SN+SD+ Neighborhood	60.77%	0	0.57	0.81	0.67
		1	0.68	0.40	0.50
SN+ Neighborhood	66.91%	0	0.65	0.72	0.68
		1	0.69	0.61	0.65
SN+ Indirect CN (n = 5, m = 6)	68.55%	0	0.69	0.66	0.67
		1	0.67	0.71	0.69
SN+ Indirect CN (n = 5, m = 3)	65.43%	0	0.71	0.51	0.59
		1	0.62	**0.79**	0.69

4.1 Supervised Learning Approaches

(a) Pre-evaluation

In this step, we have around 460 thousand pairs to train and 220 thousand pairs to test. To do this, we randomly divide all the samples into five groups for training and testing sets respectively; each set is balanced with half positive and half negative. Once the sets are properly organized, we evaluate each goup with a testing set. The results for the five groups are similar and we focus on the pros and cons of the features chosen, using the average as the performance score of the prediction model. The result is provided in Table 6.

Table 6. Result of SVM on the five subsets.

Features used on the five subsets	Precision	Recall	*F*-measure
SN	0.7315	0.6559	0.6916
SN+JC+AA+PA	0.7308	0.6307	0.6771
SN+JC+AA+PA+CN	0.6926	0.6148	0.6514
SN+JC+AA+PA+CN+SD	0.6864	0.4046	0.5091

The *SN* model is able to achieve a precision level of 0.73, which is higher than the other combinations. However, when the Shortest Distance (SD) and other Neighborhood-based features (JC, AA, PA and CN) are taken into account, the precision is reduced to 0.68 and the recall is reduced to 0.40, which is quite low. If we exclude the SD, the precision will recover to 0.69. The SD might not perform well because the categories of the article dumped are not broad enough. Since all the articles are in the same category, it is easy to link an editor to any article that he/she has never written before. Thus, using the SD can result in lots of false positive samples. However, if we take the weighted distance, the results might be different. In addition to the impact of SD, we can see that the SN can be a dominating feature in the five subsets.

4.2 Results of Supervised Learning Approach and Baseline

After excluding the SD, we apply a supervised learning approach to the whole sample collection. The results are given in Table 9. In the baseline methods, the precision of the Association Rule (AR) is higher than for the other methods (0.77). However, since there are few pairs that correspond to the rules, we have a poor recall using the association rule (0.10).

Due to the relative paucity of rules, the number of pairs that editor corresponds to association rules is not very much. Even if we lower the support from 20 to 10, the increasing number of rules still cannot include all the pairs that are potentially positive. Apart from the number of rules, we also notice that the average size of the frequent editor-set is around four, which represents the average number of editors that will consecutively edit an article. However, it should be noted that the Association Rule does not consider time sequence. Hence, we cannot know which one editor affects the others from these rules. A few rules are listed in Table 7.

Table 7. Rules derived from edit history.

Support	Confidence	Left-hand editor-set	=>	Right-hand editor-set
22	**0.957**	Neonblak	=>	Floydspinky71
28	**0.875**	Supertigerman	=>	Tewapack
10	**0.833**	Faigl.ladislav	=>	Eliyak
46	**0.807**	Chochopk	=>	Japanese Searobin
27	**0.771**	Persian Poet Gal	=>	Can't sleep, clown will eat me
13	**0.765**	Tecmobowl	=>	Masonpatriot
12	**0.750**	Hut 8.5	=>	RexNL

In Table 8, the result of the graph partition baseline is as expected: simply partitioning the graph into equal sized parts fails to represent the different communities in the edit network. Increasing the number of partitions results in smaller sized partitions, which is similar to the smaller k in KNN. Therefore, if we take the number of partitions to be two to four, the result will be similar to KNN of from $k = 100$ to 300. Nevertheless, since partitioning the graph reduces the cost of finding nearest neighbors, the result of GP-KNN is slightly better than KNN if we take various sizes of partitions, as the result of the Graph Partition data in Table 8 shows.

Table 8. Results of KNN and GP.

	Precision	Recall	*F*-measure
KNN	0.653	0.532	0.586
GP-KNN ($P = 4$)	0.657	0.701	0.678
GP-KNN ($P = 3$)	0.644	0.710	0.675
GP-KNN ($P = 2$)	0.583	0.886	0.703

Table 9. Results of SVM on the whole collection.

Features used on whole collection	Precision	Recall	*F*-measure
Association Rule	0.7740	0.1000	0.1770
K-Nearest Neighbor	0.6530	0.5320	0.5860
Graph Partition	0.6940	0.4450	0.5420
SN	0.7442	0.6621	0.7007
SN+JC	**0.7453**	0.6630	**0.7017**
SN+AA	**0.7468**	0.6573	0.6992
SN+PA	0.7388	0.6526	0.6930
SN+JC+PA	0.7400	0.6509	0.6926
SN+AA+PA	0.7294	0.6732	0.7002
SN+JC+AA	**0.7486**	0.6567	0.6996
SN+JC+AA+PA	0.6954	**0.6789**	0.6871
SN+CN+JC+AA	0.7032	**0.6893**	0.6962
SN+CN+AA+PA	0.7300	0.6734	**0.7005**
SN+CN+JC+PA	0.6978	0.6783	0.6879
SN+CN+JC+AA+PA	0.7018	**0.6894**	0.6956

For the results of SVM, although SN is the dominant feature for supervised learning in the smaller subset, the combination of the JC and AA scores can still improve the performance for the whole sample collection.

The reason why other features do not perform much better might be the characteristics of different classes of editors. Therefore we divide the result into three groups, Bot (BOT), IP Address (IP) and Registered User (REG). The precision is shown in Table 10. We can see that the precision of other features can achieve a better performance for REG class. BOT is programmed, and anonymous users seldom contribute to articles. Therefore, we can focus on the editors of the REG class. Analyzing this shows that the more informative features such as JC and AA can still lead us to a better results.

Table 10. Precision on different editor groups.

	TOTAL	**BOT**	**IP**	**REG**
SN	0.7442	0.7004	0.7987	0.7406
SN+AA	0.7468	0.6715	0.8004	**0.7494**
SN+JC+PA	0.7400	0.6750	0.7990	**0.7512**

To observe the performance on the group that we are interested in, we locked in on the REG editors and then filtered out the registered editors that have edited less than 50 times (*i.e.* the number of submitted revisions is smaller than 50) and obtained a set composed of 2115 editors (REG_50). At this point in our study we had around 140 thousand training pairs and 90 thousand testing pairs. The results are given in Table 11.

It is worth noting that if we combine the indirect Common Neighbor (iCN), the results can be improved. In the projection of editors, the indirect common neighbor stands for the number of co-authors. Since it is known that there are large numbers of "talk pages" between the aggressive Wikipedians, it is more likely that one edit an article under the influence of the co-authors. Therefore it is intriguing to detect the community to improve the prediction model.

Table 11. The result on the REG_50 group.

Direct features	Indirect features	Precision	Recall	F-measure
SN		0.6519	0.7131	0.6745
SN+JC+AA		0.7383	0.6487	0.6906
SN+CN+JC+AA		0.7361	0.6579	0.6948
SN	+iCN	0.7105	0.6508	0.6793
SN+JC+AA	+iCN	**0.7407**	**0.6484**	**0.6915**
SN+CN+JC+AA	+iCN	0.7386	0.6560	0.6949

6. DISCUSSION

Our observation shows that nodes with a high degree can still be an important factor to the link prediction model. It is important to note that we only evaluated these algorithms over a small period of time and for a specific category. The supervised link prediction model is based on the amount of historical information available for some editors or articles. Thus, we would like to know what kind of information on the bipartite network can improve the performance significantly in the future. With the right kind of data, perhaps in the future we could predict the edge occurrence even for nodes with low or average degrees. Furthermore, we would like to examine whether different communities exist in Wikipedia and get a better idea of the scale of these communities. Since the community can be a core part of social network, studying the characteristics of different communities might improve the prediction model.

REFERENCES

1. L. A. Adamic and E. Adar, "Friends and neighbors on the Web," *Social Networks*, Vol. 25, 2003, pp. 211-230.
2. A. L. Barabasi, H. Jeong, Z. Neda, E. Ravasz, A. Schubert, and T. Vicsek, "Evolution of the social network of scientific collaborations," *Physica A: Statistical Mechanics and its Applications*, Vol. 311, 2002, pp. 590-614.
3. G. Beenen, K. Ling, X. Wang, K. Chang, D. Frankowski, P. Resnick, and R. E. Kraut, "Using social psychology to motivate contributions to online communities," in *Proceedings of ACM Conference on Computer Supported Cooperative Work*, 2004, pp. 212-221.
4. N. Benchettara, R. Kanawati, and C. Rouveirol, "Supervised machine learning applied to link prediction in bipartite social networks," in *Proceedings of IEEE International Conference on Advances in Social Networks Analysis and Mining*, 2010, pp. 326-330.

5. U. Essen and V. Steinbiss, "Cooccurrence smoothing for stochastic language modeling," in *Proceedings of IEEE International Conference on Acoustics*, *Speech and Signal Processing*, Vol. 1, 1992, pp. 161-164.
6. M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," in *Proceedings of Workshop on Link Analysis*, *Counterterrorism and Security*, 2006.
7. Z. Huang, X. Li, and H. Chen, "Link prediction approach to collaborative filtering," in *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, 2005, pp. 141-142.
8. T. Iba, K. Nemoto, B. Peters, and P. A. Gloor, "Analyzing the creative editing behavior of Wikipedia editors: Through dynamic social network analysis," *Procedia – Social and Behavioral Sciences*, Vol. 2, 2010, pp. 6441-6456.
9. J. Kamps and M. Koolen, "Is Wikipedia link structure different?" in *Proceedings of the 2nd ACM International Conference on Web Search and Data Mining*, 2009, pp. 232-241.
10. L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, Vol. 18, 1953, pp. 39-43.
11. A. Kittur and R. E. Kraut, "Beyond Wikipedia: coordination and conflict in online production groups," in *Proceedings of ACM Conference on Computer Supported Cooperative Work*, 2010, pp. 215-224.
12. J. M. Kleinberg, "The small-world phenomenon: an algorithmic perspective," in *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, 2000, pp. 163-170.
13. J. Kunegis, E. W. D. Luca, and S. Albayrak, "The link prediction problem in bipartite networks," in *Proceedings of the 13th International Conference on Information Processing and Management of Uncertainty*, 2010, pp. 380-389.
14. L. Lee, "Measures of distributional similarity," in *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, 1999, pp. 25-32.
15. J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins, "Microscopic evolution of social networks," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008, pp. 462-470.
16. D. Liben-Nowell and J. Kleinberg, "The link prediction problem for social networks," in *Proceedings of the 12th ACM International Conference on Information and Knowledge Management*, 2003, pp. 556-559.
17. M. Mitzenmacher, "A brief history of generative models for power law and lognormal distributions," *Internet Mathematics*, Vol. 1, 2004, pp. 226-251.
18. M. E. J. Newman, "Clustering and preferential attachment in growing networks," *Physical Review E* (*Statistical*, *Nonlinear*, *and Soft Matter Physics*), Vol. 64, 2001, p. 025102.
19. M. E. J. Newman, "The structure of scientific collaboration networks," in *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 98, 2001, pp. 404-409.
20. G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*, McGraw-Hill, Inc., NY, 1986.
21. J. L. Kemeny and J. L. Snell, *Finite Markov Chains*, Springer-Verlag, Berlin, 1960,

reprint 1976.
22. T. Tylenda, R. Angelova, and S. Bedathur, "Towards time-aware link prediction in evolving social networks," in *Proceedings of the 3rd ACM Workshop on Social Network Mining and Analysis*, 2009, pp. 1-10.
23. J. Voss, "Measuring Wikipedia," in *Proceedings of the 10th International Conference of the International Society for Scientometrics and Informetrics*, 2005.

Yang-Jui Chang (張洋瑞) received his M.S. degree in the Department of Computer Science and Information Education at the National Cheng-Kung University, Taiwan, in 2013. He is a Software R&D Engineer in ASUS Mobile Application eXperience Center of ASUSTek Computer Inc. now.

Yu-Chuan Tsai (蔡毓娟) is the Ph.D. student in the National Cheng Kung University and received her Master's degree from the Department of Information Engineering in I-Shou University. She is currently working in the Library and Information Center, Nation Kaohsiung University. Her research interest is in the area of Web technology and privacy preserving.

Hung-Yu Kao (高宏宇) received his B.S. and M.S. degrees in Computer Science from the National Tsing Hua University, Hsinchu, Taiwan, in 1994 and 1996, respectively. In July 2003, he received his Ph.D. degree from the Electrical Engineering Department, National Taiwan University, Taipei, Taiwan. He was a postdoctoral fellow of the Institute of Information Science (IIS), Academia Sinica, from 2003 to 2004. Dr. Kao is currently an Associate Professor of Computer Science and Information Engineering at the National Cheng Kung University. His research interests include web information retrieval/extraction, search engine, knowledge management, data mining, social network analysis and bioinformatics. He has published more than 40 research papers in refereed international journals and conference proceedings. He is a member of IEEE and ACM.