Knowl Inf Syst (2012) 32:141–166
DOI 10.1007/s10115-011-0427-z

REGULAR PAPER

MatchSim: a novel similarity measure based on maximum neighborhood matching

Zhenjiang Lin · Michael R. Lyu · Irwin King

Received: 25 January 2010 / Revised: 1 April 2011 / Accepted: 27 May 2011 /
Published online: 14 June 2011

Abstract Measuring object similarity in a graph is a fundamental data- mining problem in various application domains, including Web linkage mining, social network analysis, information retrieval, and recommender systems. In this paper, we focus on the neighbor-based approach that is based on the intuition that "similar objects have similar neighbors" and propose a novel similarity measure called *MatchSim*. Our method recursively defines the similarity between two objects by the average similarity of the maximum-matched similar neighbor pairs between them. We show that MatchSim conforms to the basic intuition of similarity; therefore, it can overcome the counterintuitive contradiction in SimRank. Moreover, MatchSim can be viewed as an extension of the traditional neighbor-counting scheme by taking the similarities between neighbors into account, leading to higher flexibility. We present the MatchSim score computation process and prove its convergence. We also analyze its time and space complexity and suggest two accelerating techniques: (1) proposing a simple *pruning strategy* and (2) adopting an *approximation algorithm* for maximum matching computation. Experimental results on real-world datasets show that although our method is less efficient computationally, it outperforms classic methods in terms of accuracy.

Keywords Link-based similarity measure · Link analysis · Web mining · Maximum matching

Z. Lin (✉) · M. R. Lyu · I. King
Department of Computer Science and Engineering,
The Chinese University of Hong Kong, Shatin, NT, Hong Kong
e-mail: zjlin@cse.cuhk.edu.hk; linallen5460@gmail.com

M. R. Lyu
e-mail: lyu@cse.cuhk.edu.hk

I. King
e-mail: king@cse.cuhk.edu.hk

1 Introduction

Measuring similarity between *objects* (web pages, persons, academic articles, movies, etc.) is required by many applications in different domains, such as Web mining, social networks, citation analysis, information retrieval, and recommender systems. For example, on the Web, besides traditional *keyword-based* search, some search engines also support *instance-based* search: searching *similar* or *related* web pages to a given one.[1] In automated submission web directories, human editing is replaced with automated processing, which requires special techniques for website content categorization and filtering.[2] Identifying online social communities that consist of highly related web pages or individuals is another important application [7]. In this paper, we use the Web as an example to illustrate our ideas. Nevertheless, the proposed method is also applicable to any data sources with a graph structure.

Because various properties of objects can be exploited to estimate similarity, accordingly similarity measures are usually grouped into different classes. For example, for web pages, there are two basic approaches, *text-based* and *link-based*. Originated from information retrieval, the *text-based* methods exploit textual content of web pages. One commonly used method is *cosine TFIDF* [2,30]. New methods have been proposed in the past few years [1,37,40]. The major deficiency of the text-based methods is that they usually require large storage and long computing time, due to the need for full-text comparison. Secondly, the large amounts of web pages with low-quality or even malicious textual contents may reduce their accuracy [13]. The *link-based* methods exploit relationships between objects (e.g., hyperlinks between web pages). They can be further grouped into 1) the *path-based* methods, such as the Maximum Flow/Minmum Cut [26] and the Katz measure [16], which "refine the notion of shortest-path distance by implicitly considering the ensemble of all paths between two pages [22]," and 2) the *neighbor-based* methods, which share a simple intuition that "similar objects have similar neighbors," such as Co-citation [33], Bibliographic coupling [17], Jaccard Measure [14], and SimRank [15].

In this paper, we focus on the neighbor-based approach. Traditional neighbor-counting methods measure overlaps and/or differences between objects' neighbor sets. For example, Co-citation and Bibliographic coupling work by counting the numbers of common *inlink* and *outlink* neighbors, respectively. Jaccard Measure defines similarity between objects by the size of the *intersection* divided by the size of the *union* of their neighbor sets. These methods run fast and are easy to implement. But they lack flexibility because of ignoring similarities between neighbors. SimRank makes an extension by taking neighbors' similarities into account. However, it has a counterintuitive contradiction [8], which may influence its accuracy as a result. We give examples in Sect. 4.1 to demonstrate the problems of these methods.

We consequently propose a novel similarity measure called *MatchSim* in this paper, which overcomes the above problems of classic neighbor-based methods by (1) taking similarities between neighbors into account, and (2) conforming to the basic intuition of similarity. Therefore, potentially our method can produce better results. MatchSim recursively defines the similarity between two objects by the average similarity of the maximum-matched similar neighbor pairs between them. More precisely, to calculate the similarity score $sim(a, b)$ between two objects a and b, MatchSim first finds out a *maximum matching* between their similar neighbors. (If the numbers of two pages' neighbors are not the same, we simply add *dummy* neighbors that are similar to none of the others to make up the missing part.) Next,

[1] http://www.googleguide.com/similar_pages.html.

[2] http://en.wikipedia.org/wiki/Web_directory.

$sim(a, b)$ is replaced with the average similarity of the maximum-matched neighbor pairs. The process of MatchSim score computation is iterative and can be proved to converge under certain conditions. The main contributions of the paper are summarized as follows:

1. Proposing *MatchSim*, which measures similarity between any networked objects based on maximum neighborhood matching.
2. Suggesting accelerating techniques to improve efficiency of MatchSim, including a *pruning strategy* and an *approximation algorithm*.
3. Conducting extensive experiments on real-world datasets to evaluate the performance of MatchSim, as well as the accelerating techniques.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 introduces the basic notations and concepts used in the paper. Section 4 presents the MatchSim algorithm, including its key ideas, mathematical definition, iterative computing process, complexity analysis, and suggested accelerating techniques. Section 5 reports the experimental results and discussions. Conclusion and future work are given in Sect. 6. Appendix *A* presents the proof of MatchSim score computation convergence, Appendix *B* gives a brief description on the maximum matching problem, and Appendix *C* presents some basic statistics on the link structure of the datasets.

2 Related work

Similarity measures are central to many important applications such as searching, clustering, classification, and recommendation. Many of them have been developed for different kinds of data sources including text [20,37], image [10,11], sequence [21], and geographic data [9]. In [4], the similarity strategies are organized into the following four categories: (1) direct mechanisms, (2) transformation-based mechanisms, (3) information-theoretic measures, and (4) emergent measures arising from an in-depth analysis of the data.

Similarity measures are associated with the representation of underlying data sources. For example, the direct mechanisms are applicable to *feature vector representation*. Many of the traditional methods belong to this category, including *Minkowski distance*, *Euclidean distance*, *Manhattan distance*, and *cosine similarity*, as well as the *set*-based distances including *Dice similarity*, *Amsler*, *Jaccard measure*, and *Overlap similarity*, which was recently applied to multivariate data source [38,39]. We refer readers to [4] which contains a comprehensive description of similarity measures and data representations.

Link structure has been proven to be a useful source of data for extracting knowledge in the areas including citation analysis, Web mining, and social network analysis. Many *link-based* similarity measures have been developed, which can be classified into *path based* and *neighbor based*. The *path-based* methods take the global structure of graph into consideration. These methods include *Maximum Flow/Minmum Cut* [26] and Katz measure [16], which originate from graph theory, *Companion* [5], which is derived from HITS algorithm [18], and *PageSim* [24], which is based on propagation of objects' features, etc.

The *neighbor-based* methods focus on local neighborhood comparison between objects in a graph. Many of them originated from traditional domains such as IR, set theory, and citation analysis. The *Co-citation* [33] algorithm was first introduced by Small in the fields of citation analysis and bibliometrics as a fundamental metric to characterize the similarity between scientific papers. Two papers a and b are *co-cited* if they are cited by a third paper c.

Table 1 Classical neighbor-based similarity measures

Bibliographic coupling	$\|O(a) \cap O(b)\|$
Co-citation	$\|I(a) \cap I(b)\|$
Jaccard measure	$\frac{\|\Gamma(a) \cap \Gamma(b)\|}{\|\Gamma(a) \cup \Gamma(b)\|}$
SimRank	$\gamma \cdot \frac{\sum_{u \in I(a)} \sum_{v \in I(b)} sim(u,v)}{\|I(a)\|\|I(b)\|}, \gamma \in (0,1)$
Dice coefficient	$\frac{2\|\Gamma(a) \cap \Gamma(b)\|}{\|\Gamma(a)\| + \|\Gamma(b)\|}$
Overlap similarity	$\frac{\|\Gamma(a) \cap \Gamma(b)\|}{min(\|\Gamma(a)\|, \|\Gamma(b)\|)}$

The more papers they are cited by, the stronger their relationship is. The similarity between two papers a and b is defined by $sim(a, b) = |I(a) \cap I(b)|$, where $I(a)$ refers to the set of *inlinks* of a, i.e., the papers citing a.

The *Bibliographic Coupling* [17] was proposed by Kessler to measure paper similarities. Two papers have a unit of bibliographic coupling if both cite a same paper. The idea is based on the observation that paper authors work on the same subject tend to cite the same papers. By definition, $sim(a, b) = |O(a) \cap O(b)|$, where $O(a)$ refers to the set of *outlinks* of a, i.e., the papers cited by a.

The *Jaccard Measure* [14], also known as the *Jaccard's Coefficient*, is a standard measure in information retrieval. It is defined as the size of the intersection divided by the size of the union of two sets. Given two objects a and b, let $\Gamma(a)$ and $\Gamma(b)$ denote their respective neighbor sets (either inlinks or outlinks), then the similarity is defined by $sim(a, b) = \frac{|\Gamma(a) \cap \Gamma(b)|}{|\Gamma(a) \cup \Gamma(b)|}$.

The neighbor-counting methods ignore similarities between neighbors. It may reduce their performance. The situation is even worse for the Web, which is extremely sparse. The Web contains billions of web pages, most of which have only tens or hundreds of (inlink and outlink) neighbors. Therefore, the chance that two web pages happen to share common neighbors is very slim. Thus, how to make good use of the relatively tiny-sized neighborhoods is one of the challenges for the neighbor-based methods.

The *SimRank* [15] algorithm relaxes the "neighbor-counting" strategy by taking similarities between neighbors into account. It is based on the intuition that "two objects are similar if they are referenced by similar objects [15]". More precisely, the SimRank score $sim(a, b)$ between pages a and b is recursively defined as the average similarity of all possible neighbor pairs between them times a decay factor. It has been criticized in [8] that in some cases, SimRank outputs *counterintuitive* results. We present the SimRank definition in Table 1 and will give more detailed description and discussion in Sects. 3 and 4.

Many of the traditional set-based methods can be easily converted into neighbor-based, such as *Dice Coefficient* [36] and *Overlap Similarity* (a general version of *Co-citation* and *Bibliographic Coupling*). We summarize the definitions of commonly used neighbor-based similarity measures in Table 1 and use four of the most classical methods in the examples and experiments of the paper to compete with MatchSim method. Interested readers are refereed to [22], which contains an exhaustive list of link-based similarity measures.

The practical performance of neighbor-based methods relies on the specific nature of the data they use. For example, in our experiments, Bibliographic Coupling performs much poorer on scientific articles than it does on web pages. The methods originated from set theory, such as Jaccard Measure, can use either inlink or outlink neighbors for measurement, but may produce very different results. Therefore, how to choose the "right" properties of data for similarity measurement is one of the most important practical concerns. More discussions will be given in Sect. 5.

3 Preliminaries

3.1 Basic notations

The **Web graph** is a directed graph $G = (V, E)$ with vertices V representing web pages $v_i (i = 1, 2, \ldots, n)$ and directed edges E representing hyperlinks among web pages. We denote by $I(v)$ and $O(v)$ the sets of *inlink* and *outlink* neighbors of page v, respectively. Similarly, in **citation graph**, vertices represent papers and directed edges represent citations from one paper to another. These are two types of graphs that are used as examples throughout the paper. We denote similarity score between objects a and b by notation $sim(a, b)$, the value of which depends on the similarity function used in context.

3.2 Similarity

Similarity is a fundamental concept in almost all research domains. In philosophy, *similarity* is regarded as the relation of sharing *properties* (or *features*) between two *objects*. According to *bundle theory*, an object consists of its properties and nothing more. Therefore, the similarity between objects is essentially the similarity between the collections of their properties.

In [23], the authors proposed that similarity obeys the following three basic *intuitions*, which are good guidelines for designing consistent similarity measures.

S1: the more commonality two objects share, the more similar they are;
S2: the more differences two objects have, the less similar they are;
S3: the maximum similarity is reached when two objects are *identical*, no matter how much commonality they share.

3.3 SimRank algorithm

SimRank is an iterative neighbor-based similarity measure. Numerically, for objects a and b in a graph, this is specified by defining $sim(a, b) = 1$ for $a = b$ and

$$sim(a, b) = \gamma \cdot \frac{\Sigma_{u \in I(a)} \Sigma_{v \in I(b)} sim(u, v)}{|I(a)||I(b)|} \tag{1}$$

for $a \neq b$, where $\gamma \in (0, 1)$ is a constant. If $I(a)$ or $I(b)$ is empty, then $sim(a, b)$ is zero by definition. The SimRank iteration starts with $sim(a, b) = 1$ for $a = b$ and $sim(a, b) = 0$ for $a \neq b$. The SimRank score between a and b is defined as the fixed point of Eq. (1). $I(a)$ and $I(b)$ can also be replaced by $O(a)$ and $O(b)$ when SimRank uses outlinks instead.

From Eq. (1), we can see that $\Sigma_{u \in I(a)} \Sigma_{v \in I(b)} sim(u, v)$ is the sum of similarity over all neighbor pairs between $I(a)$ and $I(b)$. $|I(a)||I(b)|$ is the number of all possible neighbor pairs. Therefore, $sim(a, b)$ is the product of constant γ times the average similarity over all possible neighbor pairs between $I(a)$ and $I(b)$.

4 MatchSim algorithm

4.1 What inspired MatchSim?

MatchSim is inspired by two major drawbacks of classic neighbor-based similarity measures. One is that the neighbor-counting methods ignore similarities between neighbors. The other

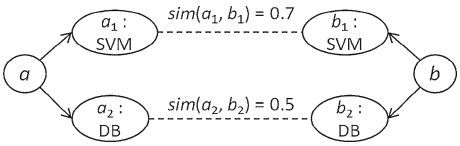

Fig. 1 Objects a and b have similar neighbors

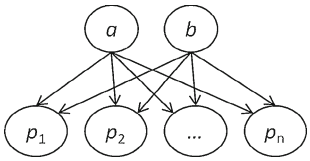

Fig. 2 Objects a and b have n common neighbors

is that the SimRank method violates intuition (S1) of similarity in some cases. By overcoming these drawbacks, potentially our method can produce more accurate results. In the following examples, we illustrate the problems and introduce the basic idea of MatchSim. In next subsection, we give MatchSim's formal definition, and in Sect. 4.2.1, we explain its advantages by showing how it solves the problems.

Example 1 Figure 1 presents a snippet of citation graph in which a_1 and b_1 are scientific papers about SVM (Support Vector Machine), and a_2 and b_2 are about DB (Database). Assume it is known that $sim(a_1, b_1) = 0.7$, $sim(a_2, b_2) = 0.5$, and $sim(a_1, b_2) = sim(a_2, b_1) = 0$.[3]

Neighbor-counting methods will conclude that a and b are not similar at all (i.e., $sim(a, b) = 0$) because they have no common neighbors, which is clearly *inaccurate*. SimRank takes the similarities between neighbors into account. More precisely, in SimRank,

$$sim(a, b) = \gamma \cdot \sum_{i=1,2} \sum_{j=1,2} sim(a_i, b_j)/4 = 0.3\gamma > 0,$$

which makes more sense. But if we remove the most similar neighbor pairs (a_1, b_1), $sim(a, b)$ will increase to $\gamma \times sim(a_2, b_2)/1 = 0.5\gamma$, which is evidently *counterintuitive*.

Example 2 Here, we reveal the drawback of SimRank with an extreme case. It has been criticized in [8] that when objects a and b have exactly $n (n > 0)$ common neighbors, and the SimRank score between any distinct neighbors is 0, then $sim(a, b)$ approaches to 0 as n increases (see Fig. 2). This means that in this case, the more common neighbors that a and b have, the less similar they are. Clearly, in this case, SimRank violates intuition (S1) of similarity.

The problem of SimRank is caused by its strategy of "considering the overall sum of similarities between neighbors." Intuitively, in Example 1, we say that a and b are similar simply based on the fact that they have *pairwise similar* neighbors (a_1, b_1) and (a_2, b_2). Therefore, we can use *the average similarity of the pairwise similar neighbors* as the measurement of $sim(a, b)$. This is the key idea of MatchSim.

[3] The assumption is reasonable since papers with the same topic should be more similar as they are prone to citing more common references.

4.2 MatchSim definition

Given two distinct objects a and b in a graph of size n, we obtain a weighted bipartite graph $G_{a,b} = (I(a), I(b), E, w)$, where $E = \{(u, v)|u \in I(a), v \in I(b)\}$ and $w(u, v) = sim(u, v)$. The MatchSim score is defined by

$$sim(a, b) = \frac{\widehat{W}(a, b)}{max(|I(a)|, |I(b)|)}. \tag{2}$$

In the cases that $|I(a)| = 0$ or $|I(b)| = 0$, since there is no way to infer any similarity, we define $sim(a, b) = 0$. When $a = b$, we define $sim(a, b) = 1$.

In Eq. (2), $\widehat{W}(a, b)$ denotes the weight of a maximum matching between $I(a)$ and $I(b)$, i.e.,

$$\widehat{W}(a, b) = W(m^*_{ab}) = \sum_{(u,v)\in m^*_{ab}} sim(u, v), \tag{3}$$

where m^*_{ab} is a maximum matching between $I(a)$ and $I(b)$. Because we always convert $I(a)$ and $I(b)$ to be "equally sized" before computing m^*_{ab}, we just define $l_{ab} = |m^*_{ab}| = max(I(a), I(b))$. Since any matching between $I(a)$ and $I(b)$ is of size l_{ab}, the factor $\frac{\widehat{W}(a,b)}{max(I(a),I(b))}$ in Eq. (2) is actually the average similarity of the maximum matching between a's and b's neighbors.

Each (ordered) pair of pages a and b corresponds to one equation of the form in Eq. (2), resulting in a set of n^2 MatchSim equations. The n^2 MatchSim scores are defined by the (n^2-dimensional) *fixed point* of the equations, which can be reached by iterative computation. The details of MatchSim iteration will be given in Sect. 4.3.

Finding the maximum-matched similar neighbor pairs is actually the well-known *maximum matching* or *assignment* problem [3] and can be solved by K-M (Kuhn-Munkres) algorithm [19] in polynomial time. We give a brief introduction on the assignment problem in Appendix *B*. Interested readers are referred to [12] for a comprehensive overview on this topic.

4.2.1 Discussions on MatchSim

First, recall the examples in Sect. 4.1, we now show that MatchSim can successfully overcome the drawbacks of the classic methods. (Note that we use outlink neighbors as input in the examples.) 1) Compared to the neighbor-counting methods, MatchSim takes the similarities between neighbors into account. Therefore, it can measure that in Example 1, at least $sim(a, b)$ is nonzero. 2) In Example 1, (a_1, b_1) and (a_2, b_2) are the maximum matching between $O(a)$ and $O(b)$, thus MatchSim calculates $sim(a, b) = (sim(a_1, b_1) + sim(a_2, b_2))/2 = 0.6$. If we remove (a_1, b_1), $sim(a, b) = sim(a_2, b_2)/1$ drops to 0.5, which makes more sense than SimRank. In Example 2, obviously $(p_i, p_i)(i = 1, \ldots, n)$ are the maximum matching. Therefore, we have $sim(a, b) = \sum_{i=1,n} sim(p_i, p_i)/n = n/n = 1$.

Particularly, MatchSim confirms to the intuition of similarity. In Example 2, MatchSim outputs that

$$sim(a, b) = \frac{n}{n} = 1 = max_{a,b\in V} sim(a, b),$$

which means MatchSim conforms to intuition (S3) of similarity. It is easy to see that MatchSim also conforms to intuitions (S1) and (S2). By this way, we can ensure that MatchSim will not produce "unreasonable" results.

Second, by considering similarities between neighbors, MatchSim can be viewed as an extension of Jaccard Measure. In Eq. (2), $\widehat{W}(a, b)$ is the "overlap" of similarity between the maximum-matched neighbors, and $max(I(a), I(b))$ is the volume of union between the maximum-matched neighbors.

Third, either inlink or outlink neighbors can be used in MatchSim, but may result in very different accuracy. Actually, choosing the "right" type of neighbors as input is very important to any neighbor-based similarity measures. Generally speaking, the more neighbors two objects have, the more accurately can we measure their similarity. This conjecture is supported by the experimental results in Sect. 5 where we will give more discussions on this issue.

Last, we list some other properties of MatchSim as follows, which are easy to deduce from its definition.

1. It is *symmetric*: $sim(a, b) = sim(b, a)$;
2. It is *bounded*: $sim(a, b) \in [0, 1]$;
3. It reaches a *maximum* value of 1, if and only if a and b are *identical*, i.e., $sim(a, b) = 1 \Leftrightarrow a = b$ or, $a \neq b$ and $I(a) = I(b) \neq \emptyset$.

4.3 MatchSim computation

For a graph G of size n, we compute the n^2 MatchSim scores iteratively. For each iteration k, we can keep the n^2 scores $sim_k(*, *)$, where $sim_k(a, b)$ is the score between a and b in iteration k. We successively compute $sim_{k+1}(*, *)$ based on $sim_k(*, *)$. That is, on each iteration $k + 1$, we update the $sim_{k+1}(a, b)$ using the similarity scores from the precious iteration k. Formally speaking, we compute $sim_{k+1}(a, b)$ from $sim_k(*, *)$ as follows:

$$sim_{k+1}(a, b) = \frac{\widehat{W}_k(a, b)}{max(|I(a)|, |I(b)|)}, \tag{4}$$

where $\widehat{W}_k(a, b)$ is computed based on the scores $sim_k(*, *)$.

The MatchSim computation starts with $sim_0(a, b) = 1$ for $a = b$ and $sim_0(a, b) = 0$ for $a \neq b$. The MatchSim score between a and b is defined as $lim_{k\to\infty} sim_k(a, b)$. We proved that with the given initial values, the limiting values exist and are unique, i.e., the MatchSim iteration converges. The detailed proof of convergence is given in Appendix *A*. In all of our experiments, MatchSim converges after about 15 iterations, so we may choose to fix a number $K = 15$ of iterations to perform.

4.4 Complexity analysis

Time complexity: For any two objects a and b in a graph $G = (V, E)$ of size n, we adopt K-M algorithm to compute $\widehat{W}(a, b)$ in Eq. (2), so the corresponding time complexity is l_{ab}^3, where $l_{ab} = |m_{ab}^*| = max(|I(a)|, |I(b)|)$. In each iteration, MatchSim invokes K-M algorithm n^2 times. Suppose there are K iterations and let $L = max_{a,b\in V}(l_{ab})$, the time complexity of MatchSim is thus $O(Kn^2L^3)$.

Space complexity: MatchSim has to store n^2 MatchSim scores. Moreover, the K-M algorithm invoked needs to store the similarity matrix of two objects, the size of which is $O(L^2)$. Therefore, the space complexity of MatchSim is $O(n^2) + O(L^2) = O(n^2 + L^2)$.

The impact of L on the space complexity of MatchSim is rather limited, since usually $n \gg L$. The impact on the time complexity, however, can be very large due to the factor L^3. Thus, to accelerate MatchSim computation, we need to reduce the factor L^3.

4.5 Accelerating techniques

From complexity analysis, we can see that the speed of MatchSim heavily depends on that of K-M algorithm, which is $O(L^3)$ where $L = max_{a,b \in V}(l_{ab})$. We suggest two accelerating techniques for MatchSim. First, an approximation algorithm of complexity $O(L^2)$ is introduced to replace K-M algorithm. Second, for each object, we sort the importance of its neighbors by their PageRank [28] scores and compute MatchSim scores between objects using the top $F(\ll L)$ important neighbors of them only. This may significantly reduce L and accelerate the K-M algorithm as a result. Three approximation algorithms for MatchSim are proposed based on these two techniques and their combination. They are MatchSim$_A$, MatchSim$_F$, and MatchSim$_{AF}$, respectively.

4.5.1 Approximate maximum matching algorithm

In [6], the authors proposed an approximation algorithm, known as the *Path Growing Algorithm* (PGA), for finding a maximum weight matching in an arbitrary graph. The authors proved that the *performance ratio* of GPA is 1/2. Technically, we say that an approximation algorithm has a performance ratio of c, if for all graphs it finds a matching with a weight of at least c times the weight of an optimal solution. The computation time of GPA is $O(|E|) = O(n^2)$ for a bipartite graph of size n. Therefore, in MatchSim, the time required to compute a maximum matching between two objects a and b using GPA drops to $O(l_{ab}^2)$. The complexity of the resulting "approximate" MatchSim, called **MatchSim**$_A$, is consequently reduced to $O(Kn^2L^2)$.

4.5.2 Pruning unimportant neighbors

Because objects in a graph are not equally important (such as web pages), we suggest pruning *unimportant* neighbors to reduce the value of L. It is based on an intuitive assumption that unimportant neighbors contribute less to the measurement of similarity. In this paper, the importance of objects is measured by *PageRank* (PR) scores. Therefore, we suggest another approximate version of MatchSim, named **MatchSim**$_F$, which uses only the top F important neighbors of objects.

The pruning strategy accelerates MatchSim by reducing the value of L. We will show in Sect. 5 that it reduces more than 90% runtime of MatchSim algorithm in the experiments. In this paper, we always assume that PR scores are available; otherwise, we may need to choose other pruning strategies since computing PR scores is also a time-consuming task.

4.6 A toy example

We conclude this section with a toy example, which illustrates the basic process of MatchSim computation. Figure 3 presents a more complete version of the graph G in Fig. 1. For simplicity, we suppose objects $p_i (i = 1, \ldots, 5)$ have no outlink neighbors. We can easily know that $sim(p_i, v) = 0 (\forall v \in G$ and $v \neq p_i, i = 1, \ldots, 5)$.

At the beginning, MatchSim assigns initial values to MatchSim scores, with $sim(x, y) = 0$ if $x \neq y$ or $sim(x, y) = 1$ if $x = y$, for any $x, y \in G$. Next, the MS scores are updated iteratively by applying Eq. (2) until convergence.

In the first iteration, suppose we first update $sim(a, b)$. Because the initial values of $sim(a_i, b_j) (i, j = 1, 2)$ are zeros, we get $sim(a, b) = 0$. Next, by applying Eq. (2), we compute $sim(a_1, b_1) = (sim(p_1, p_1) + sim(p_2, p_2) + sim(p_3, \theta))/3 = 2/3$ and $sim(a_2, b_2) =$

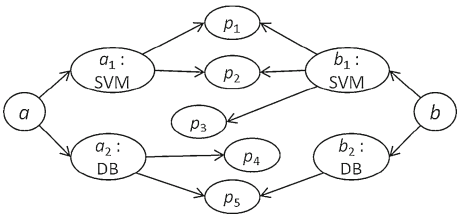

Fig. 3 A toy example

$(sim(p_4, \varphi)+sim(p_5, p_5))/2 = 1/2$, where θ and φ are *dummy* objects. We omit the similar process of updating other MS scores, which are evidently zeros.

In the second iteration, because $sim(a_1, b_1) = 2/3$, $sim(a_2, b_2) = 1/2$, and $sim(a_1, b_2) = sim(a_2, b_1) = 0$, we can find out that $\{(a_1, b_1), (a_2, b_2)\}$ is the *maximum matching* between $O(a)$ and $O(b)$. Therefore, we have $sim(a, b) = (sim(a_1, b_1) + sim(a_2, b_2))/2 = 7/12$. In the third iteration, MatchSim will end because the MS scores remain the same.

5 Experimental results

First, we evaluate the effectiveness of the accelerating techniques on MatchSim and estimate the optimal pruning parameter F. Second, we test MatchSim against other classical neighbor-based methods including Co-citation, Bibliographic coupling, Jaccard Measure, and SimRank. Throughout the experiments, we focus on the top $N = 20$ similar objects returned by the algorithms and fix the iteration numbers of MatchSim and SimRank to be $K = 15$. The hardware environment is Celeron 2.8 G CPU, 4 G memory, and 80 G hard disk. The programs are written in C, and the OS is Windows XP Pro SP2.

5.1 Datasets

We have four real-world datasets. The CW and GS datasets are crawled by ourselves from the Web using BFS (Breadth-First Search) algorithm. The CiteSeer and Cora datasets are two commonly used datasets containing pre-classified academic papers [32]. Brief descriptions on the datasets are as follows.

1. **The CSE Web (CW) dataset** is a set of web pages crawled from the website of our department.[4] It contains 22,615 textual web pages (html or text pages) and 120,947 hyperlinks connecting them together. The average inlink/outlink number is about 5.3.
2. **The Google Scholar (GS) dataset** is a set of academic papers crawled from Google Scholar.[5] It contains 20,000 papers (without fulltext) and 87,717 citations linking them together. To obtain the papers, we first submitted the keyword "web mining" to Google Scholar and employed the top 50 returned papers as seeds to crawl the remaining papers by following the "Cited By" hyperlinks of the returned papers.
3. **The CiteSeer and Cora datasets** are two smaller datasets containing computer science papers and can be downloaded freely on the Web.[6] The papers in both datasets have been classified into classes according to their topics. Because the citation graphs extracted from

[4] http://www.cse.cuhk.edu.hk.

[5] http://scholar.google.com.

[6] http://www.cs.umd.edu/projects/linqs/projects/lbc/index.html.

Table 2 Properties of the datasets

	CW	GS	CiteSeer	Cora
Type of objects	Web page	Paper	Paper	Paper
Type of links	Hyperlink	Citation	Citation	Citation
# of objects	22,615	20,000	2,110	2,485
# of links	120,947	87,717	3,757	5,209
Inlinks/outlinks per object	5.3	4.4	1.8	2.1
Inlink dangling nodes (%)	0%	57.7%	39.4%	42.3%
Outlink dangling nodes (%)	14.7%	0.06%	24.7%	16.4%

the datasets are not fully connected, we use their *maximum graph components* instead of the original graphs in the remaining of the paper. The new CiteSeer and Cora graphs contain 2,110 and 2,485 papers, respectively.

One major problem of the above datasets is *incompleteness*, which results in large amounts of dangling nodes. In GS dataset, due to the crawling strategy, about 57.7% of the papers have no inlinks (we call them the *inlink dangling nodes*), but only about 0.06% have no outlinks (we call them the *outlink dangling nodes*). Similar situation happens to the CW, CiteSeer, and Cora datasets (see the last two rows in Table 2).

Many link-based methods can use both inlinks and outlinks as input, depending on the properties of the datasets. In our experiments, large amounts of dangling nodes can significantly reduce the accuracy of similarity measures. Therefore, choosing suitable kind of links is a very important issue. Actually, in [25], we have reported that by combining both kinds of links, the accuracy link-based similarity measures can be improved. In the experiments of this paper, we choose inlinks for CW dataset and outlinks for other three datasets as default input of the algorithms.

There are some differences between the *web graph* and *citation graph* extracted from the datasets. First, the web graph of CW dataset is relatively complete, while apparently the citation graphs are not so, since typically a computer science paper has more than 10 references. Second, the citation graphs are almost *directed acyclic graphs* since citations rarely form cycles, while the web graph is more complex. The above differences may also influence the practical performance of link-based methods.

We summarize the basic properties of the datasets in Table 2. The distributions of the papers over classes in CiteSeer and Cora datasets are listed in Table 3. Histograms of links in the datasets are given in Appendix *C*.

5.2 Ground truth

A good evaluation of similarity measure is difficult without performing extensive user studies or having a reliable ground truth. In this paper, we choose different metrics to serve as ground truth of similarity for different datasets.

1. For the CW dataset, we use the textual similarity between CW web pages as ground truth and choose the *cosine TFIDF*, which is a widely used text-based similarity metric in IR.
2. For the GS dataset, we use the "Related Articles" provided by Google Scholar as a rough ground truth. Based on our observation, the "Related Articles" are generally reasonable.

Table 3 Distribution of papers over classes

CiteSeer	# of papers	Cora	# of papers
Agents	463	Case_based	285
AI	115	Genetic_algorithms	406
DB	388	Neural_networks	726
IR	304	Probabilistic_methods	379
ML	532	Reinforcement_learning	214
HCI	308	Rule_learning	131
		Rule_theory	344
Total	2,110	Total	2,485

To achieve this, Google Scholar must have used various kinds of article properties, such as textual information (title, keywords, abstract, or maybe full text), authors, or references.

3. For the CiteSeer and Cora datasets, since all of the papers have been classified, we use the classifications as ground truth and adopt the classical *precision*, *recall*, and *F*-measure.

5.2.1 About the cosine TFIDF metric

It has been reported that TFIDF performs poorly in terms of accuracy when applied to the Web [27]. We believe that it is mainly because generally web pages are (1) *extremely diverse*: topics of web pages cover almost every field in the world, and (2) *unreliable* and *untrustworthy*: large amounts of web pages with low-quality or even malicious textual content exist on the Web [13]. All of these may influence the accuracy of text-based similarity measures.

The cosine TFIDF weighting scheme is widely used in IR to determine the similarity between two documents [2,29–31]. However, its precision is not very high [34,35]. In this paper, we use it as a rough metric of similarity for the web pages in CW dataset.

Cosine TFIDF is suitable for the CW dataset because the CW web pages contain high-quality textual contents. First, the topics of CW web pages' contents are more focused and limited. Most of them are about several specific research topics in computer science and/or mathematics. Second, the quality of CW pages are relatively high and can be guaranteed, since they are created and edited by researchers or web administrators. In other words, the CW web pages are more like a traditional well-written and well-organized corpus of academic articles than the unreliable and untrustworthy general Web pages.

5.3 Evaluation methods

Let $top_{A,N}(v)$ denote the set of top N similar objects to object v retrieved by algorithm A. We denote the "overall quality" of $top_{A,N}(v)$ by value $score_{A,N}(v)$. Here, "overall quality" may refer to score of textual similarity or precision, etc, depending on the context. The average of $score_{A,N}(v)$ over $v \in V$, denote by $\Delta(A, N)$, is adopted to measure the quality of the top N results retrieved by algorithms A. That is,

$$\Delta(A, N) = \frac{\sum_{v \in V} score_{A,N}(v)}{\|V\|}.$$

5.3.1 Basic metrics for the datasets

(**1**) **CW dataset:** The cosine TFIDF similarity score of two web pages u and v is just the cosine of the angle between TFIDF vectors of the pages [2], which is defined by

$$cosTFIDF(u, v) = \frac{\sum_{t \in u \cap v} W_{tu} \cdot W_{tv}}{\|u\| \cdot \|v\|},$$

where W_{tu} and W_{tv} are TFIDF weights of term t for web pages u and v, respectively, $\|u\| = \sqrt{\sum_{t \in u} W_{tu}^2}$ and $\|v\| = \sqrt{\sum_{t \in v} W_{tv}^2}$. For the CW dataset, we define

$$score_{A,N}(v) = \sum_{u \in top_{A,N}(v)} cosTFIDF(u, v),$$

and $\Delta^T(A, N) = \Delta(A, N)$, which measures the average cosine TFIDF score of the top N similar web pages returned by algorithm A.
Before applying cosine TFIDF, we pre-process CW dataset with common data cleaning techniques including *stemming* and *removing stop words*.

(**2**) **GS dataset:** For an article v in citation graph G, the list of its "Related Articles" returned by Google Scholar is denoted by $RA(v)$. We define

$$related_N(v) = \{\texttt{top N related articles}v_i | v_i \in RA(v) \cap V\}.$$

The precision of similarity measure A over top N results is:

$$GSprec_{A,N}(v) = \frac{|top_{A,N}(v) \cap related_N(v)|}{|top_{A,N}(v)|}.$$

Therefore, for the GS dataset, we simply define

$$score_{A,N}(v) = GSprec_{A,N}(v),$$

and $\Delta^P(A, N) = \Delta(A, N)$, which measures the average precision of algorithm A over top N results.

(**3**) **CiteSeer and Cora datasets:** In these datasets, two objects are similar if they are classified into the same class. Let $similar(v)$ denote the set of papers whose class labels are the same as that of v. We use the *precision*, *recall*, and *F-measure* to evaluate the performance of algorithm A.

$$precision_{A,N}(v) = \sum_{v \in V} \frac{|top_{A,N}(v) \cap similar(v)|}{|top_{A,N}(v)|},$$

$$recall_{A,N}(v) = \sum_{v \in V} \frac{|top_{A,N}(v) \cap similar(v)|}{N},$$

$$Fscore_{A,N}(v) = \sum_{v \in V} \left(2 \cdot \frac{precision_{A,N}(v) \cdot recall_{A,N}(v)}{precision_{A,N}(v) + recall_{A,N}(v)}\right).$$

Similarly, let $\Delta^{precision}(A, N)$, $\Delta^{recall}(A, N)$, and $\Delta^{Fscore}(A, N)$ denote the metrics of "overall quality" of A over the top N results.

5.3.2 Other metrics

We also designed additional measures to help us look more insights into the accuracy and efficiency of the algorithms. They are described as follows.

(**3**) **Overall Accuracy (OA) and Distance of Accuracy (DA) metrics:** Given top N similar objects, we respectively define the *OA* and *DA* metrics by

$$OA(A, N) = \frac{1}{N}\sum_{i=1}^{N}\Delta(A, i),\ \ DA(A, B, N) = \frac{1}{N}\sum_{i=1}^{N}\frac{|\Delta(A, i) - \Delta(B, i)|}{\Delta(B, i)}.$$

For an algorithm A, we can plot a 2-dimensional *accuracy curve*, with the x-axis representing N and the y-axis representing $\Delta(A, N)$. We use $OA(A, N)$ to reflect the "overall accuracy" of A over the top N rankings, and $DA(A, B, N)$ reflect the "distance" between accuracy curves of algorithms A and B.

(**4**) **Ratio of OA (ROA) and Ratio of Runtime (RRT) metrics:** ROA and RRT are designed for computing the ratio of two algorithms' performance over the top N results in terms of accuracy and running time, respectively. We define the *runtime* of similarity measure A as the time it needs for computing all of the similarity scores. It does not include the time for loading or saving data from or into storage. We denote the average runtime of A by $RT(A)$.

$$ROA(A, B, N) = \frac{OA(A, N)}{OA(B, N)},\ \ RRT(A, B) = \frac{RT(A)}{RT(B)}.$$

5.4 Evaluations on the accelerating techniques

We first evaluate the effectiveness of the accelerating techniques on *MatchSim* (MS). In CW dataset, inlinks are used by Matchsim as input, and in GS dataset, outlinks are used as input. The accelerating techniques include:

T_1 : the *GPA algorithm* for calculating maximum matchings, and
T_2 : the *pruning strategy* based on PR scores.

Besides, we need to estimate the optimal pruning parameter F for MS_{AF}.

We set the F to be 10, 20, 30, 40 and ∞, where ∞ means "no pruning" on neighbors. For each F, we run MS_F and MS_{AF} on the datasets. Note that MS_∞ and $MS_{A\infty}$ are actually MS and MS_A, respectively.

5.4.1 Results on GS dataset

Tables 4 and 5 show the results on CW and GS datasets. For an algorithm A (which is MS_F or MS_{AF}), we define $DA = DA(A, MS_\infty, N)$, $ROA = ROA(A, MS_\infty, N)$, and $RRT = RRT(A, MS_\infty)$ in the tables. That is, we use the results of MS as the benchmark to assess algorithm A.

More precisely, $DA(\geq 0)$ measures the closeness between the result of A and that of MS, where smaller DA means closer results. $ROA(\geq 0)$ measures the ratio of "overall accuracy" of A to that of MS, where greater ROA means A achieves better results. $RRT(\geq 0)$ measures the "relative speed" of A to MS, where smaller RRT means A is faster. When the results of A and MS are the same, the values of the metrics are 0, 100%, and 100%, respectively.

Table 4 MatchSim: accelerating techniques on GS

F		10	20	30	40	∞
$P(\%)$		7.65	4.07	2.73	1.94	0.00
MS_F	$DA(10^{-2})$	12.44	6.06	3.34	1.42	0.00
	$ROA(\%)$	87.64	94.09	96.78	98.82	100
	$RRT(\%)$	4.81	8.24	11.88	15.86	100
MS_{AF}	$DA(10^{-2})$	11.88	6.00	2.89	1.21	0.94
	$ROA(\%)$	88.10	94.06	97.16	98.90	99.54
	$RRT(\%)$	1.81	2.35	2.76	3.13	6.50

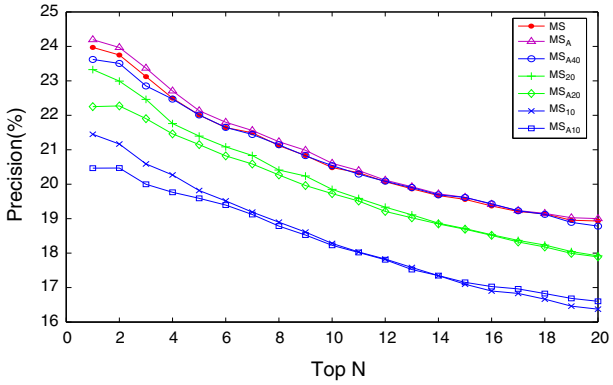

Fig. 4 Accuracy of accelerating techniques on GS dataset

Additionally, to reveal the number of nodes affected by parameter F, we denote by P the percentage of the nodes whose neighbors are pruned given a certain F. From Table 4, we can see that:

1. For MS_F, its accuracy increases and approaches that of MS as F increases (i.e., $F \rightarrow \infty \Rightarrow ROA \rightarrow 100\%$, $DA \rightarrow 0$). On the other hand, its runtime also increases with F, but is much smaller than that of MS when $F \leq 40$. These results show that the pruning technique (T_2) really works. That is, T_2 accelerates MatchSim significantly with relatively small loss of accuracy.
2. For MS_{AF}, it always runs much faster than MS_F for a given $F (F = 10, \ldots, 40, \infty)$ with almost the same accuracy. In the special case, when $F = \infty$ (i.e., only T_1 is used), the accuracy of MS_A is very close to that of MS ($DA = 0.94 \times 10^{-2}$ and $ROA = 99.54\%$), while the runtime is much less ($RRT = 6.50\%$). This shows that the T_1 technique also works.

We also plot some of the accuracy curves of the algorithms in Fig. 4. (We exclude the curve of MS_{40}, which are very close to those of MS, MS_A, and MS_{A40}, from Figs. 4 and 5 to make the figures clearer.) Based on these results, we conclude that the best version of MatchSim for the GS dataset is MS_{A40}.

Table 5 MatchSim: accelerating techniques on CW

F		10	20	30	40	∞
$P(\%)$		6.42	2.77	1.46	0.84	0.00
MS_F	$DA(10^{-2})$	22	11.85	6.40	1.82	0.00
	$ROA(\%)$	78.08	88.20	94.58	98.81	100
	$RRT(\%)$	3.91	5.51	8.02	9.73	100
MS_{AF}	$DA(10^{-2})$	23.62	14.17	7.31	2.76	1.08
	$ROA(\%)$	76.45	85.87	93.12	97.22	98.99
	$RRT(\%)$	0.52	1.06	1.63	2.25	2.89

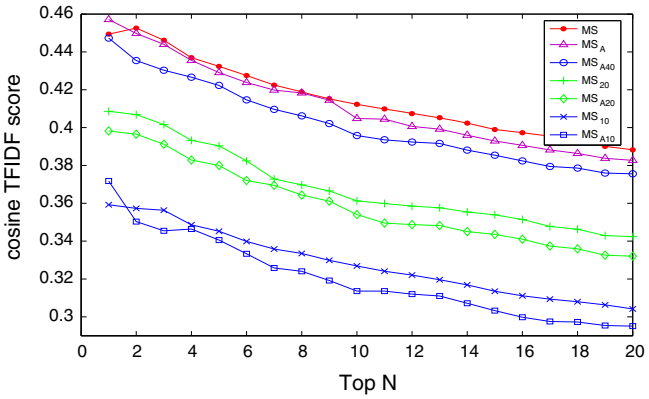

Fig. 5 Accuracy of accelerating techniques on CW dataset

5.4.2 Results on CW dataset

We also conduct experiments on the CW dataset and show the results in Table 5 and Fig. 5. It is easy to see that a similar conclusion can be drawn from the results. We therefore suggest MS_{A40} to be the best version of the approximate MatchSim for CW dataset, too.

5.5 Testing MatchSim on CW and GS datasets

In this section, we compare the performance of MS and MS_{A40} with other neighbor-based similarity measures, including Bibliographic coupling (BC), Co-citation (CC), Jaccard Measure (JM), and SimRank (SR). The formal definition of these algorithms has been given in Sect. 2. For SimRank, we set $\gamma = 0.8$. Because of the reason explained in Sect. 5.1, when applied to the CW dataset, MatchSim, SimRank, and Jaccard Measure use inlinks as input and use outlinks when applied to other datasets.

5.5.1 The accuracy

We plot in Fig. 6 the $\Delta^P(A, N)$ curves of the methods on GS dataset and in Fig. 7 the $\Delta^T(A, N)$ curves of the methods on CW dataset. We also report in Table 6 the $ROA(*, MS, 20)$ values of the algorithms to compare the overall performance in accuracy

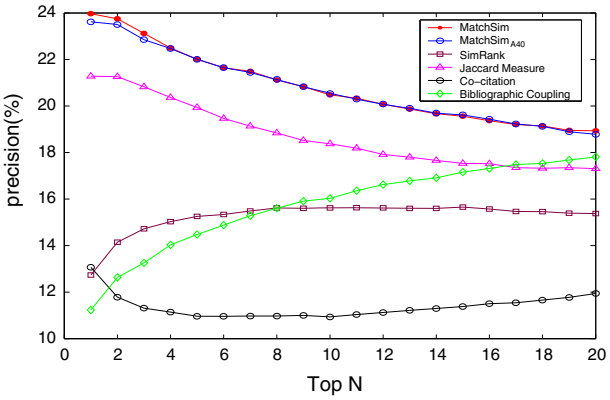

Fig. 6 Performance results on the GS dataset

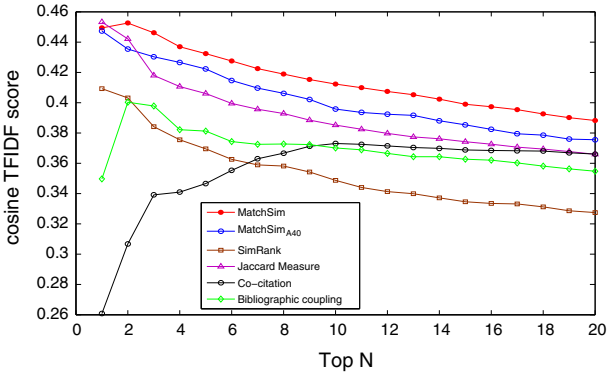

Fig. 7 Performance results on the CW dataset

Table 6 ROA of the algorithms on GS and CW datasets

	BC	CC	JM	SR	MS_{A40}	MS
GS	0.55	0.76	0.89	0.73	1.00	1.00
CW	0.89	0.85	0.94	0.85	0.97	1.00

of the algorithms. The observed results (in *italic*) and the corresponding discussions are listed as follows.

1. *The two versions of MatchSim, MS, and MS_{A40} outperform all of the other methods in almost all cases.* This demonstrates the effectiveness of the proposed MatchSim method and the accelerating techniques.
2. *Jaccard Measure also performs very well*. Considering that it needs much less runtime than MatchSim and SimRank, this method would be a good tradeoff between accuracy and efficiency.
3. *MatchSim and Jaccard Measure perform much better on the top results (e.g., top 5) than other methods*. This shows that these methods are particularly suitable for the scenarios where top results are very important.

Table 7 Given web page KING, the top-1 results returned by the algorithms

	The most similar page	# of inlinks	# of common inlinks
MatchSim	CHAN	10	5
Jaccard	LYU	15	7
Co-citation	LYU	15	7
SimRank	MEMPM	1	1

4. *Co-citation performs very poorly on the GS dataset*. This is because Co-citation algorithm uses inlinks in GS dataset, more than half of which are dangling nodes. Actually, other methods including MatchSim and SimRank also have the same problem. This indicates that choosing the "right" type of neighbors as input is very important to the neighbor-based algorithms.

5.5.2 The runtime

In the experiments on CW and GS datasets, we observed that the runtime of MS_{A40} and SR are more than 30 and 20 min, respectively, while the neighbor-counting algorithms need only a few seconds. This is because MatchSim and SimRank are iterative algorithms; thus, the cost of computing each similarity score is very high (both are $O(KL^2)$, where L is the average number of neighbors and K is the number of iterations). The complexity of direct algorithms is $O(L)$. Theoretically, the runtime of MatchSim is $O(KL)$ times longer than that of direct algorithms, and typically $K = 15$ and L is around 5 in the experiments. Designing specific techniques to significantly improve the efficiency of MatchSim is one of the most important and challenging problems.

5.5.3 A qualitative example

Now let's see a simple example to get more insights into the algorithms. Given web page KING,[7] which has 10 inlinks and 2 outlinks, we list the most similar web pages returned by the algorithms in Table 7. Since no web page shares any common outlink with KING, *Bibliographic Coupling* method cannot find any similar web pages. We thus omit it from the table.

We can see that *MatchSim* returns CHAN,[8] which has 10 inlinks and shares 5 common inlinks with KING. *Jaccard* and *Co-citation* return LYU,[9] which has 15 inlinks and shares 7 common inlinks with KING. All of these results seem reasonable, and it is hard to tell which one is the best.

The MEMPM[10] returned by *SimRank* is obviously not good. The only inlink of MEMPM is the homepage of mempm_toolbox, which also links to KING. The reason why SimRank choose MEMPM to be the most similar web page to KING is caused by the counterintuitive loophole criticized in Sect. 4.1.

[7] Prof. King's homepage http://www.cse.cuhk.edu.hk/king.

[8] Prof. Chan's homepage http://www.cse.cuhk.edu.hk/lwchan/.

[9] Prof. Lyu's homepage http://www.cse.cuhk.edu.hk/lyu.

[10] The register web page of mempm_toolbox.

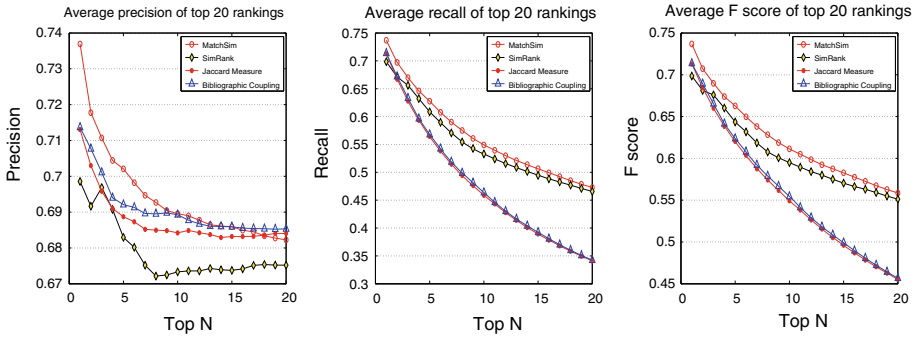

Fig. 8 Average precision, recall, and F scores on the CiteSeer dataset

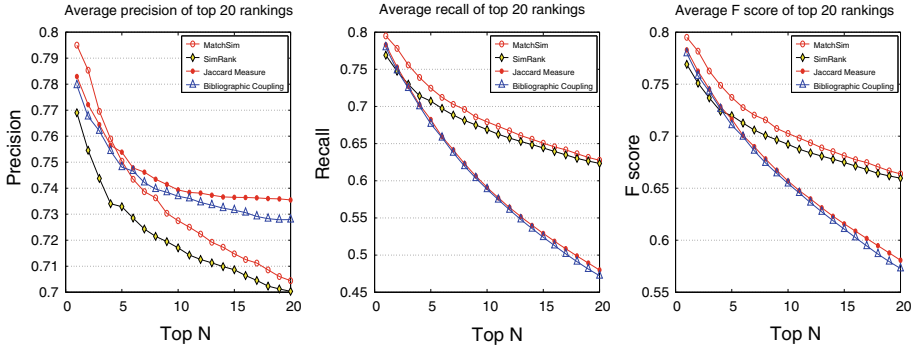

Fig. 9 Average precision, recall, and F scores on the Cora dataset

5.6 Testing MatchSim on citeSeer and cora datasets

In this section, we test MatchSim against other methods on two smaller datasets: the CiteSeer and Cora datasets. The papers in both datasets have been pre-classified into classes according to topics. On our observation, the algorithms using outlinks as input perform much better than those using inlinks. Therefore, we use outlinks as input for MatchSim, SimRank, and Jaccard Measure. Because Co-citation, which uses inlinks, performs very poorly in terms of precision, we exclude its curves to make the figures clearer.

5.6.1 Experimental results

The results are presented in Figs. 8 and 9, respectively. In the figures, the average scores are taken over the results returned by the algorithms to all the objects that have outlinks; we omit the objects that have no outlinks to emphasize the difference of the competing algorithms, since all of the algorithms return no similar objects in these cases. The runtime is given in Table 8. We summarize the basic observations and interpretations as follows. More discussions can be found in Sect. 5.6.2.

1. **Precision**: MatchSim performs the best on CiteSeer, but not very well on Cora. SimRank achieves the worst precision on both datasets, which is actually caused by its counterintuitive loophole.

Table 8 Runtime (in second) of the algorithms on CiteSeer and Cora datasets

	BC	CC	JM	SR	MS
CiteSeer	171	132	174	1,632	1,680
Cora	99	97	99	1,515	1,275

Table 9 Comparison of top the 5 results, given object article 36

Top	SimRank (79)	MatchSim (79)	Jaccard (30)
1	114_Agents {329}	63_AI {5, 97, 274, 329, 661, 916}	63_AI {see left}
2	203_Agents {274, 661}	97_AI {5, 329, 661}	97_AI {see left}
3	735_Agents {274, 661}	813_ML {77, 339, 735, 916}	5_AI {97, 916}
4	948_AI {329, 661}	617_ML {126, 243, 328, 661, 916}	203_Agents {274, 661}
5	97_AI {5, 329, 661}	949_AI {5, 63, 97}	603_Agents {5, 661}

2. **Recall**: MatchSim and SimRank achieve much higher recall than the neighbor-counting algorithms. This is because both algorithms take similarities between neighbors into account. As a result, they can retrieve more objects which may certainly increase the chance of finding more real similar objects.
3. **F score**: The overall performance of MatchSim is the best on both datasets. BC and JM are the worst due to their low recall scores.
4. **Runtime**: From Table 8, we can see that the efficiency is the major bottleneck of MatchSim. The problem becomes worse as the size of graph increases.

5.6.2 A qualitative example

Let's look further into the results by examining an example. The object is article 36 from CiteSeer dataset, which has the label "AI" and cites six other articles {5, 97, 274, 329, 661, 916}. Table 9 lists the top 5 results (from the top down) returned by each algorithm, the format of which is article's id followed by its label and references. The numbers in the first row indicate the total numbers of articles retrieved by the algorithms. Here, we omit BC whose results are the same as those of JM.

From the table, we can see that

(1) MS and SR find more results. It can certainly lead to higher possibility of finding the "real similar articles", i.e., those having the same labels as those of the object article.
(2) The precision of SR is the worst. This is because of its loophole criticized before. For example, although both reference lists of articles 114 and 97 are subsets of article 36's references, intuitively article 97, which has more references, should be ranked higher. Obviously, SimRank made a mistake here.
(3) The rankings produced by MS and JM are more reasonable. The precision of JM is even better than that of MS in this example. However, JM finds out less articles (lower recall), which influences its overall performance. SimRank, on the other hand, can find much more articles (much higher recall), which may lead to higher F scores.

To summarize, from this example, we can see that Matchsim can find out more results than the neighbor-counting methods and at the same time ensure that the results are reasonable by conforming to the intuitions of similarity. Therefore, it can achieve the best overall performance (the highest F scores) in the experiments. Certainly, the major problem of MatchSim is its computational efficiency, which is our future work.

6 Conclusion and future work

In this paper, we propose a neighbor-based similarity measure called MatchSim, based on the intuition that "similar objects have similar neighbors." By taking neighbors' similarity into account and conforming to the intuitions of similarity, our method can produce higher quality results. The main contributions of the paper are summarized as follows:

1. Proposing a novel neighbor-based similarity measure called MatchSim, which computes object similarity based on maximum neighborhood matching.
2. Suggesting accelerating techniques to improve computational efficiency of MatchSim.
3. Conducting extensive experiments on real-world datasets to demonstrate the effectiveness of MatchSim.

There are a number of avenues for future work: (1) The efficiency of MatchSim have to be improved to make the algorithm more practical. (2) MatchSim can be easily extended to the "bipartite" version, which can be employed in recommender systems. (3) Inlinks and outlinks are two kinds of properties of networked objects. We have noticed that by combining both kinds of links, the accuracy of the several link-based methods can be improved. We believe this will also work on MatchSim and leave it one of our future work.

Acknowledgments We thank anonymous reviewers for their very useful comments and suggestions. The work described in this paper was fully supported by two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK 415410 and Project No. CUHK 415210), and a Google Focused Grant Project under "Mobile 2014".

Appendix A: Convergence proof of MatchSim

We now prove the existence and uniqueness of the n^2-dimensional fixed point $sim(*,*)$ of the n^2 MatchSim equations (4). First, we give a simple fact that, for any a and b, the sequence $sim_k(a,b)(k=1,2,\ldots)$ is bounded and nondecreasing.

Fact. $0 \le sim_k(a,b) \le sim_{k+1}(a,b) \le 1, \forall a,b \in V, k \ge 0$.

By using mathematical induction, the proof is easy and thereby is omitted here. By the Completeness Axiom of calculus, each sequence $sim_k(a,b)$ converges to a limit $sim(a,b) \in [0,1]$. Therefore, the fixed point $sim(*,*)$ of the MatchSim equations exists.

Next, we prove the uniqueness of fixed point $sim(*,*)$. Suppose $sim(*,*)$ and $sim'(*,*)$ are two fixed points of the n^2 MatchSim equations. For all $a,b \in V$, let $\delta(a,b) = |sim(a,b) - sim'(a,b)|$ be their difference. Let $D = \delta(x,y) = max_{a,b \in V}\delta(a,b)$, where $x,y \in V$, be the maximum value of any difference. We need to prove $D=0$. Certainly $D=0$ if $x=y$, in which case $sim(x,y) = sim'(x,y) = 1$, or if x or y has no neighbors, in which case $sim(x,y) = sim'(x,y) = 0$.

In other cases ($x \ne y$ and $|I(x)||I(y)| \ne 0$), we suppose $sim(x,y) > sim'(x,y)$. From Eq. (2),

$$D = \delta(x,y) = sim(x,y) - sim'(x,y)$$
$$= \frac{\widehat{W}(x,y)}{max(I(x),I(y))} - \frac{\widehat{W}'(x,y)}{max(I(x),I(y))} = \frac{1}{max(I(x),I(y))} \cdot [W(m_{xy}) - W'(m'_{xy})],$$

where m_{xy} (m'_{xy}) is a maximum matching between $I(a)$ and $I(b)$ computed using $sim(*,*)$ ($sim'(*,*)$), and $W(m_{xy})$ ($W'(m'_{xy})$) is the corresponding maximum weight.

Let M_{xy} be the set of matchings between $I(x)$ and $I(y)$, then we have $m_{xy}, m'_{xy} \in M_{xy}$, and

$$W'(m'_{xy}) = max_{m \in M_{xy}} W'(m) \Rightarrow W'(m'_{xy}) \geq W'(m_{xy}).$$

Thus,

$$\begin{aligned} W(m_{xy}) - W'(m'_{xy}) \leq W(m_{xy}) - W'(m_{xy}) &= \sum_{(u,v)\in m_{xy}} sim(u,v) - \sum_{(u,v)\in m_{xy}} sim'(u,v) \\ &= \sum_{(u,v)\in m_{xy}} [sim(u,v) - sim'(u,v)] \\ &\leq \sum_{(u,v)\in m_{xy}} |sim(u,v) - sim'(u,v)| \qquad (5) \\ &\leq \sum_{(u,v)\in m_{xy}} D. \qquad (6) \end{aligned}$$

Therefore,

$$\begin{aligned} D &= \frac{1}{max(I(x), I(y))} \cdot [W(m_{xy}) - W'(m'_{xy})] \\ &\leq \frac{1}{max(I(x), I(y))} \cdot [W(m_{xy}) - W'(m_{xy})] \\ &\leq \frac{1}{max(I(x), I(y))} \cdot max(I(x), I(y)) \cdot D = D. \end{aligned}$$

Here, we come to $D \leq D$. Next, we continue the proof under two complementary conditions.

Condition (1): the "=" relationships in the above inequalities *always* hold for any $(x, y) \in S$, where

$$S = \{(x, y) | sim(x, y) - sim'(x, y) = D\}.$$

From inequalities (5) and (6), it must follow that $\forall (x, y) \in S \Rightarrow m_{xy} \subset S$. On the other hand, we also have

$$sim(x, y) = \frac{\sum_{(u,v)\in m_{xy}} sim(u, v)}{max(I(x), I(y))}.$$

Thus, we get fact(1): for any $(x, y) \in S$, the value of $sim(x, y)$ *only* depends on those of $sim(u, v)$, where $(u, v) \in m_{xy} \subset S$. (That is, the computation of $sim(x, y)$ is closed on set S.)

We also know fact(2): since $x \neq y$ for any $(x, y) \in S$, the initial value of $sim(x, y)$, $(x, y) \in S$, is zero. (Sect. 4.3)

From facts (1) and (2), it follows that $sim(x, y) = 0$, for any $(x, y) \in S$. Since $sim(x, y) - sim'(x, y) = D \geq 0$ and $sim'(x, y) \geq 0$, it follows $D = 0$.

Condition (2): the "=" relationships in the above inequalities do not *always* hold for any $(x, y) \in S$.

Evidently, we can always choose a $(x, y) \in S$ so that at least one of the "=" relationships in the inequalities does not hold. By restarting the proof process with this (x, y), we will come to $D < D$, which does not hold for any D.

From the proofs under *Condition*s (1) and (2), it follows that $D = 0$.

Appendix B: The assignment problem

The **Assignment Problem** consists of finding a maximum (weight) matching in a weighted bipartite graph. Given two sets, A and B, of equal size n, together with a weight function $w : A \times B \rightarrow \Re^+$, we obtain a weighted bipartite graph $G = (A + B, E, w)$, where $E = \{(a, b)|a \in A, b \in B\}$. A *matching* in G is a set of pairwise non-adjacent edges, i.e., no two edges share a common vertex. In other words, a matching in G is a bijection $m : A \leftrightarrow B$. Let M denote the set of matchings between A and B, and $W(m) = \sum_{(a,b)\in m} w(a, b)$ denote the weight of matching m. The objective of the assignment problem is to find a *maximum matching*, denoted by m^*, such that:

$$W(m^*) = max_{m\in M} W(m).$$

Evidently, m^* may not be unique.

If the graph is not completely bipartite (A and B are not of equal size), dummy vertices and zero-weighted edges are inserted to make up the missing part. The problem can then be solved in the usual way and still give the best solution to the problem. Therefore, in the paper, we always convert A and B to be "equally sized" before computing the m^*. Thus, we always consider A and B to be of size $max(|A|, |B|)$.

Appendix C: Histograms of links in the datasets

See Figs. 10, 11, 12 and 13

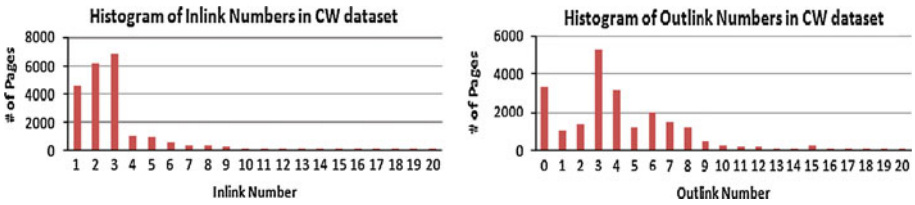

Fig. 10 Histograms of links in CW dataset (Inlink/Outlink Number ≤20)

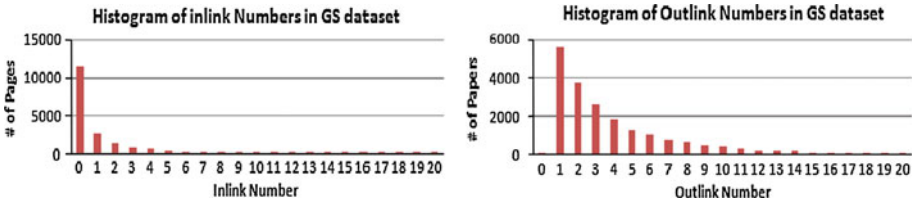

Fig. 11 Histograms of links in GS dataset (Inlink/Outlink Number ≤20

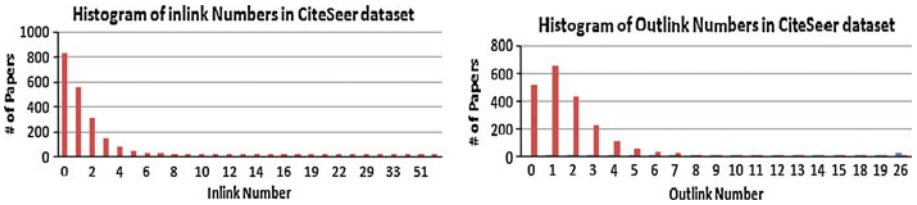

Fig. 12 Histograms of links in CiteSeer dataset

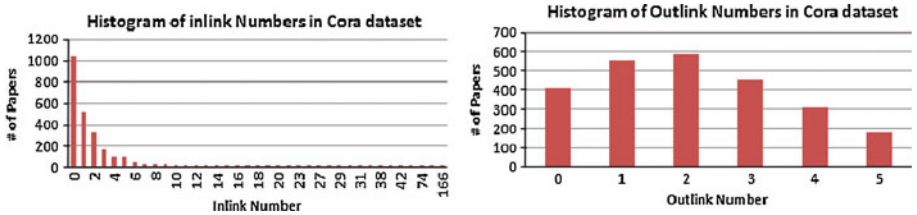

Fig. 13 Histograms of links in Cora dataset

References

1. Aliguliyev RM (2009) A new sentence similarity measure and sentence based extractive technique for automatic text summarization. Expert Syst Appl 36(4):7764–7772
2. Baeza-Yates RA, Ribeiro-Neto BA (1999) Modern information retrieval. ACM Press/Addison-Wesley, NY
3. Burkard R, Dell'Amico M, Martello S (2009) Assignment problems. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA
4. Cunningham P (2009) A taxonomy of similarity mechanisms for case-based reasoning. IEEE Trans Knowl Data Eng 21(11):1532–1543
5. Dean J, Henzinger MR (1999) Finding related pages in the World Wide Web. Comput Netw (Amsterdam, Netherlands, 1994) 31(11–16):1467–1479
6. Drake DE, Hougardy S (2003) A simple approximation algorithm for the weighted matching problem. Inf Process Lett 85(4):211–213
7. Flake GW, Lawrence S, Giles CL, Coetzee FM (2002) Self-organization and identification of web communities. Computer 35(3):66–71
8. Fogaras D, Rácz B (2005) Scaling link-based similarity search. In: WWW '05: proceedings of the 14th international conference on World Wide Web, ACM, New York, USA, pp. 641–650
9. Formica A, Elaheh P (2010) Content based similarity of geographic classes organized as partition hierarchies. Knowl Inf Syst 20(2):221–241
10. Goldberger J, Gordon S, Greenspan H (2003) An efficient image similarity measure based on approximations of kl-divergence between two gaussian mixtures. In: ICCV'03: proceedings of the 9th IEEE international conference on computer vision, IEEE Computer Society, Washington, DC, USA, pp. 487
11. Gueguen L, Datcu M (2008) A similarity metric for retrieval of compressed objects: application for mining satellite image time series. IEEE Trans Knowl Data Eng 20(4):562–575
12. Gupta A, Ying L (1999) On algorithms for finding maximum matchings in bipartite graphs. In: Technical report RC 21576 (97320), IBM T. J. Watson Research Center
13. Gyöngyi Z, Molina HG (2005) Web spam taxonomy. In: First international workshop on adversarial information retrieval on the Web (AIRWeb 2005)'
14. Jain AK, Dubes RC (1988) Algorithms for clustering data. Prentice-Hall, Inc., NJ, USA
15. Jeh G, Widom J (2002) SimRank: a measure of structural-context similarity. In: KDD '02: proceedings of the 8th ACM SIGKDD, ACM Press, NY, USA, pp. 538–543
16. Katz L (1953) A new status index derived from sociometric analysis. Psychometrika 18(1):39–43
17. Kessler M (1963) Bibliographic coupling between scientific papers. Am Doc 14(10–25)
18. Kleinberg JM (1999) Authoritative sources in a hyperlinked environment. JACM 46(5):604–632
19. Kuhn HW (1955) The Hungarian method for the assignment problem. Naval Res Logist Quart 2:83–97

20. Li Y, McLean D, Bandar ZA, O'Shea JD, Crockett K (2006) Sentence similarity based on semantic nets and corpus statistics. IEEE Trans Knowl Data Eng 18(8):1138–1150
21. Lian X, Chen L (2008) Efficient similarity search over future stream time series. IEEE Trans Knowl Data Eng 20(1):40–54
22. Liben-Nowell D, Kleinberg J (2003) The link prediction problem for social networks. In: CIKM '03: prodeedings of the 12th international conference on information and knowledge management, ACM, pp. 556–559
23. Lin D (1998) An information-theoretic definition of similarity. In: ICML '98: proceedings of the fifteenth international conference on machine learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 296–304
24. Lin Z, King I, Lyu MR (2006) PageSim: a novel link-based similarity measure for the World Wide Web. In: WI '06: proceedings of the 5th international conference on web intelligence, IEEE Computer Society, Hong Kong, pp. 687–693
25. Lin Z, Lyu MR, King I (2007) Extending link-based algorithms for similar web pages with neighborhood structure. In: WI '07: proceedings of the 6th international conference on web intelligence, IEEE Computer Society, Washington, DC, USA, pp. 263–266
26. Lu W, Janssen J, Milios E, Japkowicz N, Zhang Y (2006) Node similarity in the citation graph. Knowl Inf Syst 11(1):105–129
27. Maguitman AG, Menczer F, Roinestad H, Vespignani A (2005) Algorithmic detection of semantic similarity. In: WWW '05: proceedings of the 14th international conference on World Wide Web, ACM, New York, NY, USA, pp. 107–116
28. Page L, Brin S, Motwani R, Winograd T (1998) The PageRank citation ranking: bringing order to the Web, Technical report, Stanford Digital Library Technologies Project
29. Ramos J (2003) Using TF-IDF to determine word relevance in document queries, Technical report, Department of Computer Science, Rutgers University, 23515 BPO Way, Piscataway, NJ, 08855e
30. Salton G (1989) Automatic Text Processing. Addison-Wesley, MA
31. Salton G, Buckley C (1987) Term weighting approaches in automatic text retrieval, Technical report, Ithaca, NY, USA
32. Sen P, Namata GM, Bilgic M, Getoor L, Gallagher B, Eliassi-Rad T (2008) Collective classification in network data. AI Magazine 29(3):93–106
33. Small H (1973) Co-citation in the scientific literature: a new measure of the relationship between two documents. J Am Soc Inf Sci 24(265–269)
34. Sugiyama K, Hatano K, Yoshikawa M, Uemura S (2003) Refinement of TF-IDF schemes for web pages using their hyperlinked neighboring pages. In: HYPERTEXT '03: proceedings of the 14th ACM conference on Hypertext and hypermedia, ACM, NY, USA, pp. 198–207
35. Sugiyama K, Hatano K, Yoshikawa M, Uemura S (2005) Improvement in TF-IDF scheme for web pages based on the contents of their hyperlinked neighboring pages. Syst Comput Japan 36(14):56–68
36. van Rijsbergen CJ (1979) Information Retrieval. Butterworth-Heinemann
37. Wan X (2008) Beyond topical similarity: a structural similarity measure for retrieving highly similar documents. Knowl Inf Syst 15(1):55–73
38. Wang H (2006) Nearest neighbors by neighborhood counting. IEEE Trans Pattern Anal Mach Intell 28(6):942–953
39. Wang H, Murtagh F (2008) A study of the neighborhood counting similarity. IEEE Trans Knowl Data Eng 20(4):449–461
40. Wei F, Li W, Lu Q, He Y (2010) A document-sensitive graph model for multi-document summarization. Knowl Inf Syst 22(2):245–259

Author Biographies

Zhenjiang Lin received the B.S. (1998) and M.S. (2003) in computational mathematics from Nankai University and Tsinghua University, respectively. He is currently a Ph.D. candidate in the Department of Computer Science and Engineering at the Chinese University of Hong Kong. His research interests include recommendation algorithms, social network analysis, web mining, link analysis, and link-based similarity measurement techniques and their applications. He has technical publications in journals (TKDE and AMC) and conferences (CIKM and WI). He also has many years of experience in system design and software development. Currently, he is looking for an academic or industrial position related to his research and work backgrounds.

Michael R. Lyu received the B.S. (1981) in electrical engineering from National Taiwan University, the M.S. (1985) in computer engineering from University of California, Santa Barbara, and the Ph.D. (1988) in computer science from University of California, Los Angeles. He is a Professor in the Computer Science and Engineering Department of the Chinese University of Hong Kong. His research interests include software reliability engineering, software fault tolerance, distributed systems, data mining, social networks, Web engineering, multimedia technologies, and mobile networks, where he has published close to 400 papers in these areas. He has been an Associate Editor of IEEE Transactions on Reliability, IEEE Transactions on Knowledge and Data Engineering, Journal of Information Science and Engineering, and Wiley Software Testing, Verification & Reliability Journal. Dr. Lyu is an IEEE Fellow, an AAAS Fellow, and received IEEE Reliability Society 2010 Engineer of the Year Award.

Irwin King research interests include machine learning, social computing, web intelligence, data mining, and multimedia information processing. In these research areas, he has over 210 technical publications in journals and conferences. In addition, he has contributed over 20 book chapters and edited volumes. He is the Book Series Editor for Social Media and Social Computing with Taylor and Francis. He is also an Associate Editor of the IEEE Transactions on Neural Networks (TNN). He is a senior member of IEEE and a member of ACM, International Neural Network Society (INNS), and Asian Pacific Neural Network Assembly (APNNA). Currently, he is a member of the Board of Governors and Vice-President for Membership of the International Neural Network Society (INNS). He is Professor of Computer Science and Engineering Department at the Chinese University of Hong Kong. Currently, he is on leave to work with AT&T Labs Research and is also a visiting professor with the University of California, Berkeley.