Generating Useful Network-based Features for Analyzing Social Networks

Abstract

Recently, many Web services such as social networking services, blogs, and collaborative tagging have become widely popular. Many attempts are being made to investigate user interactions by analyzing social networks among users. However, analyzing a social network with attributional data is sometimes not an easy task because numerous ways exist to define features through aggregation of different tables. In this study, we propose an algorithm to identify important network-based features systematically from a given social network to analyze user behavior efficiently and to expand the services. We apply our method for link-based classification and link prediction tasks with two different datasets, i.e., an @cosme (an online viral marketing site) dataset and a Hatena Bookmark (collaborative tagging service) dataset, and show the usefulness of our algorithm. Our algorithm is general and can provide useful network-based features for social network analyses.

Introduction

Recently, numerous studies have been done to process network data. Web 2.0 services such as social networking services, blogs, and collaborative tagging have the characteristic that users mutually interact. Such interaction among users creates a social network among users. Some examples are friends relations among users in social networking services (Adamic & Glance 2005; Ahn *et al.* 2007), comment/trackback/blogroll relations among blogs (Furukawa *et al.* 2007), and similarity networks of tags, users, and resources in collaborative tagging systems (Golder & Huberman 2006; Mika 2007).

Many attempts are being made to analyze user interactions by analyzing social networks among users. For example, L. Backstrom et al. analyzed the social groups and community structure on LiveJournal and DBLP data (Backstrom *et al.* 2006). They inferred eight community features and six individual features, and reported that one feature is unexpectedly effective: an individual with friends in a group is significantly more likely to join if these friends are themselves mutual friends than if they are not. Apparently, greater potential exists for such new features using a network structure, which is the motivation of this research.

Analyzing social networks is important for various AI topics: For example, there are many attempts to apply social networks for recommendation, marketing, information gathering, and so on (Staab *et al.* 2005). Ontology creation and social network is related each other (Mika 2005), quote “social networks and semantics are just flip-sides of the same coin.”

Social networks have been of interest also in data mining community. *Link mining* (Getoor & Diehl 2005) is a recently-developed research area in the intersection of works in link analysis, web mining, relational learning, and so on. An actively studied task in link mining is *link-based classification*, i.e., classifying samples using the relations and links that are present among them. Another prominent task is *link prediction*, predicting whether there would be a link between a pair of nodes (in the future) given the (previously) observed links. However, an important difficulty lies in the fact that numerous methods exist to aggregate features for link-based classification and link prediction. The network structure among users influences each user in a different way. Therefore, it is difficult to know the useful feature aggregation in advance.

In this paper, we propose an algorithm to identify important network-based features systematically from a given social network to analyze user behavior efficiently. In our approach, we first define general operators that are applicable to the social network. Then the combinations of the operators provide different features, some of which correspond to traditional social network indices; others are considered to be new. We apply our method for both link-based classification and link prediction on two different datasets, i.e., the @cosme dataset and the Hatena Bookmark dataset. The former, @cosme, is the largest online community site for women in Japan in which users can review cosmetic products and register other users as friends. Hatena Bookmark is a largest collaborative tagging service for Web pages in Japan. Using these datasets, we demonstrate the effectiveness of our approach; generating various kinds of network-based features, the performance of link-based classification and link prediction increase compared to existing approaches.

This paper is organized as follows. The next section presents related works and introduction of various indices in social network analysis. Then, we propose our method for feature generation by defining node sets, operators, and

Table 1: Features used in link-based classification (Backstrom *et al.* 2006).

Number of friends in community
Number of adjacent pairs in S
Number of pairs in S connected via a path in E_C
Average distance between friends connected via a path in E_C
Number of community members reachable from S using edges in E_C
Average distance from S to reachable community members using edges in E_C

- S denotes the set of friends of an individual.
- E_c denotes the set of edges in the community C.

aggregation methods. Finally, we describes experimental results for two datasets, followed by relevant discussion and conclusions.

Related Works

Features used in Social Network Analysis

In this paper, we propose an algorithm that can systemically generate network-based features that is useful for link-based classification and link prediction. We first explain commonly used features / indices in social network analysis and complex network studies. We call such attributes *network-based features* throughout the paper.

A simple feature of a network is its *density*. It describes the general level of linkage among the network nodes. The graph density is defined as the number of edges in a (sub-)graph, expressed as a proportion of the maximum possible number of edges. *Centrality measures* are popular indices of a node. They measure the structural importance of a node, e.g. the power of individual actors. There are several centrality measures such as the degree, closeness and betweenness centrality (Freeman 1979). *Structural equivalence* and *structural holes* are useful concepts in social network analysis. There are other indices popularly used in complex network theories: *Characteristic path length* is the average distance between any two nodes in the network (or a component). *Clustering coefficient* is the proportion of edges between the nodes within a node's neighborhood divided by the number of edges that can possibly exist between them.

We do not explain all indices, but readers can consult literature related to social network analysis (Wasserman & Faust 1994; Scott 2000).

Other Features used in Related Works

There are other features that are are often used in literature. We introduce some features for link-based classification and link prediction.

L. Backstrom et al. (2006) analyzes community evolution, and shows some *structural features* characterizing individuals' positions in the network. The examples of these features are shown in Table 1. D. Liben-Nowell et al. (2003) have elucidated features using network structures for link prediction. In the link prediction problem, one can predict whether the link will exist between nodes x and y by assigning the connection weight $score(x,y)$ to pairs of nodes x and y. In their paper, they define $score(x,y)$ based on various network-based features as shown in Table 2.

Table 2: Features used in link prediction (Liben-Nowell & Kleinberg 2003).

name	feature
graphic distance	d_{xy}
common neighbors	$\lvert\Gamma(x)\cap\Gamma(y)\rvert)$
Jaccard's coefficient	$\frac{\lvert\Gamma(x)\cap\Gamma(y)\rvert}{\lvert\Gamma(x)\cup\Gamma(y)\rvert}$
Adamic / Adar	$\sum_{z\in\Gamma(x)\cap\Gamma(y)} \frac{1}{log\lvert\Gamma(z)\rvert}$
preferential attachment	$\lvert\Gamma(x)\rvert\cdot\lvert\Gamma(y)\rvert)$

- d_{xy} is the distance between node x and y.
- $\Gamma(x)$ is the set of nodes adjacent to node x.

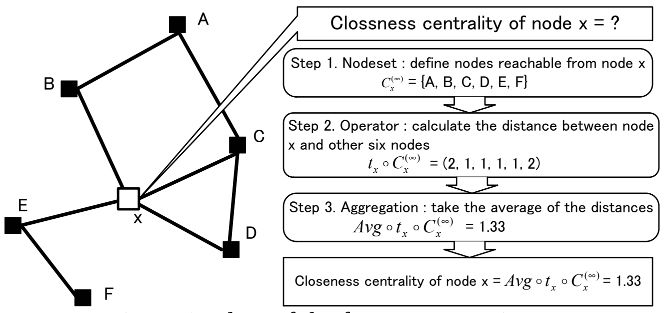
A
B
C
D
E
F
x
Clossness centrality of node x = ?
Step 1. Nodeset : define nodes reachable from node x
$C_x^{(\infty)}$ = {A, B, C, D, E, F}
Step 2. Operator : calculate the distance between node x and other six nodes
$t_x \circ C_x^{(\infty)}$ = (2, 1, 1, 1, 1, 2)
Step 3. Aggregation : take the average of the distances
$Avg \circ t_x \circ C_x^{(\infty)}$ = 1.33
Closeness centrality of node x = $Avg \circ t_x \circ C_x^{(\infty)}$ = 1.33

Figure 1: Flow of the feature generation

Methodology

In this section, we define elaborate operators to generate social-network features for link-based classification and link prediction. Using our model, we can generate features that are often used in social network analysis in addition to new features. Our intuition is simple: recognizing that traditional studies in social science have demonstrated the usefulness of several indices, we can assume that feature generation toward the indices is also useful.

How can we design the operators to construct various types of social network features effectively? Through trial and error, we designed systematic feature generation in three (or optionally four) steps:

Step 1 We first select a set of nodes.

Step 2 The operators are applied to the set of nodes to produce a list of values.

Step 3 The values are aggregated into a single feature value.

Step 4 Optionally, we can take the average, difference, or product of two values obtained in Step 3.

For example, when calculating the closeness centrality[1] of node x, we can discern its value basically in three steps as shown in Fig. 1: we first select reachable nodes from x; secondly, we calculate the distance between node x and each node; finally, we take the average of these distances. In addition, we can get the value of the closeness centrality of node x. Therefore, we can eventually construct indices used in social network analysis, through these steps. Below, we explain each step in detail.

Step 1: Defining a Node Set

First, we define a node set. We consider two types of node sets as follows: one is based on a network structure; the other is based on the category of a node.

[1]Average distance from node x to all others

Most straightforwardly, we can choose the nodes that are adjacent to node x. The nodes are, in other words, those of distance one from node x. The nodes with distance two, three, and so on are definable as well. This node set is defined by the distance from node x. We define a set of nodes $C_x^{(k)}$ as a set of nodes within distance k from x. For example, we can denote the node set adjacent to node x as $C_x^{(1)}$.

We can define a set of nodes with a particular value of some attribute. This node set is effective for link-based classification because a network constructed using nodes with a particular categorical value is sometimes different from that with a different categorical value. Although various attributes can be targeted theoretically, we specifically examine the value of the category attribute of a node to be classified. We define the node set where the categorical value A is a as $N_{A=a}$. Considering both distance-based and category-based node sets, we can define the conjunction of the sets, e.g., $C_x^{(1)} \cap N_{A=a}$.

Step 2: Operation on a Node Set

Given a node set, we can conduct several calculations for the node set. Below, we define operators with respect to two nodes, and then expand it to a node set with an arbitrary number of nodes.

The simple operation for two nodes is to check whether the two nodes are adjacent or not. We denote this operators as $s^{(1)}(x,y)$, which returns 1 if nodes x and y are connected to each other, and 0 otherwise. We also define operator $t(x,y) = \arg\min_k \{s^{(k)}(x,y) = 1\}$to measure the geodesic distance between the two nodes on the graph. These two operations are applied to each pair of nodes in N if given a set of more than two nodes (denoted as N). This calculation is defined as follows.

$$Operator \circ N = \{Operator(x,y) \mid x \in N, y \in N, x \neq y\}$$

For example, if we are given a node set $\{n_1,\ n_2,\ n_3\}$, we calculate $s^{(1)}(n_1,n_2)$, $s^{(1)}(n_1,n_3)$, and $s^{(1)}(n_2,n_2)$ and return a list of three values, e.g. $(1,0,1)$. We denote this operation as $s^{(1)} \circ N$.

In addition to s and t operations, we define two other operations. One is to measure the distance from node x to each node, denoted as t_x. $t_x \circ N$ measures the distance of each node in N from node x. Another operation is to check the shortest path between two nodes. Operator $u_x(y,z)$ returns 1 if the shortest path between y and z includes node x. Consequently, $u_x \circ N$ returns a set of values for each pair of $y,z \in N$. The other is to calculate the structural equivalence between node x and y. This is denoted as $e_x(y)$.

Step 3: Aggregation of Values

Once we obtain a list of values, several standard operations can be added to the list. Given a list of values, we can take the summation (*Sum*), average (*Avg*), maximum (*Max*), and minimum (*Min*). For example, if we apply *Sum* aggregation to a value list $(1,0,1)$, we obtain a value of 2. We can write the aggregation as e.g., $Sum \circ s^{(1)} \circ N$. Although other operations can be performed, we limit the operations to the four described above. The value obtained here results in the network-based feature for a node x.

Additionally in Step 4, we can aggregate two feature values into single feature values. We can take the difference or the ratio of two obtained values, which is sometimes useful for link-based classification. For example, $Sum \circ s^{(1)} \circ C_x^{(1)}$ divided by $Sum \circ s^{(1)} \circ C_x^{(\infty)}$ can be a feature using two features in Step 3.

For Link Prediction: Relational Features

For link prediction tasks, we generate network-based features which represent a score (i.e. connection weight) on two nodes x and y. Two directions can be taken to generate network-based features depending on two nodes. One is to aggregate two separately obtained values for each node. For example, we can calculate *preferential attachment* ($|\Gamma(x)| \cdot |\Gamma(y)|$) by respectively counting the links of nodes x and y, and thereby obtaining a value by the product of two values. We define operators on the two values as Step 4 especially for link prediction, to take average, maximum, minimum, difference, ratio, and product.

Another approach to generate network-based features on two nodesdefines a node set that is relevant to both node x and y. For example *common neighbors* ($|\Gamma(x) \cap \Gamma(y)|$) depend on the number of common nodes which are adjacent to nodes x and y. Because we have defined the conjunction and disjunction of two sets, the value is obtainable simply by counting the sets (realized using $s^{(\infty)}$).

To cover more features, several operators should be added / modified for link prediction aside from link-based classification. In Step 2, we define an operator γ as $\gamma(x) = \frac{1}{log|\Gamma(x)|}$.We also redefine operators t_x and u_x because we apply these operators to the conjunction and summation of the node set from nodes x and y: The operator u_x is modified as $u_{xy}(z,w)$, which returns 1 if shortest path between z and w includes l_{xy} and 0 otherwise. Therein, l_{xy} signifies the link between nodes x and y, and we assumed that two nodes x and y are connected. The operator t_x is modified as $t_{xy}(z) = Min\{t(x,z), t(y,z)\}$.

Constraints

We summarize the node sets, operators, and aggregations for link-based classification and link prediction in Table 3. The operators are categorized into three steps. (Step 4 is optional.) In each step, input and output are different. Some operators are used for classification, and some are used for link prediction. To show the effectivenss of each operator in experiments, we group operators into Method 1 to Metho 4 (for classification) and into Method 1 and Method 2 (for link prediction).

We have $4 \times 4 \times 4 = 64$ features for link-based classification. Plus, $4 \times 4 \times 2$ more features exist considering Step 4; there are 96 features in total. For link prediction, we can generate 126 features in Method 1 and 160 features in Method 2.

Some resultant features sometimes correspond to well-known indices, as we intended in the design of the operators. For example, we can denote the network density as $Avg \circ s^{(1)} \circ N$, degree of nodes as $Sum \circ t_x \circ C_x^{(1)}$, characteristic path length as $Avg \circ t \circ N$, and betweenness centrality

Table 3: Operator list

Step	Notation	Input	Output	Description	LC*	LP*
1	$C_x^{(k)}$	node x	a node set	nodes within distance k from x	$\surd$ (1)	$\surd$(1)
	$C_y^{(k)}$	node y	a node set	nodes within distance k from x		$\surd$(1)
	$N_{A=a} \cap C_x^{(k)}$	node x	a node set	nodes within distance to x and the attribute A is a	$\surd$ (3)	
	$C_x^{(k)} \cap C_y^{(k)}$	node x and y	a node set	nodes within distance k from x and within distance k from y		$\surd$(2)
	$C_x^{(k)} \cup C_y^{(k)}$	node x and y	a node set	nodes within distance k from x or within distance k from y		$\surd$(2)
2	$s^{(k)}$	a node set	a list of values	1 if connected within distance k, 0 otherwise	$\surd$ (1)	$\surd$(1,2)
	t	a node set	a list of values	distance between a pair of nodes	$\surd$ (1)	$\surd$(1,2)
	t_x	a node set	a list of values	distance between node x and other nodes	$\surd$ (2)	$\surd$(1,2)
	γ	a node set	a list of values	number of links in each node		$\surd$(2)
	u_x	a node set	a list of values	1 if the shortest path includes x, 0 otherwise	$\surd$ (2)	$\surd$(1,2)
	e_x	a node set	a list of values	structural equivalence between node x and other nodes	$\surd$ (2)	
3	Avg	a list of values	a value	average of values	$\surd$ (1)	$\surd$(1,2)
	Sum	a list of values	a value	summation of values	$\surd$ (1)	$\surd$(1,2)
	Min	a list of values	a value	minimum of values	$\surd$ (1)	$\surd$(1,2)
	Max	a list of values	a value	maximum of values	$\surd$ (1)	$\surd$(1,2)
4	$Diff$	two values	value	difference of two values		$\surd$(1,2)
	Avg	two values	value	average of two values		$\surd$(1,2)
	$Product$	two values	value	product of two values		$\surd$(1,2)
	$Ratio$	two values	value	ratio of two values	$\surd$ (4)	$\surd$(1,2)
	Max	two values	value	maximum of two values		$\surd$(1,2)
	Min	two values	value	minimum of two values		$\surd$(1,2)

- *: LC stands for link-based classification, and LP stands for link prediction. The number in the parentheses is Method number.
- Aggregate operators in Step 4 is optional. This aggregates two features values obtained in Step 3 into one single feature value.

as $Sum \circ u_x \circ C_x^{(\infty)}$. It represents the average of edge existence among all nodes; it therefore corresponds to the network density. The combinations of operators can realize the network-based features.

In addition to commonly used network-based features, we can generate several features that have been shown to be effective in existing studies (Backstrom *et al.* 2006).

As for link prediction, we can also generate several features that are often used in the literature: common neighbors is realized by $Ratio\{Sum \circ t_{xy} \circ (C_x^{(1)} \cap C_y^{(1)}), Sum \circ t_{xy} \circ (C_x^{(1)} \cup C_y^{(1)})\}$, Jaccard coefficient is by $Ratio\{Sum \circ t_{xy} \circ (C_x^{(1)} \cap C_y^{(1)}), Sum \circ t_{xy} \circ (C_x^{(1)} \cup C_y^{(1)})\}$, and Adamic/Adar is written as $Sum \circ \gamma \circ (C_x^{(1)} \cap C_y^{(1)})$.

In summary, using our feature generation mechanism, several features are obtainable including traditional network features and newly discovered useful features.

Experimental Result

We evaluate our algorithm with two datasets: the @cosme dataset and Hatena Bookmark dataset. We apply our algorithm to these datasets and thereby generate network features for each node. After generating features, we investigate which features are better to classify the entities. We use a decision-tree technique following (Backstrom *et al.* 2006) to generate the decision tree (using the C4.5 algorithm (Quinlan 1993)).

@cosme dataset @cosme (www.cosme.net) is the largest online community site of "for-women" communities in Japan. It provides information and reviews related to cosmetic products. And a user can register other users who can be trusted, thereby creating a social network of users in which the node is a user and a link is a trusted relation between users.

For link-based classification, training and test data are created as follows. We first choose a community as a target. Then, we randomly select users in the community as positive examples. As negative examples, we select those who are not in the community but who have friends who are in the target community. Therefore, our problem setting is difficult because even a negative example has some relation to the positive class. The target communities are selected from popular communities with more than 1000 members[2]. The negative examples are the nodes which are not in the category but which have a direct relation with other nodes in the community. Therefore, the settings are more difficult than those used when we select negative examples randomly.

For the link prediction task, we use two methods, Method 1 and Method 2. The training and test data are made by selecting links between the time T and time T''. The positive examples are randomly picked up among links created between time T and T' ($T < T' < T''$). The negative examples are those created between time T' and T''. Therefore, the setting is more difficult than to predict link existence between two randomly picked-up nodes without considering temporal order. After we generate the network features, we use the C4.5 algorithm to determine whether a link will be formed between two nodes.

Hatena Bookmark dataset Hatena (www.hatena.ne.jp) is a large-scale community site in Japan run by Hatena Co. Ltd., integrated with blog hosting, knowledge sharing, photo sharing and so on. *Hatena bookmark* is a collaborative tagging similar to *del.icio.us*, where each user can make bookmarks of web pages, and post their comment.Users can apply tags to each URL, and each bookmark is connected through these tags. A bookmarking datum consists of a user name, resource (URL), tags, and a date.

Although no explicit social network exists in the dataset, we create a social network as follows: we first define similarity between users considering the overlap of those URLs to which the users annotate tags. This is a common approach

[2]Such as the *I love Skin Care* community, the *Blue Base* community, and the *I love LUSH* community. For *I love Skin Care* community, there are 2807 positive nodes and 2923 negative nodes.

Table 4: Recall, precision, and F-value as adding operators.

	(a) @cosme			(b) Hatena Bookmark		
	Recall	Precision	F-val.	Recall	Precision	F-val.
baseline	0.43	0.600	0.495	0.628	0.704	0.661
Method 1	0.387	0.593	0.465	0.499	0.726	0.581
Method 2	0.432	0.581	0.491	0.509	0.720	0.585
Method 3	0.499	0.574	0.532	0.673	0.707	0.681
Method 4	0.604	0.607	0.604	0.692	0.758	0.717

to analyze the relation of tags, users, and resources (Mika 2007). Similarity between two nodes x and y is defined as $LinkWeight(x,y) = \frac{|U(x) \cap U(y)|}{|U(x)| \cdot |U(y)|}$, where $U(x)$ is the set of URLs user x has marked and the $|U(x)|$ is the number of URLs to which user x makes tags. In our experiment, we set the threshold as 0.00002, so that the average degree of each node becomes approximately equal to that in @cosme dataset.

Our task is to predict whether or not a user will use a particular tag, in other words, tag prediction using a social network. We create the training and test data similarly for the @cosme dataset. We select the 10 target tags randomly from tags. Examples of the tags we use are *software*, *game*, *book*, *movie*, and *music*. For the *software* tag, there are 1203 positive nodes and 1195 negative nodes.

Results: Link-based Classification

We generate the features defined in Table 3 for each dataset. To clarify the effectiveness of operators, we first limit the operators to Method 1, as shown in Table 3; then we incrementally add the operators of Method 2, Method 3, and Method 4. For that reason, more features are generated as the method increases; in light of the difference of performance, we can see the usefulness of operators on each method.

The result for the @cosme dataset is presented in Table 4(a). Performance is improved if we use more operators. For comparison, we make a baseline classifier which uses features used in the study by Backstromet al. (Backstrom *et al.* 2006) (as shown in Table 1). Our proposed method yielded better performance when we included the operators up to Method 3 or Method 4. Especially, if we use full features, the F-value becomes as high as 0.6, which is much larger than the baseline.

The result for the Hatena Bookmark dataset is shown in Table 4(b). The general tendency resembles that for the @cosme dataset; however, because we use more operators, the performance is also improved. The proposed method is better when we include the operator up to Method 3 or Method 4.

Detailed Analyses of Useful Features We calculate a score for each feature after obtaining the decision tree to discover useful features among those generated. We add the score $1/r$ to the feature if it appears in the r-th level of the decision tree; then we sum up the scores in all the case.Table 5 shows the effective features (which appear often in the obtained decision trees) in the @cosme dataset. In summary, various features are demonstrably important for classification, some of which correspond to well-known indices in social network analysis such as the degree of node in the second, betweenness centrality in the fifth, and characteristic path length in the third. The top feature is the number of links among positive nodes that are reachable from the node, which corresponds to the network density. This means that the denser the network density among the nodes, the more likely it is that the node will join the community. The feature in the seventh is the same as the feature used in (Backstrom *et al.* 2006). This means that if the adjacent nodes are mutually connected, the node will join the community.

Some features seem to be new indices, but their meanings resemble those of the existing indices. Nevertheless, the ratio of values on positive nodes to all nodes is useful in many cases. We can find structural equivalence in the 9th and 10th, which means that if the network structures of nodes among the target node are important. The results support the usefulness of the indices that are commonly used in the social network literature, and illustrate the potential for further composition of useful features.

Table 5: Top 10 effective features in the @cosme dataset for link-based classification.

Feature	Description
$Sum \circ t \circ (C_x^{(\infty)} \cap N_{C=c})$	Number of links among nodes reachable from x and attribute C is c.
$Sum \circ s^{(1)} \circ C_x^{(1)}$	Number of links among nodes adjacent to x.
$Avg \circ t \circ C_x^{(\infty)}$	Characteristic path length of nodes reachable from x.
$Avg \circ t \circ (C_x^{(\infty)} \cap N_{C=c})$	Characteristic path length of nodes reachable from x and attribute C is c.
$Sum \circ u_x \circ (C_x^{(\infty)} \cap N_{C=c})$	Betweenness centrality of nodes reachable from x and attribute C is c.
$Sum \circ t_x \circ C_x^{(1)}$	Number of nodes adjacent to x.
$Sum \circ s^{(1)} \circ (C_x^{(1)} \cap N_{C=c})$	Number of links among positive nodes adjacent to node x.
$Avg \circ u_x \circ C_x^{(1)}$	Betweenness centrality of nodes adjacent to x.
$Max \circ e_x \circ C_x^{(\infty)}$	Maximum of the structural equivalent of nodes reachable from x.
$Sum \circ e_x \circ (C_x^{(\infty)} \cap N_{C=c})$	Summation of the structural equivalent of nodes reachable from x and attribute C is c.

Table 6: Recall, precision, and F-value in the @cosme dataset as adding operators.

	Recall	Precision	F-value
graphic distance	0.1704	0.6687	0.2708
common neighbors	0.1704	0.6687	0.2708
Jaccard's coefficient	0.1396	0.7031	0.2326
Adamic/Adar	0.1704	0.6686	0.2708
preferential attachment	0.5553	0.5779	0.5658
Method 1	0.5772	0.6333	0.5982
Method 2	**0.5687**	**0.6721**	**0.613**

Results: Link Prediction

For the link prediction problem, we use Method 1 and Method 2 to generate network-based features. For comparison, we use other methods, as listed in Table 2 because these methods are often used in the link-prediction literature.

We calculate the recall, precision and F-value using 10-fold cross validation, as presented in Table 6. The F-values obtained by Method 1 and Method 2 are better than those for other methods. Graphic distance, common neighbors and Adamic/Adar produce poor results. This poor performance might be because the common neighbors between two node is almost zero in our dataset; in other words, rarely is a user one who has mutual acquaintances between two randomly picked up users. In our algorithm, the performance is good because of the numerous common nodes within a distance of 2 or more.

Table 7: Top 10 effective features in the @cosme dataset for link prediction (Method 1)

Feature	Description
$Max\{Avg \circ t \circ C_x^{(2)}, Avg \circ t \circ C_y^{(2)}\}$	Maximum of the average distance.
$Max\{Sum \circ s^{(1)} \circ C_x^{(1)}, Sum \circ s^{(1)} \circ C_y^{(1)}\}$	Maximum of the clustering coefficient.
$Min\{Sum \circ t_x \circ C_x^{(1)}, Sum \circ t_x \circ C_y^{(1)}\}$	Minimum of the number of adjacent nodes.
$Max\{Avg \circ s^{(1)} \circ C_x^{(2)}, Avg \circ s^{(1)} \circ C_x^{(2)}\}$	Minimum of the network density.
$Max\{Avg \circ u_x \circ C_x^{(2)}, Avg \circ u_x \circ C_y^{(2)}\}$	Maximum of the betweenness centrality.
$Min\{Avg \circ t \circ C_x^{(2)}, Avg \circ t \circ C_y^{(2)}\}$	Minimum of the average path length.
$Max\{Sum \circ u_x \circ C_x^{(2)}, Sum \circ u_x \circ C_y^{(2)}\}$	Maximum of the betweenness centrality.
$Max\{Sum \circ t_x \circ C_x^{(1)}, Sum \circ t_x \circ C_y^{(1)}\}$	Maximum of the number of adjacent nodes.
$Avg\{Sum \circ s^{(1)} \circ C_x^{(1)}, Sum \circ s^{(1)} \circ C_y^{(1)}\}$	Average of the clustering coefficient.
$Sum \circ u_x \circ C_x^{(2)} - Sum \circ u_x \circ C_y^{(2)}$	Difference of the betweenness centrality.

We also examine which features are effective for link prediction. Table 7 portrays the top-ten effective features for link prediction using Method 1. In Method 1, we can see average path length in the first, node clustering in the second and degree of node in the third, which are often used in social network analyses. The notable characteristic of this result is that all features are the maximum of minimum of the indices of two nodes, which means that whether the node connects to the other nodes or not depends mainly on the index of one node, rather than the pairs of nodes.

In Method 2, the top-three combinations are $Min \circ \gamma \circ (C_x^{(2)} \cup C_y^{(2)})$, $Max \circ \gamma \circ (C_x^{(2)} \cup C_y^{(2)})$, and $Avg \circ \gamma \circ (C_x^{(2)} \cup C_y^{(2)})$. As a result, top-eight features include the operator γ in Step 2, which shows that the feature by Adamic/Adar is often a good feature for predicting links.

Based on these two results, we can infer that our proposed method is effective in various cases. As presented above, the feature of a larger node set (a node set with distance k from node x) sometimes improves the accuracy more than the feature adjacent to node x. That fact suggests the potential importance for aggregating features on a wider range for link mining.

Discussion

We have defined the operators while considering a trade-off: keeping operators simple and covering various indices. Other features cannot be composed in our current setting. Eigenvector centrality is a difficult index to implement using operators because it requires iterative processing (or matrix processing). We do not argue that the operators that we define are optimal or better than any other set of operators. Elaborate analysis of possible operators is an important future task.

Because we add more and more operators, the number of features we can generate becomes huge. For example, Method 2 in link prediction produces 160 features in all. Therefore, in the training process, the decision tree might be overfitted. We can use information gain and other techniques for feature selection.

Future studies will compare the performance with other existing algorithms for link-based classification, i.e., *approximate collective classification algorithms* (ACCA) (Sen & Getoor 2007). Our algorithm falls into the family of models proposed in Inductive Logic Programming (ILP) called *propositionalization* and *upgrade*.

Conclusion

In this paper, we proposed an algorithm to generate network-based features using a social network data. Our algorithm can generate features which are well studied in social network analysis, and some useful new features in a systematic fashion. We applied our proposed method to two datasets for link-based classification and link prediction tasks and thereby demonstrated that some features are useful for predicting user interactions. We found empirically that commonly used indices such as centrality measures and characteristic path length are useful ones among all possible indices used in social networking analysis. In the link prediction tasks, the operators from Adamic/Adar were also shown to be useful.

As a future study, we seek to build a prototypical system to demonstrate the usefulness of our network-based features such as recommendation of friends, or recommendation of products.

References

Adamic, L., and Glance, N. 2005. The political blogosphere and the 2004 u.s. election: Divided they blog. In *LinkKDD-2005*.

Ahn, Y.; Han, S.; Kwak, H.; Moon, S.; and Jeong, H. 2007. Analysis of topological characteristics of huge online social networking services. In *Proc. WWW2007*.

Backstrom, L.; Huttenlocher, D.; Lan, X.; and Kleinberg, J. 2006. Group formation in large social networks: Membership, growth, and evolution. In *Proc. SIGKDD'06*.

Freeman, L. C. 1979. Centrality in social networks: Conceptual clarification. *Social Networks* 1:215–239.

Furukawa, T.; Matsuo, Y.; Ohmukai, I.; Uchiyama, K.; and Ishizuka, M. 2007. Social networks and reading behavior in the blogosphere. In *Proc. ICWSM 2007*.

Getoor, L., and Diehl, C. P. 2005. Link mining: A survey. *SIGKDD Explorations* 2(7).

Golder, S., and Huberman, B. A. 2006. The structure of collaborative tagging systems. *Journal of Information Science*.

Liben-Nowell, D., and Kleinberg, J. 2003. The link prediction problem for social networks. In *Proc. CIKM*, 556–559.

Mika, P. 2005. Ontologies are us: A unified model of social networks and semantics. In *Proc. ISWC2005*.

Mika, P. 2007. Web semantics: Science services and agents on the world wide web. 5–15.

Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning*. California: Morgan Kaufmann.

Scott, J. 2000. *Social Network Analysis: A Handbook (2nd ed.)*. SAGE publications.

Sen, P., and Getoor, L. 2007. Link-based classification. Technical Report CS-TR-4858, University of Maryland.

Staab, S.; Domingos, P.; Mika, P.; Golbeck, J.; Ding, L.; Finin, T.; Joshi, A.; Nowak, A.; and Vallacher, R. 2005. Social networks applied. *IEEE Intelligent Systems* 80–93.

Wasserman, S., and Faust, K. 1994. *Social network analysis. Methods and Applications*. Cambridge: Cambridge University Press.