International Journal of Intelligence Science, 2015, 5, 196-207
Published Online October 2015 in SciRes. http://www.scirp.org/journal/ijis
http://dx.doi.org/10.4236/ijis.2015.55017

DBpedia-Based Fuzzy Query Recommendation Algorithm and Its Applications in the Resource-Sharing Platform of Polar Samples

Wenfang Cheng[1,2*], Qing'e Wu[3,4], Xiao Cheng[5], Jie Zhang[1,2], Zhuanling Song[6]

[1]Polar Research Institute of China, Shanghai, China
[2]Key Laboratory for Polar Science, State Oceanic Administration, Shanghai, China
[3]School of Computer Science, Fudan University, Shanghai, China
[4]College of Electric and Information Engineering, Zhengzhou University of Light Industry, Zhengzhou, China
[5]College of Global Change and Earth System Science, Beijing Normal University, Beijing, China
[6]The First Institute of Oceanography, SOA, Qingdao, China
Email: [*] chengwenfang@pric.org.cn

Received 5 August 2015; accepted 9 October 2015; published 12 October 2015

Abstract

In order to continuously promote the polar sample resource services in China and effectively guide the users to access such information as needed, a fuzzy algorithm based on DBpedia has been proposed through the analysis of the characteristics of the query recommendations in search engines, namely, to search similar entry queues by constructing a DBpedia category tree, then use the fuzzy matching algorithm to work out the entry similarity, and then present the example query applications of this algorithm on the resource-sharing platform of polar samples. Comparing the traditional literal character matching method and DBpedia semantic similarity algorithm, the experimental results show that the fuzzy query algorithm based on DBpedia features has a higher search accuracy rate, stronger anti-interference capability, and more flexible algorithm use by virtue of its fuzzy weight adjustment.

Keywords

Search Engine, Fuzzy Query, Semantic Similarity, Wiki

[*]Corresponding author.

How to cite this paper: Cheng, W.F., Wu, Q.E, Cheng, X., Zhang, J. and Song, Z.L. (2015) DBpedia-Based Fuzzy Query Recommendation Algorithm and Its Applications in the Resource-Sharing Platform of Polar Samples. *International Journal of Intelligence Science*, **5**, 196-207. http://dx.doi.org/10.4236/ijis.2015.55017

1. Introduction

As one of the most important modules of information platforms, each search engine has its query recommendation almost as a standard function for its search module. In particular, when the users are not clear about their search objects, the relevant search results given by the search engine can effectively guide the users to gradually get access to the information they need [1]. During the past decade, the query recommendation technology has obtained fairly satisfactory results in e-commerce platforms; therefore, in order to improve the value of scientific data, this paper examines the application of the query recommendation technology on resource-sharing platform of polar samples (BIRDS). It provides information on major samples collected in the polar regions during previous Chinese polar expeditions with large numbers of valuable samples available, including snow, ice cores, flora and fauna, meteorites, sediments, and rocks. Since the establishment of BIRDS in 2006, it has released 7058 resources, 12,279 pictures and 6.4 GB data. On average, every day it is visited by 895 users from 187 countries with 112 million clicks and 16.6 TB content downloads. However, most users came from the cooperated institutes of BIRDS, and the average PV (pages view) per day per user normally was less than 20. In order to further promote the polar sample resource services in China, to provide more samples and relevant information to scientists who have made an attempt to carry out related polar research but without access to filed study, and to improve the utilization efficiency of polar sample resources, this paper has designed a fuzzy query recommendation algorithm based on Wikipedia knowledge database and applied it to BIRDS [2].

Scientific data platforms have a relatively smaller number of users than e-commerce platforms and thus have a simpler user log; thereby this paper presents the query recommendation [3] based on contents. In addition, this paper has fully considered the fact that search engine is one of the most important factors of information platforms, as well as the character or word association characteristics [4] to have proposed the DBpedia-based fuzzy query algorithm, namely, to search similar entry queues by constructing a DBpedia category tree, then use the fuzzy matching algorithm to work out the entry similarity, and then present the example query applications of this algorithm on BIRDS.

In the past 10 years, many scientific data platforms have offered query recommendation services which received good effect, but this fuzzy query recommendation algorithm based on DBpedia is first presented and applied on scientific data platforms especially on polar data sharing systems. After the application on BIRDS, the platform's daily users increased to thousands, and the daily PV per user was stabled at about 15. More platform visit data show the algorithm proposed better effect than normal query recommendation algorithm.

2. Algorithm of Category Tree on a Knowledge Database

2.1. Construction of Semantically Similar Category Tree

The construction of a knowledge database is a complicated task involving many disciplines. The associated database [5]-[7] constructed in this paper based on DBpedia has the following characteristics:

- Its entries are based on Wikipedia, and are able to develop along with changes of Wikipedia.
- It has extracted structured data from Wikipedia and converted such data into the form of Linked Data.
- Its knowledge database uses the form of ontological construction to organize the entries.
- It has an open API interface, enabling the machine to understand such structural data.

The DBpedia body mainly consists of four types of system structures [8]:

(1) "part-of" relationship, indicating the relationship between the part and the whole of a concept.

(2) "kind-of" relationship, indicating an integrated relationship between concepts.

(3) "instance-of" relationship, indicating the relationship between the concept and its instances.

(4) "attribute-of" relationship, indicating that one concept is the attribute of another concept.

The method in this paper only uses the relationship between "concepts" and "categories" to construct a category tree. For instance, in Chinese Wikipedia, the free encyclopedia:

Skua category: Gull passerine|Stercorariidae.

Stercorariidae category: Gull passerine.

Gull passerine category: Birds|Charadriiformes.

Emperor penguin category: IUCN threatened species|Antarctica|Spheniscidae.

Spheniscidae category: birds|Sphenisciformes|Spheniscidae.

Sphenisciformes category: birds|Sphenisciformes|Spheniscidae.

Finally a knowledge database as below can be obtained:

Skua -> Stercorariidae -> Gull passerine -> Birds.

Emperor penguin -> Spheniscidae -> Sphenisciformes -> Birds.

In the process of the category tree extraction, it is particularly important to set the height of the category tree; if it is too high, the traversal speed will be influenced; if too low, the matching effect will be reduced.

2.2. Construct a Proper Height for the Category Tree

A category tree's error rate will be quite high if it is constructed too small. On the other hand, if the tree is too big, the apparent error rate obtained by means of learning set test is very small, but its true error rate may still be relatively large. Therefore, we need to construct a tree of a proper/appropriate size to minimize the real error rate.

The purpose/aim of decision tree learning is to obtain a simple tree with a strong predictive capacity. When the tree is in its full growth, its predictive capacity will be reduced. In order to solve this problem, we need to obtain a tree of the proper/appropriate size. In generally, there are two methods available.

Method-1: Define the conditions that the tree will stop growing.

1) Partition the number of instances to the minimum. When the size of the data subset corresponding to the current node is smaller than the number of specified minimum partition instances, no further partition is needed even though they do not belong to the same category.

2) Partition threshold value. When the difference between the value obtained by means of the applied partition method and the value of its parent node is smaller than the specified threshold value, no further partition is needed.

3) Maximize tree depth. When the length of further partition will exceed the maximal tree depth, stop partitioning.

Method-2: Carry out pruning after a complete decision tree is generated by evaluating subtrees. The entire decision trees will perform better if a subtree is removed, then the subset is pruned. Specifically, the implementing process in the Breiman CART [9] are as follows:

1) Tree construction

The decision tree is made up of the data sets partitioned by attribute values, and thereby needs to define the measurement partitioned by attribute, namely, according to this measurement, the optimal partitioning attributes for current data subset can be worked out.

When the fuzzy function of calculation cost for node has been selected, during the process of the tree growth, we are always trying to find an optimal bifurcation value to partition the samples in the node, so that the cost could be minimized. The fuzzy function $\phi(P)$ is used to represent the fuzzy degree of the tree node t or error partition index, namely:

$$E(t)=\phi(D)=-\sum_{i\neq l} p_i p_l = 1-\sum_{i=1}^{c} p_i^2. \tag{1}$$

Here, $D=\{p_1, p_2, \cdots, p_c\}$ is a decision set, c denotes the number of the decision-making categories in the decision set, $p_i \geq 0$ indicates the proportion of the i^{th} decision-making category in the decision set D and $\sum_{i=1}^{c} p_i = 1$.

In the bifurcation tree of the CART algorithm architecture, the amount of changes of fuzzy degree due to bifurcation is as follows:

$$\Delta E(t)=E(t)-p_l E(t_l)-p_r E(t_r) \tag{2}$$

where, t is the bifurcation node; $E(t)$ is the fuzzy degree of the node t; $E(t_l)$ and $E(t_r)$ are the fuzzy degree of the left and right bifurcation node, respectively; p_l and p_r denote the percentage of the node t in the left and right bifurcation samples, respectively. For bifurcation of each internal node t, take the largest change of fuzzy degree in all possible bifurcation ways of t. For other nodes, repeat the same search process.

2) Pruning

The large scale trees are generated by the above algorithm and its apparent error rate is very small, but its true

error rate may still be relatively large. We must construct a tree with a small true error rate by means of the pruning technique. We use a certain algorithm to prune the branches of this tree continuously. During the pruning process, we will obtain a list of decresing trees to form a sequence of pruned trees, and each tree in this sequence will have a smaller apparent error rate [9] compared with other subtrees of the same size, and then we can conclude that this sequence is an optimal one. The bifurcation tree can be pruned on the basis of the minimal cost complexity principle as below:

In general, a tree can be expressed by *T*, the subtree with the root node of *t* is expressed by T_t, then the pruned subtree T_{t3} will shrink into a terminal node *t*3, the pruned tree can be expressed as $T-T_{t3}$, and there is the $T-T_{t3} \subset T$, which is the subset of *T*. Use $\overline{T}$ to express the terminal node set in the trees *T*, and the number of the corresponding terminal nodes is $\left|\overline{T}\right|$. The impurity index of the tree *T* is defined as follows:

$$E(T)=\sum_{t\in\overline{T}} E(t). \tag{3}$$

$E(t)$ denotes the fuzzy index of the tree node *t* or the square error of the fitting node data set of the node *t* in the Equation (3), and the error index is the fuzzy function $E(t)$.

The pruning principle of the decision tree, namely, the cost complexity measurement is displayed as:

$$E_a(T)=E(T)+a\left|\overline{T}\right| \tag{4}$$

where $E_a(T)$ denotes a linear combination of the tree impurity index cost $E_a(T)$ and its complexity. Therein, *a* is the complexity parameter resulted from the complexity of a tree and $\left|\overline{T}\right|$ indicates the number of the terminal nodes for the tree *T*.

To find the next smallest tree of the tree *T*: For each internal node t of the tree *T*, we need to work out the value *a* of the penalty factor for the next tree $T-T_t$ wrongly partitioned, and label the value as a_t, which is the ratio between the change amount of the error index before and after the current tree is pruned and the change of the terminal node number:

$$a_t=\frac{E(t)-E(T_t)}{\left|\overline{T_t}\right|-1}. \tag{5}$$

The node we need to select is an internal node with minimal a_t. The whole tree pruning process is to calculate a_t, then seek the smallest a_t, and then select $T-T_t$ as the next pruning object.

For each given value *a*, a smallest tree $T(a)$ can always be found based on the corresponding to the measurement of its cost complexity:

$$E_a(T(a))=\min_{T\subset T\max} E_a(T). \tag{6}$$

When the value *a* increases, $T(a)$ always remains smallest until it reaches a jump point a', and then the tree $T(a')$ becomes a new smallest tree.

After the smallest tree $T(a)$ is determined, its height can be defined as $h=n(t_f)-n(t_0)+1$, where $n(t_f)$ is the number of layers of the final leaf nodes while $n(t_0)$ is the number of layers of the root nodes.

For such cases in this paper, we can work out the appropriate height of a tree as $h=5$ according to the above algorithm.

3. Similarity Algorithm

The similar degree of traditional category tree is mainly calculated through the following two methods: character direct search method and vector included angle cosine calculation method. However, these two methods are oversimplified and thus the similarity is seriously affected. Therefore, based on the character search method, this paper has proposed a fuzzy query algorithm based on DBpedia.

3.1. Literal Character Matching Method (CCQ)

Literal character matching method is the easiest method, it judges the similarity by the proportionality of com-

mon words on two words. For example, there is one common word between *adelie penguins* and *emperor penguin*, then the matching value is 0.5.

Algorithm:

First, get all samples' name from database, named $Word_1$, $Word_2$, … $Word_i$, …, $Word_n$;

Second, calculate each correlation for $word_i$, and the value is correlation [$word_j$, $word_i$] = Max (length (common words ($word_j$, $word_i$))/lengh ($word_i$), length (common words (Pinyin ($word_j$), Pinyin ($word_i$)))/lengh (Pinyin ($word_i$))).

3.2. Fuzzy Query Algorithm (WIKIFQ)

If the search contents are classified according to such attributes of a word or a phrase as pronunciation, meaning and relevance, refer to the concrete fuzzy query algorithm [10].

1) Classification of the query contents

Firstly, the samples to be queried are classified appropriately. The query criteria are: the smaller distance between the example that belongs to a certain category and the center within the category is the better; the larger distance that is from the center among categories is the better. According to the attributes of each category, the average value of each category is calculated as the category center $v_i, i = 1, 2, \cdots, n$. Assume u_{ik} is the membership function of the kth sample to the ith category, $0 \le u_{ik} \le 1$ and $0 < \sum_{k=1}^{N} u_{ik} < N$, $U = \{u_{ik}\}$. Assume $d_{ik} = \|x_k - v_i\|$ is the distance between the sample x_k and the center v_i of the ith category, and $m > 1$ is a fuzzy weighted exponent. Define the distances within a category and among categories, to make the distances satisfy that the distance is from the center within the category is smaller, and the distance is from the center among categories is larger.

Define the distance within a category

$$J_m(U_i, v_i) = \sum_{j=1}^{N_i} u_{ij}^m d_{ij}^2. \tag{7}$$

Define the distance among categories

$$J_m\left(U_{\setminus\{U_i\}}, v_i\right) = \sum_{i=1}^{n} \sum_{j=1}^{N-N_i} u_{ij}^m d_{ij}^2. \tag{8}$$

Synthesize the Equations (7) and (8) to define the objective function $J_m(U, n)$ as

$$J_m(U, n) = \sum_{i=1}^{n} \sum_{j=1}^{N} u_{ij}^m d_{ij}^2 \tag{9}$$

where, $U_i = \{u_{ij}\}$ is applicable to a certain i, while $U = \{u_{ij}\}$ is applicable to all i.

An objective finally needs to be classified into a certain kind of problems according to a certain membership degree, so the objective function satisfies a certain constraints as follows:

$$\sum_{i=1}^{n} u_{ij} = 1, \forall 1 \le j \le N. \tag{10}$$

According to the objective function, the following conditions should be met: 1) the defined u_{ij} should be inversely proportional to d_{ij}, namely, u_{ij} is a monotone decreasing function about d_{ij}. 2) u_{ij} is a monotone increasing function about the fuzzy weighted exponent m. 3) u_{ij} is the membership degree, so $0 \le u_{ij} \le 1$. Moreover, it requires that each category must contain one sample at least, but the sample may not belong to the same category, so $0 < \sum_{j=1}^{N} u_{ij} < N$ is true. 4) Simultaneously, u_{ij} satisfies the Equation (10).

According to 1)-4), u_{ij} can be defined as follows:

$$u_{ij} = \left\{ \sum_{k=1}^{n} \left(\frac{d_{ij}}{d_{kj}} \right)^{\frac{2}{m-1}} \right\}^{-1}. \tag{11}$$

It can be proved that the Equation (11) satisfies the conditions 1)-4).

Under the constraint (10), the minimal value of the Equation (9) can be obtained by iterating repeatedly the Equation (11) to determine the final u_{ij}.

Based on u_{ij}, assume the center of each category is v_i, and it can be calculated as follows:

$$v_i = \frac{\sum_{j=1}^{N} \left(u_{ij}\right)^m x_j}{\sum_{j=1}^{N} \left(u_{ij}\right)^m}. \tag{12}$$

2) Character matching query

For a sample *x* to be queried, calculate the distance between *x* and the center v_i, select *k* characters or words that is closest to *x*, which are represented by $x_1, x_2, \cdots, x_k$ respectively. Define an ordered pair $\langle x, f(x) \rangle$, and $f(x)$ is the category which the sample *x* belongs to: $f(x): R \to W$, in which *R* is a set of samples to be queried, *W* is a finite set $\{w_1, w_2, \cdots, w_n\}$ of category, w_i is the *i*th type content of the partition query content, and *n* is the number of categories. Then $f(x)$ can be calculated as follows:

$f(x) = \arg\max_{w_j \in W} \sum_{i=1}^{k} \sigma\left(w_j, f(x_i)\right)$, where $\sigma\left(w_j, f(x_i)\right) = \begin{cases} 1, & f(x_i) = w_j \\ 0, & f(x_i) \neq w_j \end{cases}$. If $f(x) = w_{j^*}$, then the sample *x* belongs to the category w_{j^*}.

The above expressions can also be written as: $S_j = \sum_{i=1}^{k} \sigma\left(w_j, f(x_i)\right)$, here $f(x) = \arg\max_j S_j$, then *x* belongs to the category w_{j^*}, $j = 1, 2, \cdots, n$.

3.3. Selection of the Fuzzy Weighted Exponent *m*

For the Equation (11), when $m \to 1$, each u_{ij} in the Equation (11) satisfies $u_{ij} \to 0$ or 1, and when $m = 1$, there is no weighted value u_{ij}; when $m \to +\infty$, each u_{ij} satisfies $u_{ij} \to 1/n$, and the partition is most fuzzy at this time. It is clear that the exponent *m* directly determines the fuzziness of the classification results.

To classify the search contents, we can use the fuzzy method and algorithm [11]-[13] of target recognition researched by BEZDEK, Lin Qing, Wei Meia and others. The fuzzy degree of classification is defined as follows:

$$F_m(U, n) = \frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{N} \left| u_{ij} - \left(u_{ij}\right)_{0.5} \right| \tag{13}$$

where, BEZDEK [11]

$$\left(u_{ij}\right)_{0.5} = \begin{cases} 1, u_{ij} \geq 0.5 \\ 0, u_{ij} < 0.5 \end{cases} \tag{14}$$

A fuzzy decision-making problem is formed by the intersection of a given fuzzy target G_f and a fuzzy constraint C_f, *i.e.*,

$$D = G_f \cap C_f.$$

In this paper, the fuzzy object of a decision-making problem that a keyword or a word is queried is defined as follows:

$$G_f = \min\left\{ J_m\left(U^*, n\right) \middle| m \in (1, +\infty) \right\} \tag{15}$$

where, U^* is the set of the final u_{ij} determined when the Equation (9) reaches a minimal value. In addition, while completing the content fuzzy classification, this algorithm also requires that the content should be partitioned as clearly as possible in order to correctly distinguish the category membership of each sample. Therefore, the selection of the parameter *m* is subject to another constraint, namely, the selected value can not make the classification results of the fuzzy classification algorithm overly fuzzy. The partition fuzzy degree is a good measurement to evaluate fuzzy classification to partition fuzziness. As a result, the fuzzy constraint of the decision that the parameter m is preferred is as follows:

$$C_f = \min\left\{F_m\left(U^*, n\right) \middle| m \in \left(1, +\infty\right)\right\}. \tag{16}$$

When G_f and C_f are treated as fuzzy sets, they can be characterized by their membership functions respectively. In order to ensure that the membership functions of the fuzzy object G_f and the fuzzy constraint C_f have the same increasing or decreasing extent, the membership functions of G_f and C_f can be defined respectively as follows:

$$\mu_G\left(m\right) = \left(\frac{J_m\left(U^*, n\right)}{\max\limits_{\forall m} J_m\left(U^*, n\right)}\right)^3 \tag{17}$$

$$\mu_C\left(m\right) = F_m\left(U^*, n\right) \tag{18}$$

The membership function of fuzzy decision can be expressed as $\mu_D\left(x\right) = \min\left\{\mu_G\left(x\right), \mu_C\left(x\right)\right\}$, and the final decision-making result is the solution to satisfy $\mu_D\left(x^*\right) = \max\limits_{\forall x}\left\{\mu_D\left(x\right)\right\}$.

Consequently, the optimal weighted index m^* is the *m* value corresponding to the maximum membership degree of the intersection of fuzzy subsets corresponding to the fuzzy object and fuzzy constraint, respectively. The optimal weighted index m^* can be obtained by the following formula:

$$m^* = \arg\limits_{\forall m}\left\{\max\left\{\min\left\{\mu_G\left(m\right), \mu_C\left(m\right)\right\}\right\}\right\}. \tag{19}$$

The m^* obtained based on the Formula (19) will be able to ensure that the classified objective function and the classified partition fuzziness could be minimized by a larger membership degree, so that the fuzzy classification achieved by the fuzzy classification algorithm could not only express the similar information among samples, but also ensure the clarity of the sample classification. Therefore, a corresponding better fuzzy classification result will be obtained.

4. Experiment

4.1. Data Processing

1) Construct a DBpedia Database

A data rather large in amount cannot be indexed or retrieved if placed directly in a document, so a MYSQL database should be built. Download the XML dump file of 2013 November from Chinese Wikipedia, extract the three files to get zhwiki-latest-categorylinks.sql, zhwiki-latest-pages.sql and zhwiki-latest-redirect.sql (with the total of 1.34 GB), and import such files into the DBpedia Chinese entry database already obtained with approximately 3,102,000 page records, 315,000 category records and 7,736,000 categorylinks records.

2) Get *N* entries from the database of BIRDS [14]. *N* = Random (50 - 100)

3) Extract the category tree from DBpedia and then form a weight matrix [15] [16]. The code for a category tree of an individual entry is shown in **Figure 1** and **Figure 2**.

4) Superimpose the fuzzy matching algorithm to get similarity

In the experiment, take the fuzzy weighted exponent $m = 1.75$.

5) Test environment

CPU: 2.5 GHZ × 2 core

Memory: 4.0 GB

OS: Windows 7 - 64 bit

```
#Get category by name
def getCategorysByName(name, printable=False):

    if printable:
        print 'getCategorysByName'
        print u'%s' %name
    cursor = conn.cursor()

    cursor.execute('SELECT page_id, page_is_redirect FROM page where page_namespace=14 and page_title=%s ' , (name))
    page = cursor.fetchone()

    if not page:
        cursor.execute('SELECT page_id, page_is_redirect FROM page where page_namespace=0 and page_title=%s ' , (name))
        page = cursor.fetchone()
        if not page:
            return None
    if page['page_is_redirect']:
        cursor.execute('SELECT rd_title FROM redirect where rd_from=%s ', (page['page_id']))
        redirect = cursor.fetchone()
        if not redirect:
            return None
        return getCategorysByName(redirect['rd_title'], printable)

    categorys = []
    cursor.execute('SELECT cl_to FROM categorylinks where cl_from=%s ', (page['page_id']))
    rows = cursor.fetchall()
    for row in rows:
        if printable:
            print ' ', row['cl_to'].decode('utf8')
        categorys.append(row['cl_to'].decode('utf8'))
    cursor.close()
    return categorys
```

Figure 1. Get category by entry name.

```
#Get categroy tree, the depth <= 5, result is stored in out which is a dictionary.
def getCategoryTree(name,  out={}, depth=0, printable=False):
    if depth == 0:
        if printable:
            print 'getCategoryTree'
            print u'0 %s' %name
        out[name] = 1
    if depth>3:
        return
    depth += 1
    categorys = getCategorysByName(name)
    if not categorys:
        return
    for category in categorys:
        if repr(name) == repr(category):
            continue
        if printable:
            print ' '*depth, depth,category
        if not out.get(category):
            out[category] = 5-depth
        getCategoryTree(category, out, depth, printable)
```

Figure 2. Get category list.

4.2. Experimental Results

In order to verify the efficiency of this algorithm, we compare it with the traditional literal string matching algorithm and DBpedia semanteme-based algorithm (**Table 1**). The results indicate that the fuzzy matching algorithm is more accurate than other algorithms in terms of semantic analysis, and is even capable of detecting certain relationships between two seemingly different words.

4.3. Discussion

From the test result shown in **Table 1**, we can conclude that **WIKIFQA** method can detect similar data more efficiently and more accurately. After the application of the WIKIFQA on 2013 years, users have received more convenient service, and the value of PV, IP increased also (**Figure 3**).

But the algorithm is dependent on wiki database too much, and its accuracy is mostly affected by the quantity and quality of Wikipedia pages. To improve the accuracy, we have to update the database quarterly automatically.

5. Conclusion

This paper provides a DBpedia-based fuzzy query algorithm and gives the feature extraction method for fuzzy algorithm eigenvalue and implementation of semantic matching algorithm based on the analysis of the characteristics of the polar sample data and Wikipedia Chinese data. The experimental results show that this WIKIFQA

Table 1. Comparison of WIKIFQA with CCQ and WIKIQA algorithms.

Word pairs	Literal character matching method (CCQ)	DBpedia semantic algorithm (WIKIQA)	Fuzzy matching algorithm based on DBpedia (WIKIFQA)
Skua-Skue	0.5000	0.0000	1.0000
Skua-Penguin	0.0000	0.2371	0.2371
Stratosphere-Troposphere	0.6667	0.8899	0.8899
Stratosphere-Mesosphere	0.3333	0.8899	0.8899
Stratosphere-Planetary Boundary Layer	0.3333	0.0000	0.0000
Cinerite-Sulfate	0.0000	0.0624	0.0624
Cyclone-Drought	0.0000	0.1346	0.1346
Lightning-Windstorm	0.0000	0.3008	0.3008
Frost-Fog	0.0000	0.2961	0.2961
Rainfall-Natural Phenomena	0.0000	0.3968	0.3968
Rainfall-Rainbow	0.0000	0.2353	0.2353
Rainfall-Storm	0.0000	0.3205	0.3205
Storm-Tornado	0.5000	0.2121	0.2121
Aurora-Rainbow	0.0000	0.3574	0.3574
Desert-Steppe	0.0000	0.8819	0.8819
Desert-Swamp	0.0000	0.5269	0.5269
Desert-Hill	0.0000	0.3971	0.3971
Desert-Island	0.0000	0.3627	0.3627
Desert-Ocean	0.0000	0.3299	0.3299
Ascomycota-Basidiomycota	0.7500	0.5055	0.5055
Auriculariales-Dacrymycetales	0.6667	0.3557	0.6667
Calanoida-Harpacticoida	0.7500	0.9687	0.9687
Calanoida-Cyclopoida	0.7500	0.9687	0.9687
Amphibian-Echinoderm	0.5000	0.3328	0.5000
Amphibian-Reptiles	0.5000	0.6789	0.6789
Molluscs-Reptiles	0.5000	0.3295	0.5000
Molluscs-Vertebrate	0.5000	0.4775	0.5000
Lobopodia-Arthropods	0.5000	0.6184	0.6184
Onychophora-Tardigrade	0.5000	0.9848	0.9848
Nemathelminthes-Arthropods	0.5000	0.7044	0.7044
Geology-Geophysics	0.6667	0.5654	0.5654
Geology-Orography	0.3333	0.5320	0.5320
Geology-Biology	0.3333	0.4715	0.4715
Geology-Literature	0.3333	0.3571	0.3571
Hydrosphere-Lithosphere	0.5000	0.3436	0.3436
Hydrosphere-Atmosphere	0.5000	0.2353	0.2353
Hydrosphere-Biosphere	0.5000	0.4883	0.4883

Continued

Magmatite-Sedimentary	0.6667	0.6360	0.6360
Magmatite-Metamorphic Rock	0.6667	0.4027	0.4027
Magmatite-Gneiss	0.6667	0.4720	0.4720
Metamorphic Rock-Gneiss	0.3333	0.8430	0.8430
Magmatite-Extrusive Rock	0.6667	0.7815	0.7815
Magmatite-Granite	0.6667	0.5101	0.5101
Chlorophyll-Photosynthetic Pigments	0.3333	0.5670	0.5670
Chlorophyll-Photosynthesis	0.0000	0.3865	0.3865
Carbon Dioxide-Photosynthesis	0.0000	0.0300	0.0300
Water-Photosynthesis	0.0000	0.1077	0.1077
Oxygen-Photosynthesis	0.0000	0.0604	0.0604
Carbohydrate-Photosynthesis	0.2000	0.2797	0.2797
Cirque-Moraine	0.5000	0.6373	0.6373
Moraine-Glacier	0.5000	0.7061	0.7061
Glacier-Glacier	0.5000	0.9919	0.9919
Ice Sheet-Glacier	0.5000	0.8654	0.8654
Valley Glacier-Glacier	0.5000	0.6143	0.6143
Oil-Natural Gas	0.0000	0.4948	0.4948
Oil-Coal	0.0000	0.3162	0.3162
Oil-Wind Power	0.0000	0.1924	0.1924
Oil-Solar Power	0.0000	0.1272	0.1272
Oil-Power	0.0000	0.4291	0.4291
Oil -Non-Renewable Energy Resource	0.0000	0.2837	0.2837
Oil-Clean Energy	0.0000	0.1494	0.1494
Earthquake-Tsunami	0.0000	0.8564	0.8564
Earthquake-Landslip	0.0000	0.5799	0.5799
Earthquake-Debris Flow	0.0000	0.3734	0.3734
Earthquake-Volcano	0.0000	0.3636	0.3636
Earthquake-Natural Disaster	0.0000	0.7747	0.7747
Chinese Antarctic Greatwall Station-Greatwall Station	0.4286	0.9940	0.9940
Chinese Antarctic Greatwall Station-China Zhongshan Station	0.7143	0.5826	0.7143
Chinese Antarctic Greatwall Station-China Domea Station	0.7143	0.5199	0.7143
Chinese Antarctic Greatwall Station-China Yellow Station	0.5714	0.6437	0.6437
Yellow River Station-The Arctic Ocean	0.0000	0.3268	0.3268
Yellow River Station-Southern Ocean	0.0000	0.1663	0.1663
Greatwall Station-Antarctica	0.0000	0.1062	0.1062
Greatwall Station-The Arctic Ocean	0.0000	0.0890	0.0890

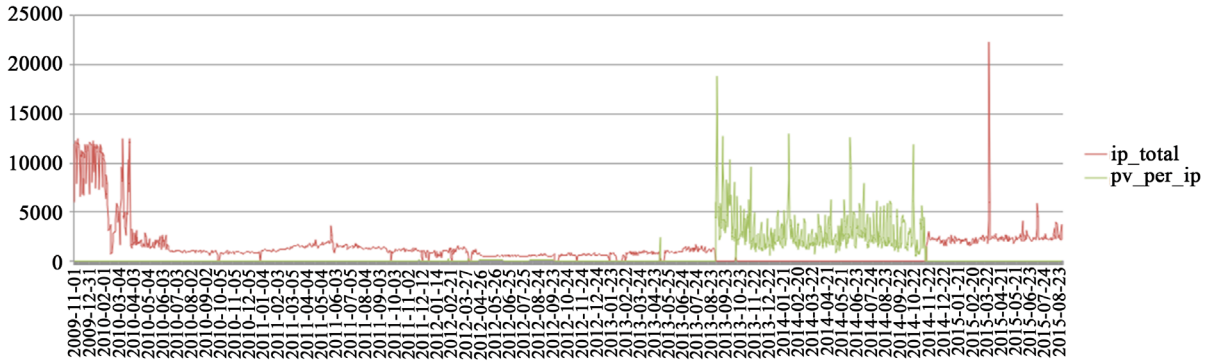

Figure 3. Value of PV, IP on BIRDS.

method can detect similar data more efficiently and more accurately to improve data accuracy, compared with the traditional literal string matching method and DBpedia semantic similarity algorithm. Also, the algorithm application on BIRDS proves its applicability and convenience.

Foundation

This work is supported by National Ocean Public Benefit Research Foundation(201305035); Key Laboratory of Digital Ocean, SOA (KLD0201408) respectively, Platform for Basic Conditions of the Ministry of Science and Technology (2005DKA21406); National postdoctoral science foundation (2013M531120); Project of Henan Province Science and Technology (No. 142300410247); Project of Henan Province Education Department (14A413002); Project of Zhengzhou Scientific and Technology Department (131PPTGG411-4).

References

[1] Anagnostopoulos, A., Becchetti, L., Castillo, C., *et al.* (2010) An Optimization Framework for Query Recommendation. *Proceedings of the* 3*rd ACM International Conference on Web Search and Data Mining*, 161-170. http://dx.doi.org/10.1145/1718487.1718508

[2] Cheng, W.F., Zhang, J., Xia, M.Y., *et al.* (2013) System Design and Implementation of a Resource-Sharing Platform for Polar Samples. *Polar Research*, **25**, 185-196.

[3] Linden, G., Smith, B. and York, J. (2003) Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing*, **7**, 76-80. http://dx.doi.org/10.1109/MIC.2003.1167344

[4] Bai, R.J., Yu, X.F. and Wang, X.Y. (2011) The Comparative Analysis of Major Domestic and Foreign Ontology Library. *New Technology of Library and Information Service*, **1**, 3-13.

[5] Gabrilovich, E. and Markovitch, S. (2007) Computing Semantic Relatedness Using Wikipedia-Based Explicit Semantic Analysis. *IJCAI*, **7**, 1606-1611.

[6] Sheng, Z.-C. and Tao, X.-P. (2011) Semantic Similarity Computing Method Based on Wikipedia. *Computer Engineering*, **37**.

[7] Liu, J. and Yao, T.-F. (2010) Semantic Relevancy Computing Based on Wikipedia. *Computer Engineering*, **36**.

[8] Chao, L.M., Zhang, Y. and Xing, C.X. (2011) DBpedia and Its Typical Applications. *New Technology of Library and Information Service*, **3**, 80-87.

[9] Breiman, L., Friedman, J.H., Oishen, R.A., *et al.* (1984) Classification and Regression Trees. Wadsworth, Inc.

[10] Chen, Z.W., Wu, Q.E and Yang, W.D. (2015) Target Image Classification through Encryption Algorithm Based on the Biological Features. *International Journal of Intelligence Science* (*IJIS*), **5**, 6-12. http://dx.doi.org/10.4236/ijis.2015.51002

[11] Bezdek, J.C. (1981) Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York. http://dx.doi.org/10.1007/978-1-4757-0450-1

[12] Qing, L., Xu, X.-D. and Wang, S.-T. (2012) Fuzzy Particle Filter for Object Tracking. *AASRI Procedia*, **3**, 191-196. http://dx.doi.org/10.1016/j.aasri.2012.11.032

[13] Meia, W., Xiao, Y. and Wang, G. (2012) Object Classification Based on a Combination of Possibility and Probability Likelihood in the Bayesian Framework. *Procedia Engineering*, **29**, 9-14.

http://dx.doi.org/10.1016/j.proeng.2011.12.659

[14] Mai, F.-J., Li, D.-P. and Yue, X.-G. (2011) Research on Chinese Word Segmentation Based on Bi-Direction Marching Method and Feature Selection Algorithm. *Journal of Kunming University of Science and Technology* (*Natural Science Edition*), **36**, 47-51.

[15] Martelli, A., Ravenscroft, A. and Ascher, D. (2008) Python Cookbook. O'Reilly.

[16] Wiki API. http://zh.wikipedia.org/w/api.php