Making More Wikipedians: Facilitating Semantics Reuse for Wikipedia Authoring*

Linyun Fu, Haofen Wang, Haiping Zhu, Huajie Zhang, Yang Wang, and Yong Yu

Apex Data and Knowledge Management Lab
Dept. of Computer Science and Engineering, Shanghai Jiao Tong University
800 Dongchuan Rd, Shanghai, P.R. China, 200240
{fulinyun,whfcarter,zhu,zhjay,wwwy,yyu}@apex.sjtu.edu.cn

Abstract. Wikipedia, a killer application in Web 2.0, has embraced the power of collaborative editing to harness collective intelligence. It can also serve as an ideal Semantic Web data source due to its abundance, influence, high quality and well-structuring. However, the heavy burden of up-building and maintaining such an enormous and ever-growing online encyclopedic knowledge base still rests on a very small group of people. Many casual users may still feel difficulties in writing high quality Wikipedia articles. In this paper, we use RDF graphs to model the key elements in Wikipedia authoring, and propose an integrated solution to make Wikipedia authoring easier based on RDF graph matching, expecting making more Wikipedians. Our solution facilitates semantics reuse and provides users with: 1) a link suggestion module that suggests and auto-completes internal links between Wikipedia articles for the user; 2) a category suggestion module that helps the user place her articles in correct categories. A prototype system is implemented and experimental results show significant improvements over existing solutions to link and category suggestion tasks. The proposed enhancements can be applied to attract more contributors and relieve the burden of professional editors, thus enhancing the current Wikipedia to make it an even better Semantic Web data source.

1 Introduction

The past six years have witnessed the tremendously rapid growth of Wikipedia into the largest free encyclopedia that human beings have ever had. Up till now, the advocates have developed 251 languages of Wikipedias[1], among which the English version[2] is reported to own a prodigious number of more than 1,750,000 articles[3]. The huge impact of Wikipedia has propelled it into the top 20 most popular Web sites on the planet[4]. Moreover, [11] finds that Wikipedia comes close to Encyclopaedia

* This work is funded by IBM China Research Lab.
[1] http://meta.wikimedia.org/wiki/List_of_Wikipedias, accessed on April 27, 2007.
[2] http://en.wikipedia.org/
[3] http://en.wikipedia.org/wiki/Special:Statistics, accessed on April 27, 2007.
[4] http://www.alexa.com/data/details/traffic_details? y=t&url=Wikipedia.org, accessed on April 27, 2007.

Britannica in terms of the accuracy of its science entries, which shows that Wikipedia articles are of high quality and can provide accurate definitions for their topics.

In addition to its abundance, influence and high quality, Wikipedia is well-structured and can serve as an ideal Semantic Web data source. Thus it arouses great interests from the Semantic Web community [2, 9, 1, 4, 10, 3, 7]. All these studies are based on two critical characteristics of Wikipedia, i.e., (*internal*) *links* and *categories*. Links between Wikipedia articles allow the user to access information related to the article she is reading to perform *hyperreading* [15]. These links also point out the associated concepts and can be considered as the carriers of semantic relations. Categories help organize Wikipedia articles in a hierarchical structure. Such a hierarchy, although not so strict [5], is a quite valuable characteristic for a widely accepted Semantic Web data source.

Although links and categories are indispensable for Wikipedia, they may cause much trouble to users who want to make contributions, especially those newcomers. While authoring a Wikipedia article, a casual user often feels at a loss due to lack of knowledge about the existing information accommodated in the system. She may wonder *when it is necessary to provide a link to a related article for readers' reference* and *what categories are proper to characterize an article.* Finding answers to these two kinds of questions distracts the user from authoring the article itself. The user may simply ignore these questions and leave them to professional Wikipedia editors, or even get frustrated and give up making contributions.

In fact, the problem has already been quite serious. [22] shows that 80% articles are contributed by 10% contributors, which means most contributors do not contribute actively. Jimmy Wales, the founder of Wikipedia, even pessimistically declares that the most active 2%, which is 1400 people, did 73.4% of all the edits [12], which implies the heavy burden over professional Wikipedia editors. Properly providing users with information about links and categories at authoring time will greatly relieve the burden of Wikipedia editors and attract many more potential contributors.

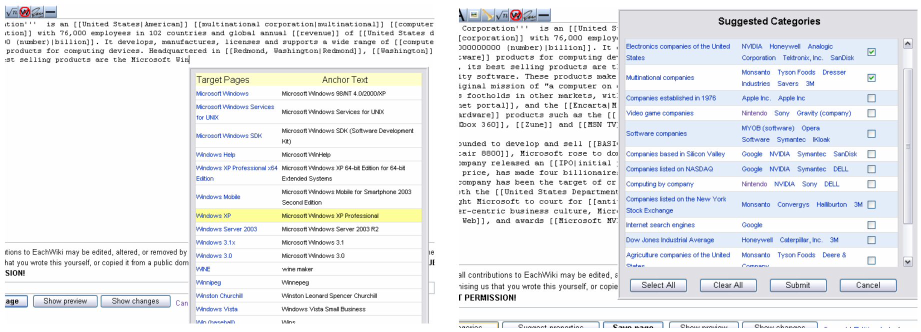
In this paper, we propose our solution to the above problem by equipping the current Wikipedia with 1) a **link suggestion** module that seamlessly integrates search and authoring to provide the user with proper inter-article links, and 2) a **category suggestion** module that helps the user to find appropriate categories for her article. The current best practice on link suggestion is prefix matching over titles of Wikipedia articles, and existing document classification approaches are not proper for the category suggestion task due to their poor effectiveness and efficiency when dealing with large-scale category systems [27].

The proposed solution is based on the idea of RDF graph matching since elements in Wikipedia can be represented as resource graphs in RDF [16]. Our method emphasizes making better use of the shallow semantics in Wikipedia. We first use RDF graphs to model resources and queries concerning our tasks, and then show how this model can be applied to find information about links and categories for the user. A prototype system has been implemented and experimental results convincingly validate the effectiveness and efficiency of our solution.

Figure 1 gives two snapshots of our prototype system, namely EachWiki[5]. When a user types some beginning part of a phrase, the link suggestion module will be

[5] Accessible at http://eachwiki.apexlab.org/

triggered and pop up a list of suggested links for the phrase. The user can select the intended link to complete and replace the prefix that has just been input, as shown in Figure 1.a. When the user confirms the changes she has made on the article by saving the page, the system will analyze the newly edited article and provide the user with a list of suggested categories. The user can select several proper categories from the list for her article, as shown in Figure 1.b.



```
ation'''  is an [[United States|American]] [[multinational corporation|multinational]] [[computer
ation]] with 76,000 employees in 102 countries and global annual [[revenue]] of [[United States d
00 (number)|billion]]. It develops, manufactures, licenses and supports a wide range of [[compute
 products for computing devices. Headquartered in [[Redmond, Washington|Redmond]], [[Washington]]
est selling products are the Microsoft Win
```

Target Pages	Anchor Text
Microsoft Windows	Microsoft Windows 98/NT 4.0/2000/XP
Microsoft Windows Services for UNIX	Microsoft Windows Services for UNIX
Microsoft Windows SDK	Microsoft Windows SDK (Software Development Kit)
Windows Help	Microsoft WinHelp
Windows XP Professional x64 Edition	Microsoft Windows XP 64-bit Edition for 64-bit Extended Systems
Windows Mobile	Microsoft Windows Mobile for Smartphone 2003 Second Edition
Windows XP	Microsoft Windows XP Professional
Windows Server 2003	Microsoft Windows Server 2003 R2
Windows 3.1x	Microsoft Windows 3.1
Windows 3.0	Microsoft Windows 3.0
WINE	wine maker
Winnipeg	Winnepeg
Winston Churchill	Winston Leonard Spencer Churchill
Windows Vista	Windows Vista Small Business
Win (baseball)	Wins

itions to EachWiki may be edited, altered, or removed by
hat you wrote this yourself, or copied it from a public dom
SION!

```
 Corporation'''  is an [[United St
[corporation]] with 76,000 employ
000000000 (number)|billion]]. It
tware]] products for computing de
, its best selling products are t
ity software. These products make
iginal mission of "a computer on
s footholds in other markets, wit
net portal]], and the [[Encarta|M
ardware]] products such as the [[
Xbox 360]], [[Zune]] and [[MSN TV

ounded to develop and sell [[BASI
tair 8800]], Microsoft rose to do
ompany released an [[IPO|initial
 price, has made four billionaire
company has been the target of cr
oth the [[United States Department
ght Microsoft to court for [[anti
er-centric business culture, Micr
 Web]], and awards [[Microsoft MV
```

Suggested Categories

Electronics companies of the United States	NVIDIA Honeywell Analogic Corporation Tektronix, Inc. SanDisk	✓
Multinational companies	Monsanto Tyson Foods Dresser Industries Savers 3M	✓
Companies established in 1976	Apple Inc. Apple Inc	☐
Video game companies	Nintendo Sony Gravity (company)	☐
Software companies	MYOB (software) Opera Software Symantec IKloak	☐
Companies based in Silicon Valley	Google NVIDIA Symantec SanDisk	☐
Companies listed on NASDAQ	Google NVIDIA Symantec DELL	☐
Computing by company	Nintendo NVIDIA Sony DELL	☐
Companies listed on the New York Stock Exchange	Monsanto Convergys Halliburton 3M	☐
Internet search engines	Google	☐
Dow Jones Industrial Average	Honeywell Caterpillar, Inc. 3M	☐
Agriculture companies of the United States	Monsanto Tyson Foods Deere & Company	☐

all contributions to EachWiki may be edited, a
nising us that you wrote this yourself, or copie
T PERMISSION!

Fig. 1. Snapshots of our prototype system. (a) is the link suggestion interface and (b) is the category suggestion interface.

The remainder of the paper is organized as follows. Section 2 gives the model of articles and queries as well as the similarity measurements for the proposed enhancements. Section 3 shows how this model can be applied to link and category suggestion tasks. Section 4 evaluates the prototype system in terms of effectiveness and efficiency. Section 5 discusses some related work. Section 6 concludes the whole paper.

2 The Model for the Proposed Enhancements

Our link and category suggestion algorithms are both based on RDF graph matching. This section shows how each Wikipedia article is modeled as a *resource graph* and how the inputs of the link and category modules are modeled as *query graphs*. A variety of similarity measurements are adopted to measure the semantic relevance between a query graph and a resource graph. This model can be applied to the link and category suggestion tasks, as will be shown in Section 3.

2.1 Wikipedia Article Model

To make the proposed enhancements effective and efficient, we need to find a representation for Wikipedia articles that is as succinct as possible, while still preserving the most useful semantic features. In the light of this motivation, we model each Wikipedia article as an RDF resource graph, as illustrated in Figure 2.

In Figure 2, oval nodes denote resource property values and rectangular nodes denote literal property values. Table 1 gives the descriptions and usage of the properties shown in Figure 2. Both the link and category suggestion modules use only a subset of these properties. Note that an article may have multiple values for each property.

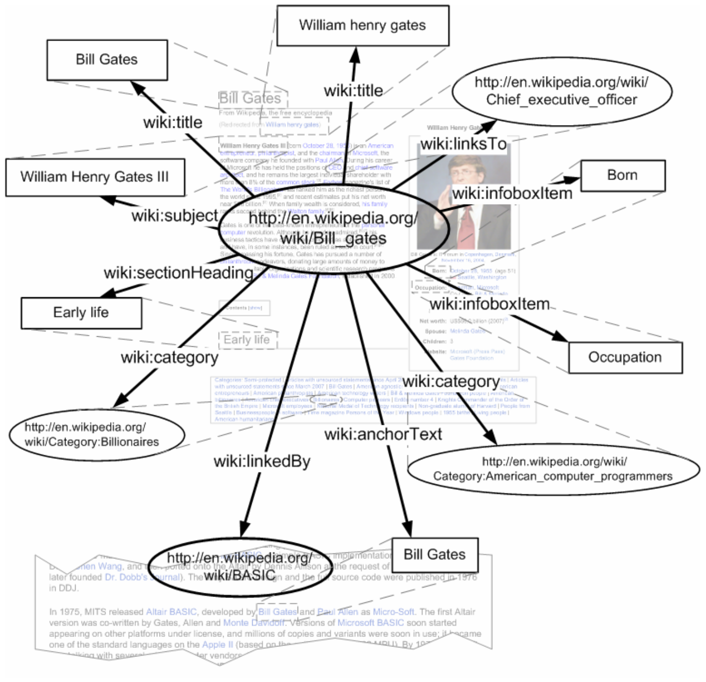

William henry gates
Bill Gates
wiki:title
wiki:title
http://en.wikipedia.org/wiki/Chief_executive_officer
wiki:linksTo
William Henry Gates III
Born
wiki:infoboxItem
wiki:subject
http://en.wikipedia.org/wiki/Bill_gates
wiki:sectionHeading
Early life
wiki:infoboxItem
Occupation
wiki:category
wiki:category
http://en.wikipedia.org/wiki/Category:Billionaires
wiki:anchorText
wiki:linkedBy
http://en.wikipedia.org/wiki/Category:American_computer_programmers
http://en.wikipedia.org/wiki/BASIC
Bill Gates

Fig. 2. Resource graph for an article in Wikipedia (only part of the property values are displayed)

Table 1. Descriptions and usage of Wikipedia article properties

Property Name	Value Description	Usage
title	The title of a Wikipedia article or the title of a redirect page that is redirected to the article	link suggestion
subject	A phrase marked in bold in the first paragraph of the article	link suggestion
anchorText	The displayed text of an link to the article	link suggestion
category	A category of the article	category suggestion
infoboxItem	An item name in the infobox	category suggestion
sectionHeading	The heading for a section of the article	category suggestion
linkedBy	Another Wikipedia article linking to the article	category suggestion
linksTo	Another Wikipedia article to which the article links	category suggestion

We select the properties of an article listed in Table 1 because they carry important semantic features of the topic concerned with the article.

The first three properties, *title*, *subject* and *anchorText*, usually summarize the topic of an article and are used by the link suggestion module. *Titles* usually sum up the topics of articles and most, if not all, of the existing Wikipedia search engines and authoring assistants, such as LuMriX[6], WikiWax[7] and Plog4U[8], support only prefix matching over Wikipedia article titles. In our model, however, titles of redirect pages of an article are treated as its *title* property values as well, and we also select the *subject* and *anchorText* properties to make better use of the shallow semantics in Wikipedia, since it is a Wikipedia convention to mark in bold the names of an article's subject when they are first mentioned in the article, and the displayed text of a link (i.e., the anchor text) usually provides the title or the alternative name of the link target [23]. Expanding the title of an article and taking the other two properties into consideration enable the link suggestion module to find the semantically relevant articles with titles completely different from the query phrase. For example, when the user types "William Henry G", the article on *Bill Gates* will be found and the user can replace the unfinished phrase with the auto-completed piped link "[[Bill Gates|William Henry Gates]]", which is beyond the capability of a simple prefix search over article titles.

The next five properties, *category*, *infoboxItem*, *sectionHeading*, *linkedBy* and *linksTo*, outline the structure of an article and are used by the category suggestion module. The basic idea of our category suggestion algorithm is to predict the input article's categories based on the categories of articles structurally similar to it, so *categories* of similar articles are candidate categories for the query article. Articles belonging to the same category usually share many common infobox items and similar section headings, and they often link to and are linked by some common articles. This is why we utilize the *infoboxItem*, *sectionHeading*, *linksTo* and *linkedBy* properties to find similar articles for category suggestion.

The resource graphs of all Wikipedia articles can be connected to form a huge RDF graph for the whole Wikipedia. In this paper, we limit our matching algorithm to the *Concise Bounded Descriptions*[9] (*CBD*) of articles, as shown in Figure 2, to make our modules efficient while still preserving most essential semantics in Wikipedia articles.

2.2 Query Model

In order to find articles most relevant to the input of either link or category suggestion module, we convert each input into an RDF query graph as shown in Figure 3.

A query graph is very much like a resource graph. The differences are as follows:

- The central node of a query graph is always a blank node, indicated by a question mark in Figure 3.
- Some properties in a query graph can be matched for arbitrary times while matching against resource graphs. This kind of properties is defined as *Cloneable Properties* and indicated by an asterisk in Figure 3.

Each of our query graph has only one blank node at the center because the query graph is used to find articles whose CBDs are similar to it; while the introduction of Cloneable Properties makes it possible to convert a phrase query into a query graph and effectively find articles similar to it, as will be shown in Section 3.

[6] http://wiki.lumrix.net/en/
[7] http://www.wikiwax.com/
[8] http://www.plog4u.org/index.php/Main_Page
[9] http://www.w3.org/Submission/CBD/

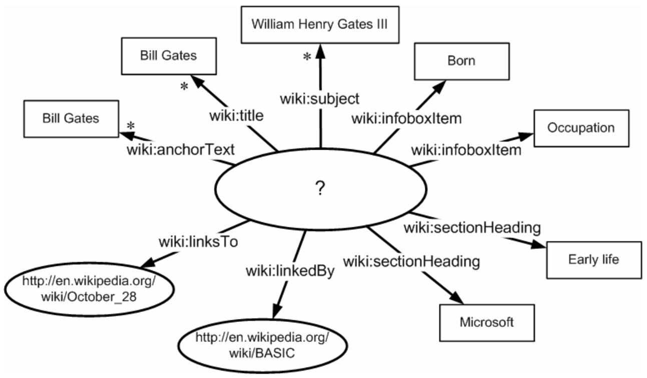


Fig. 3. A sample query graph

2.3 Similarity Measurements

Similarity between the query graph and a resource graph is derived from similarity between each query property value and the corresponding resource property value, and similarities between a pair of corresponding property values can be defined according to the need of a specific application. For our tasks, we use the following two kinds of similarities:

- **Similarity for Link Suggestion.** In order to let the user get the suggested links as early as possible, we adopt the *Prefix Similarity*, denoted by $Sim_{pre}(l_q, l_r)$, for the link suggestion task. Here l_q and l_r are literal property values of the query graph and the resource graph, respectively. This similarity presents whether l_q is a prefix of l_r, as shown in Equation (1).

$$Sim_{pre}(l_q, l_r) = \begin{cases} 1, & l_q \text{ is prefix of } l_r; \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

  The simplicity of Equation (1) makes the link suggestion module efficient enough for authoring time assistance.

- **Similarity for Category Suggestion.** In order to reduce noise, each property value indicating structural information of an article is treated as a whole and the similarity for the category suggestion is *Exact Similarity*, denoted by $Sim_{ex}(v_q, v_r)$, where v_q and v_r are corresponding property values (either literal or resource values) of the query and resource graph, respectively. This similarity is defined in Equation (2).

$$Sim_{ex}(v_q, v_r) = \begin{cases} \log(\frac{n}{df+1}) + 1, & v_q = v_r; \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

  Here n denotes the total number of articles in the dataset and df denotes the number of articles having v_q as the value for the property of concern. This similarity measurement is defined in the same spirit as the Term Frequency/Inverted Document Frequency metric widely accepted by the Information Retrieval (IR) academia and makes the category suggestion module effective.

To specify the importance of each property, it can be assigned a certain *weight*. The similarity between query graph *q* and resource graph *r*, denoted by *Similarity* (*q*, *r*), is defined as the weighted sum of the similarities of their corresponding property values, as shown in Equation (3).

$$Similarity(q,r) = \sum_{p \in P(q) \cap P(r)} w_p \cdot Sim_p(v_p(q), v_p(r)) \tag{3}$$

Here $P(g)$ denotes the bag of properties of graph g and $v_p(g)$ denotes the value of property p in g, and w_p is the weight for property p and can be set according to the importance of p in different applications. The similarity measurement for each property p, Sim_p, can be set to either Sim_{pre} or Sim_{ex}. Note that 1) for each Cloneable Property in the query graph, the weighted similarities between the only property value from the query graph and multiple property values from the resource graph are summed up; and 2) if for a given query property that is not Cloneable, there is more than one corresponding resource property value, only the one with the highest similarity is chosen to match the query property value.

2.4 Implementation

Despite the simplicity of our model, it is effective enough for the link and category suggestion tasks, as will be validated in Section 4. An additional advantage of our model is that the calculation and top-k ranking of similarity can be accelerated by utilizing an IR engine (Lucene[10] in our experiments) to index the property values for each article in advance. When implementing our prototype system, we index property values in the following way:

- Each article is mapped to a *document*;
- Each value is mapped to a *term*;
- Values of different properties are mapped to terms in different *fields*.

Note that each property value is treated as just one term to support the calculation of the adopted similarity measurements. For example, the article shown in Figure 2 is mapped to a document that has terms “Bill Gates” and “William henry gates” in its “title” field, term “http://en.wikipedia.org/wiki/BASIC” in its “linkedBy” field, etc.

When seeking similar articles with respect to a query, this index can be used to efficiently filter out articles with zero similarity scores, thus accelerating the process.

3 Applications of the Model

This section presents how the above model can be applied to the link and category suggestion tasks.

3.1 Link Suggestion

The link suggestion module takes the phrase just typed by the user as its input. It first converts the phrase into a query graph; and then it matches all the articles in the

[10] http://lucene.apache.org/

dataset against the query graph, getting a similarity score for each article; finally, all the articles are ranked according to their similarity scores and their corresponding wiki links are displayed to the user.

The query graph for a phrase contains three properties: *title*, *subject* and *anchorText*. They are all Cloneable Properties. For each of these properties, its value is set to the input phrase. Figure 4 shows an example of such a query graph.

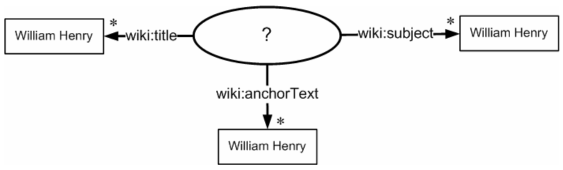


Fig. 4. Query graph for phrase "William Henry"

When matched against resource graphs of articles, all the three properties use the Prefix Similarity measurement. And the weights of these properties are set in reverse proportion to the average numbers of them for a Wikipedia article. The average numbers of values for properties *title*, *subject* and *anchorText* are 1.00, 1.03 and 22.92, respectively, so their weights are set to 23.61, 22.92 and 1.03, respectively.[11]

In order not to let the suggestion list pop up too many times and annoy the user, we set a threshold on similarity score and do not suggest links for stop words.

3.2 Category Suggestion

The process of category suggestion is divided into two steps. First, the article that needs categorizing is analyzed, converted into a query graph and matched against by resource graphs of the existing articles to find *k* most similar articles to it; then these *k* articles vote on the categories they belong to and decide the rank of the suggested categories. In our experiment, *k* is set to 200.

The query graph for a newly edited article contains four types of properties: *sectionHeading*, *infoboxItem*, *linkedBy* and *linksTo*, and it may have multiple values for each of them. Note that a newly edited article may have some incoming links because it may be written to fill the target of several broken links[12]. Figure 5 shows a sample query graph for an article.

When matched against resource graphs of existing Wikipedia articles, all the properties in the query graph use the Exact Similarity measurement. And the weights of these properties are set in reverse proportion to the average numbers of them for a Wikipedia article. The average counts of properties *sectionHeading*, *infoboxItem*, *linkedBy* and *linksTo* are 2.40, 1.41, 22.92 and 22.92, respectively, so their weights are set to 740.71, 1260.78, 77.56 and 77.56, respectively.

[11] The average number of values for the *title* property is very close to 1 because only a very small portion of the articles in our dataset have redirect links, and the average number of values for the *subject* property is above 1 because some articles have more than one phrase marked in bold in their first paragraphs.

[12] http://en.wikipedia.org/wiki/Wikipedia:Most_wanted_articles.

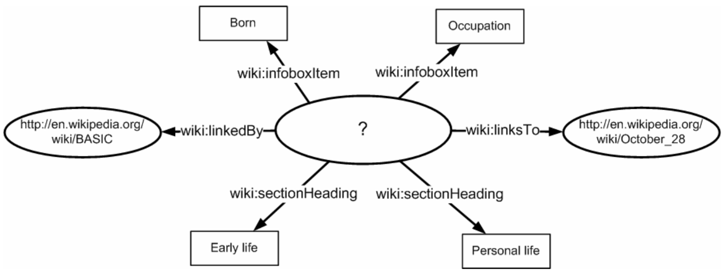


Fig. 5. Query graph for a newly edited article

After getting the similarity score of each existing Wikipedia article, the 200 articles with highest similarity scores are selected to decide the suggested category list by a vote. Each of these 200 articles votes only for the categories it belongs to, with certain amount of votes according to its similarity score. The amount of votes a category gets is shown in Equation (4), where *Vote*(*c*) denotes the amount of votes category *c* gets, *S* denotes the set of these 200 articles, *A*(*c*) denotes the set of articles that belong to category *c*, and *Similarity*(*q*, *a*) denotes the similarity between query article *q* and article *a* as defined in Equation (3).

$$Vote(c) = \sum_{a \in S \wedge a \in A(c)} Similarity(q,a) \tag{4}$$

Equation (4) is applied to all the categories that at least one of these 200 articles belongs to. Then these categories are ranked according to their *Vote* value and suggested for the user in descending order.

4 Experimental Results

4.1 Experiment Setup

Our experiments are all done on the WikipediaXML dataset snapshotted in early 2006 [14], which includes 659,353 articles, 113,483 categories and 15,119,272 links. We use the Java programming language to implement the prototype system. The system also utilizes the Dojo javascript toolkit[13] for the user interface and uses the Lucene API to index property values. All the experiments are run on a moderate PC with a 2.4GHz Intel P4 CPU and 2GB memory space.

4.2 Link Suggestion Results

To evaluate our link suggestion algorithm, we randomly extracted 576,941 links from the dataset, and use their displayed texts to construct queries. For each of these texts, the first term, the first two terms (if exist) and suchlike parts were fed to the link suggestion module, and the target of this link in the dataset was treated as the standard answer. These same queries were also fed to a prefix search engine over Wikipedia article titles for comparison.

[13] http://dojotoolkit.org/

In this way, totally 733,067 queries were constructed and tested. For each query, we recorded the ranks of the standard answer in the suggestion lists generated by the two systems. The results were shown in Table 2.

Table 2. Comparison of Link Suggestion and Prefix Matching

#query terms	#queries	Algorithm	ranked 1-5	ranked 6-10	ranked >10
1	576,941	Link Suggestion	**83.94%**	3.44%	12.62%
		Title Matching	25.30%	9.58%	65.12%
2	126,481	Link Suggestion	**90.90%**	1.18%	7.92%
		Title Matching	58.93%	4.04%	37.03%
$\geqslant$3	29,645	Link Suggestion	**83.22%**	3.20%	13.58%
		Title Matching	51.85%	2.56%	45.59%

From the data shown in Table 2, we could see that for most queries, the correct link targets were ranked among the top 5 suggestions by our link suggestion algorithm, exhibiting significant improvement over the prefix-matching-over-titles algorithm.

The efficiency of our algorithm was also evaluated. The average time cost for a popup was 0.089 seconds.

As mentioned in Section 2.1, one important characteristic of our link suggestion algorithm is that it can find articles semantically related to the query phrase but not containing the phrase as prefix of their titles. Table 3 gives some examples.

Table 3. Successful semantic findings by link suggestion

Query	Standard answer	Rank
William Henry Gates	Bill Gates	1
equerry	Master of the Horse	1
cathode	Electrode	4
1977 Election	Manitoba General Election, 1977	1

4.3 Category Suggestion Results

To evaluate the category suggestion algorithm, we randomly chose 3,670 existing articles, from the following 3 domains in the WikipediaXML dataset:

- *People* (e.g. Bill Gates, Isaac Newton, Michael Jordan);
- *Location* (e.g. Europe, United Kingdom, London);
- *Jargon* (e.g. Support Vector Machine, Web Ontology Language, PHP, Ontology).

The categories originally associated with these articles served as the ground truth, and the average numbers of categories for an article in the above domains are 3.96, 2.62 and 1.92, respectively. We fed these articles to our category suggestion module as well as a document classification system that treats articles as term weight vectors and measures similarity by the cosine of the angle between document and query vector. The only difference between these two systems is the similarity measurement adopted. The effectiveness of the two systems were measured by using metrics Mean

Table 4. Evaluation of category suggestion

Domain	#Query	Algorithm	MAP	P@1	P@2	P@5	R-prec
People	1,030	Category Suggestion	91.1%	88.0%	82.3%	62.3%	90.6%
		Document Classifier	63.6%	60.1%	55.0%	44.1%	61.4%
Location	1,066	Category Suggestion	91.8%	88.3%	78.9%	48.1%	90.3%
		Document Classifier	51.9%	50.1%	35.4%	19.2%	49.9%
Jargon	1,574	Category Suggestion	92.7%	88.3%	72.2%	37.3%	92.3%
		Document Classifier	89.1%	87.0%	63.0%	35.7%	87.3%
Overall	3,670	Category Suggestion	92.0%	88.2%	77.0%	47.5%	91.2%
		Document Classifier	71.1%	68.7%	52.7%	33.3%	69.2%

Average Precision (MAP), Precision at n (P@n) for n = 1, 2 and 5, as well as R-precision (R-prec), which are widely accepted metrics to evaluate retrieval performance. The results are shown in Table 4.

The data shown in Table 4 validate the effectiveness of our algorithm with the improvements over the existing document classification algorithm.

The efficiency of our system and the existing document classifier was also compared. The average time cost of category suggestion for an article was 0.355 seconds, and for the document classifier, the cost was the much longer 129.2 seconds.

Furthermore, scrutiny of the experimental results reveals several interesting findings:

- **Our algorithm is able to suggest *missing categories*.** For example, in the suggestion list for "United Kingdom", there exists a category named "Category:Island nations", whose evidences consist of "Republic of Ireland", "Australia", etc., but do not contain "United Kingdom" in the WikipediaXML dataset. This article is categorized into "Category:Island nations" (with a synonymous category name "Category:Island countries") in the current Wikipedia, which implies that our algorithm successfully found an error of *missing category* in the snapshot of Wikipedia in early 2006.
- **The algorithm is capable of discovering improper categorization.** For instance, in the experiment on "Web Ontology Language", our algorithm failed to suggest the category "Category:XML-based programming languages", which is a "correct" category for the article according to the ground truth. However, it is not proper to categorize Web Ontology Language (i.e. OWL) as an XML-based programming language. The elimination of this category assignment in the current Wikipedia appears to support our viewpoint.
- **The algorithm can categorize an article to the proper level of abstraction.** Take the article for "Support Vector Machine" as an example, our algorithm suggests both "Category:Machine Learning" and "Category:Artificial Intelligence", with the former ranked higher than the latter. By consulting the category hierarchy and the standard answer we found that "Category:Machine Learning" is subsumed by "Category:Artificial Intelligence" and the former should be associated with the article, which shows that our algorithm prefers to categorize an article to its immediate categories rather than their ancestors.

5 Related Work

As mentioned in Section 1, Wikipedia attracted much attention from the Semantic Web community due to its abundance, influence, high quality and well-structuring. For example, [2] proved that the URIs of Wikipedia articles are surprisingly reliable identifiers for ontology elements; [9] extracted topics and their semantic relations from Wikipedia; [1] presented an experiment to automatically annotate several semantic relationships in Wikipedia; [4] extended Wikipedia with typed links; [10] discovered that links in Wikipedia could provide useful training examples for a named entity disambiguator; [3] presented measures for automatic filtering of strong semantic connections between Wikipedia categories; and [7] provided an extension to be integrated in Wikipedia that allows the typing of links between articles and the specification of typed data inside the articles in an easy-to-use manner.

Our work is mainly motivated by [12] and [13], which pointed out that Wikipedia's authoring interface was not so convenient and smart as had been declared and that Wikipedia only had a relatively regular and small number of community members to take charge of most revision work. We hope to change such a situation.

Research on Wikipedia Modeling and RDF graph matching is closely related to our work. [8] used several features of Wikipedia articles to extract concepts and recognize hierarchical relations between them, and [16] used a specific ontology to integrate Wikipedia into the Semantic Web framework and proposed an RDF graph representation for Wikipedia articles. Our work adapts their models to our tasks by recognizing different properties for Wikipedia articles to better utilize shallow semantics in Wikipedia. [19] introduced a flexible framework for computing semantic similarity between objects represented as RDF graphs. We augment their framework with *Cloneable Properties* for our link suggestion task and define our own similarity measurements between corresponding property values. [21] presented an algorithm for analyzing general logical graphs to compute similarity scores between nodes based on their structural contexts, but the algorithm do not support weighted edges. [20] proposed an approach for semantic search by matching conceptual graphs. We simplify their approach for our real-time tasks. There are other studies related to graph matching in a more general sense. For example, [18] presented a graduated assignment algorithm for graph matching, and [17] presented the Similarity Flooding algorithm for directed labeled graph matching. Their work concentrated on exploring mapping between corresponding nodes of two graphs and do not deal with weighted edges.

Not much work has been done on the task of link suggestion in the Wikipedia environment. [6] addressed the problem of discovering missing links in Wikipedia. Their work concentrated on improving the existing link structure of Wikipedia and was not integrated into the Wikipedia authoring interface. Other related work includes several Wikipedia search engines, which only support prefix search over article titles, as mentioned in Section 2.1.

A straightforward solution to the Wikipedia category suggestion task is to use the vector space model [24, 25] to represent Wikipedia articles as term weight vectors, measure similarity between two articles by the cosine of the angle between their vectors, and suggest categories with the *k*-NN method [26]. Our evaluation showed that this solution was far from satisfactory in terms of effectiveness and efficiency.

6 Conclusion

In this paper, we use an RDF-based model to represent elements in Wikipedia authoring, and propose two enhancements of the current Wikipedia based on this model to facilitate Wikipedia authoring. Our main contributions can be summarized as follows:

- The proposal of one possible solution for improving Wikipedia authoring interface and reducing the burden of Wikipedia editors, as well as a prototype system based on the proposed enhancements;
- The proposal of a model that extracts semantic information of Wikipedia articles in a succinct way and supports different kinds of queries, along with the recognition of several important semantic properties of Wikipedia articles;
- The proposal of a flexible similarity measurement between query and resource based on the idea of RDF graph matching for effective and efficient accomplishments of the link and category suggestion tasks.

In the future, we will do more user studies and further improve the effectiveness and usability of our system according to feedbacks. We will also suggest other Wikipedia elements such as template for the user and apply our method to Semantic Wikipedia [7] to provide relation and attribute suggestion.

References

[1] Ruiz-Casado, M., Alfonseca, E., Castells, P.: From Wikipedia to Semantic Relationships: a semi-automated Annotation Approach. ESWC (2006)
[2] Hepp, M., Bachlechner, D., Siorpaes, K.: Harvesting Wiki Consensus - Using Wikipedia Entries as Ontology Elements. SemWiki (2006)
[3] Chernov, S., Iofciu, T., Nejdl, W., Zhou, X.: Extracting Semantic Relationships between Wikipedia Categories (2006)
[4] Krötzsch, M., Vrandečić, D., Völkel, M.: Wikipedia and the Semantic Web: the Missing Links (2005)
[5] Voss, J.: Collaborative thesaurus tagging the Wikipedia way. Wikimetrics (2006)
[6] Adafre, S.F., de Rijke, M.: Discovering Missing Links in Wikipedia. LinkKDD (2005)
[7] Völkel, M., Krötzsch, M., Vrandecic, D., Haller, H., Studer, R.: Semantic Wikipedia. In: WWW 2006 (2006)
[8] Kozlova, N.: Automatic Ontology Extraction for document classification. In: Weikum, G. (ed.) A thesis submitted in conformity with the requirements for the degree of Master of Science (2005)
[9] Kinzler, D.: WikiSense: Mining the Wiki. Wikimania (2005)
[10] R. Bunescu. Using Encyclopedic Knowledge for Named Entity Disambiguation. EACL 2006 (2006)
[11] Giles, J.: Internet encyclopaedias go head to head. Nature 438(7070), 900–901 (2005)
[12] Swartz, A.: Raw Thought: Who Writes Wikipedia? (2006), http://www.aaronsw.com/weblog/whowriteswikipedia
[13] Swartz, A.: Raw Thought: Making More Wikipedians (2006), http://www.aaronsw.com/weblog/morewikipedians

[14] Denoyer, L., Gallinari, P.: The Wikipedia XML Corpus. ACM SIGIR Forum 40(1) (June 2006)
[15] Zhang, Y.: Wiki Means More: Hyperreading in Wikipedia. In: HT 2006 (2006)
[16] Harth, A., Gassert, H., O'Murchu, I., Breslin, J., Decker, S.: Wikiont: An ontology for describing and exchanging wikipedia articles. Wikimania 2005 (2005)
[17] Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching. In: ICDE 2002 (2002)
[18] Gold, S., Rangarajan, A.: A graduated assignment algorithm for graph matching. In: TPAMI 1996 (1996)
[19] Oldakowski, R., Bizer, C.: SemMF: A Framework for Calculating Semantic Similarity of Objects Represented as RDF Graphs. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
[20] Zhong, J., Zhu, H., Li, J., Yu, Y.: Conceptual Graph Matching for Semantic Search. In: Priss, U., Corbett, D.R., Angelova, G. (eds.) ICCS 2002. LNCS (LNAI), vol. 2393, Springer, Heidelberg (2002)
[21] Jeh, G., Widom, J.: SimRank: a measure of structural-context similarity. In: KDD 2002 (2002)
[22] Zlatic, V., Bozicevic, M., Stefancic, H., Domazet, M.: Wikipedias: Collaborative web-based encyclopedias as complex networks. In: arXiv 2006 (2006), http://arxiv.org/pdf/physics/0602149
[23] Gabrilovich, E., Markovitch, S.: Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge. In: AAAI 2006 (2006)
[24] Salton, G., Lesk, M.E.: Computer evaluation of indexing and text processing. Journal of the ACM 15(1), 8–36
[25] Salton, G.: The SMART Retrieval System - Experiments in Automatic Document Processing. Prentice-Hall, Englewood Cliffs (1971)
[26] Mitchell, T.M.: Machine Learning. McGraw-Hill, Boston, MA (1997)
[27] Liu, T-Y., Yang, Y., Wan, H., Zeng, H-J., Chen, Z., Ma, W-Y.: Support Vector Machines Classification with A Very Large-scale Taxonomy. SIGKDD Explorations 7(1), 36–43