Semantic Web Improved with the Weighted IDF Feature

Mrs. Jyoti Gautam
Department of Computer Science and Engineering
JSSATE (Uttar Pradesh Technical University)
NOIDA, U.P., INDIA

Dr. Ela Kumar
Department of Computer Science and Engineering
Indira Gandhi Delhi Technical University for Women
Delhi, INDIA

***Abstract*—The development of search engines is taking at a very fast rate. A lot of algorithms have been tried and tested. But, still the people are not getting precise results. Social networking sites are developing at tremendous rate and their growth has given birth to the new interesting problems. The social networking sites use semantic data to enhance the results. This provides us with a new perspective on how to improve the quality of information retrieval. As we are aware, many techniques of text classification are based on TFIDF algorithm. Term weighting has a significant role in classifying a text document. In this paper, firstly, we are extending the queries by "keyword+tags" instead of keywords only. In addition to this, secondly, we have developed a new ranking algorithm (JEKS algorithm) based on semantic tags from user feedback that uses CiteUlike data. The algorithm enhances the already existing semantic web by using the weighted IDF feature of the TFIDF algorithm. The suggested algorithm provides a better ranking than Google and can be viewed as a semantic web service in the domain of academics.**

Keywords—Text classification; Semantic Web with weighted idf feature; Expanded query; New Semantic Web Algorithm; Ranking Algorithm

I. INTRODUCTION

A lot of information is available on the Internet. Search engines remain as the primary infrastructure for Information Retrieval. The relevance of the result-sets is not as desired by the user. This leads to the requirement of a good ranking algorithm to put the best results on the front.

Many popular Web services like Delicious, Citeulike and flickr.com rely on folksonomies (Gautam and Kumar, 2012).Some websites such as CiteUlike (Research Paper Recommender), Delicious (online bookmarking), Flickr (online photo management and sharing application), Furl (File Uniform Resource Locators), Blinklist (links saver), Diigo (collect and organize anything e.g. bookmarks, highlights, notes, sceenshots etc.), Otavo (collaborative web search), Stumbleupon (discovery engine), Blummy (tool for quick access to favorite web services), and Folkd (saves bookmarks and links online) etc. which contain these tag information.

Various difficulties are encountered while doing research on folksonomies. In spite of all this, the growth is tremendous in this area. Researches based on social-bookmarking have become increasingly popular, which lets users specify their keywords of interest, or tags on web resources. Social tagging, also known as social annotation or collaborative tagging is one of the major characteristics of Web 2.0. Social-tagging systems allow users to annotate resources with free-form tags. The resources can be of any type, such as Web pages (e.g., delicious), videos (e.g., YouTube), photographs (e.g., Flickr), academic papers (e.g., CiteULIke), and so on.

In this paper, we utilize the semantic tag information with web page. This information is obtained from CiteUlike (Research Paper Recommender and online Tagging System).When users submit their query; they also submit some semantic description to disambiguate the query. Then, by matching the semantic description between the query and web page, user's query intent can be well understood. The better understanding of the user's query leads to better ranking results in academic domain.

In this paper, the following approach has been adopted. We have tried to use the metadata available in the form of user feedback and semantic tags from CiteUlike.

a) A new ranking algorithm has been developed. The algorithm utilizes the weighted IDF feature of the TFIDF algorithm.

b) The query was expanded. The idea was to use "keyword + tags" instead of keywords only, so that it carries some semantic description along with it.

c) The data was obtained through CiteUlike.

d) The performance analysis was done by comparing the approach with Google by several evaluation methods.

The paper is organized by an introduction to the existing ranking methods, then the new optimized JEKS algorithm followed by significance of the algorithm. Thereafter, the experiments and analysis is done followed by significance and relevance of the research work. In the end, finally the paper is concluded.

II. THE EXISTING RANKING METHODS

Tf-idf, term frequency-inverse document frequency is a numerical statistic which reflects how important a word is to a document in a corpus. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus.

The literature (S. Lu, X. Li, S. Bai and S. Wang., 2000) provides an improved approach named tf.idf.IG to remedy this defect by Information Gain from Information Theory.

The literature (S. Lu, X. Li, S. Bai and S. Wang., 2000) provides an improved approach named tf.idf.IG to remedy this defect by Information Gain from Information Theory.

The Lingo algorithm proposed by Osinski and Weiss (2005) combines common phrase discovery and latent semantic indexing techniques to separate search results into meaningful groups. It looks for meaningful phrases to use as cluster labels and then assigns documents to the labels to form groups.

(Wu, Zhang and Yu, 2006) explored the technique of Social Annotations for the Semantic Web. These annotations are manually made by normal web users without a predefined formal ontology. The evaluation of the approach shows that the method can effectively discover semantically related web bookmarks that current social bookmark service cannot discover easily.

(Farooq, Kannampallil and Song, 2007) The authors use six tag metrics to understand the characteristics of a social bookmarking system. Possible design heuristics was suggested to implement a social bookmarking system for Cite Seer using the metrics.

The authors Cilibrasi and Vitanyi (2007) described a technique for calculating the Google similarity distance.

Jin, Lin and Lin (2008) proposed the architecture of a semantic search engine and an improved algorithm based on TFIDF algorithm. The algorithm considers crawling of static web pages. The algorithm can be considered for crawling of dynamic web pages and for parallel crawling also.

A personalized search framework was proposed by Shenliang, Shenghua and Fei (2008).It utilizes folksonomy for personalized search.

(Jiang, Hu, Li, and Wang 2009).The other method of basic TFIDF model uses supervised term weighting approach. The model uses class information to compute weighting of the terms. The approach is based on the assumption that low frequency terms are important, high frequency terms are unimportant, so it designs higher weights to the rare terms frequently.

Jomsri, Sanguansintukul and Choochaiwattana (2010) proposed a framework for Tag-Based Research Paper Recommender system. User self-defined tags were used for creating a profile for each individual user and cosine similarity was used to compare a user profile and research paper index. The recommender system demonstrated an encouraging preliminary result with the overall accuracy percentage up to 91.66%. The number of subjects is considered to be small in the experiment.

(Zhao and Zhang, 2010) proposed a new viewpoint on how to improve the quality of information retrieval. The queries are extended by "keywords+tags" instead of keywords only. A new tag based ranking algorithm (OSEARCH) was proposed and the results obtained were also compared with Google by several evaluation methods.

The authors Leung and Lee (2010) focussed on search engine personalization and developed several concept-based user profiling methods that are based on both positive and negative preferences. The proposed methods were evaluated against the previously proposed personalized query clustering method.

(Kaczmarek, 2010) introduced a novel approach to interactive query expansion. When a user executes a query, the algorithm shows potential directions in which the search can be continued.

Another supervised term weighting method, proposed by the authors (Zhanguo, Jing, Liang, Xiangyi and Yanqin, 2011), provides an improved tf-idf-ci model to compute weighting of the terms. The method uses intra and inner class information.

Various variations of the tf–idf weighting scheme are often used by search engines. Search engines use these weighted measures as a central tool in scoring and ranking a document's relevance given a user query. The tf-idf is improved by many literatures. The proportion of distribution of terms in text collection is one of the most important factors of expressing the content of text, but it is beyond tf-idf's power (Zhanguo, Jing, Liang, Xiangyi and Yanqin, 2011).

The paper proposed by (Yoo, 2011) suggests a hybrid query processing method for the effective retrieval of personalized information on the semantic web. When individual requirements change, the current method of query processing requires additional reasoning for knowledge to support personalization.

(Halpin and Lavrenko, 2011) proposed the method of relevance feedback between hypertext and semantic web search. The paper proposed investigates the possibility of using semantic web data to improve hypertext web search.

In this paper, the authors (Gracia and Mena, 2012) presented the web's natural semantic heterogeneity problems – namely, redundancy and ambiguity. The authors' ontology matching, clustering, and disambiguation techniques aim to bridge the gap between syntax and semantics for Semantic Web construction.

The authors Zhong, Li and Wu (2012) proposed an effective pattern discovery method for text mining. The paper presents an innovative and effective pattern discovery technique which includes the processes of pattern deploying and pattern evolving, to improve the effectiveness of using and updating discovered patterns for finding relevant and interesting information.

The paper (Lee, Kim and Park, 2012) proposes searching and ranking method of relevant resources by user intention on the semantic web. There are more limitations in information searching as the information on the Internet dramatically increases. To overcome the various limitations, the Semantic Web must provide search methods based on the different relationships between resources.

This paper proposed by (Gautam and Kumar, 2012) proposes a framework for a tag-based Academic Information Sharing and Recommender System which shares information such as question papers, assignments, tutorials and quizzes on a specific area.

(Shaikh, Siddiqui and Shahzadi, 2012) proposed the Semantic Web based Intelligent Search Engine. SWISE required including domain knowledge in the web pages to answer intelligent queries. The layered model of Semantic Web provides solution to this problem by providing tools and technologies to enable machine readable semantics in current web contents.

(Lee, Kim, and Park 2012) presented some proposals to improve and extend the semantic approach based on conceptual neighborhood's graphs in order to best preserve the proximity between the adapted and original documents and to deal with models that define delays and distances.

III. User Query Intent and Storage of Tags

A. Metadata Information in the Web Pages and Expansion of the Query

While talking about semantic web, metadata comes into picture. What is this semantic? How is it related to metadata? Semantic Web is something that implies the content, meaning or the metadata related to the web. This metadata information is hidden in the web pages. There are different websites which are working upon it since a long time. We have sites like Delicious, CiteUlike, Flickr etc., which allow different users to create their accounts. After creating the accounts, the users can add metadata for the different websites. This metadata conveys the content of the website as interpreted by different users.

The method should be such that which tries to capture the user's real query intent. The primary purpose of the search engines is to return the optimal results. But before returning the results, it should be able to analyze the query clearly. The simple keywords can't express user's real query intent. In order to analyze the query, some metadata information is added along with the query. The metadata information is added by expanding the query .i.e., keyword+tags instead of the keywords only.

So, the idea is to consider utilizing metadata which is available in the form of semantic tags .One area that arises is to consider utilizing the semantic tag information with web page. When users submit their query, they can also submit some simple semantic description to narrow down the query. Then by matching the semantic information between query and web page metadata, we can understand user's query intent better and return better result.

So, the idea is to utilize this semantic tag information. Here, we are proposing the development of a new algorithm based on semantic tags and the weighted IDF feature of the TFIDF algorithm.

B. Storage of Semantic Tags on Web Pages

The semantic tags of a web page are some object properties that reflect the content of the web page, such as marked with "semantic web", which signifies that the page contains information about the object of "semantic web". Of course, there may be multiple tags on a page, because the pages always contain multi information. These tags carry the metadata information along with them.

In our case, we are storing the tags from CiteUlike. A popular website in academia is CiteULike (www.CiteULike.org). CiteUlike is a free service for managing and discovering scholarly references.

- Easily store references you find online
- Discover new articles and resources
- Automated article recommendations
- Share references with your peers
- Find out who's reading what you are reading
- Store and search your PDF's

CiteULike has a filing system based on tags. Tags provide an open, quick and user-defined classification model that can produce interesting new categorizations.

Additionally, it is also capable to:

- 'tag' papers into categories.
- Add your own comments on papers.
- Allow others to see your library

The semantic tags are retrieved from CiteUlike. The URLs along with their tags are stored in a local database. For the semantic tags, each URL is opened in CiteUlike and the tags with their numeric values are stored in the database. We add tags' values in the MYSQL database. The data was retrieved from April, 2012 to June, 2013 from CiteUlike for the 50 queries. A total of 5000 URLs were opened in CiteUlike and the database was created.

IV. A new Optimized Ranking Algorithm

A. Utilizing the Weighted Inverse Document Frequency

In this paper, we are proposing a new algorithm based on semantic tags in the web pages. An enhanced semantic web algorithm is proposed. The algorithm is based on utilizing the metadata information available with the web pages by integrating in the algorithm some good features of weighted IDF.

Here, we are improving the semantic web by utilizing the weighted IDF score. We are already familiar with (1), which is applicable in the context of TFIDF (Jiang, Hu, Li, and Wang, 2009)

$$W(tk,dj,ci) = (1-\alpha).tfidfk,j + \alpha. tfidfk,j \times weighting \quad (1)$$

$$weighting = Ai/Ci, \quad \text{(Refer TABLE 1.)} \quad (2)$$

α is called a balance factor, which lies between , $0 \leq \alpha \leq 1$.

When $\alpha = 0$, (1) becomes classic TFIDF approach, and when $\alpha = 1$, (1) becomes our newly improved approach. Using balance factor, we can get better classification results.

TABLE I. Below shows the Relation of Term T_K and Category C_i.

	C_i	$\bar{C}_i$
t_k	A	B
$\bar{t}_k$	C	D

A indicates the number of documents belonging to category C_i where the term t_k occurs at least once; B indicates the number of documents not belonging to category C_i where the term t_k occurs at least once; C denotes the number of documents belonging to category C_i where the term t_k does not occur at least once; D denotes the number of documents not belonging to category C_i where the term t_k does not occur at least once.

This equation (1) is applicable for the terms of the document. The same equation can be used for tags also. Let us take an example. For the three tags, tag1, tag2, tag3 of the category Ci, if they share the same values of tf-idf but have different proportion of A and C. So, the tags which have higher values of the weighting factor make more contribution to the category Ci. Evidently, the tf-idf approach gives equal weights to the three tags unlike the weighted ones.

Now, we have integrated this equation with the other equation proposed by Zhao and Zhang (2010)

B. A New Optimized Ranking Algorithm – JEKS (Jyoti and Ela Kumar Search) algorithm

Initially, when users want to submit a query, instead of just giving the query in the form of keywords, they will also expand the query by adding some metadata information along with the query. Afterwards, the algorithm compares the inputted tags in query with the semantic information on the web pages in order to provide the user with better results.

Accordingly, the user query can be expressed as:

Query = {keyword1, keyword2,…, tag1, tag2,…}

In the above formulation, keyword1, keyword2 is the main query keyword.Tag1; tag2 is the semantic information which we are adding to expand the query. For example, Query = {research papers, web mining) represents that the user wants to find information relating to research papers on web mining.

Similarly, Query = {resources, information retrieval}

represents that the user wants to find information relating to resources in the field of information retrieval.

Once, the query is submitted, the system creates a vector of all the user tags.

V_usrt = {user_tag1, user_tag2,…}

Once the query is submitted to the search engine, the engine returns an initial result page list. The vector of all the tags on the result pages is recorded.

V_rest = {r_tag1, r_tag2,…}

Where, r_tag1, r_tag2 represent semantic tags on result pages.

The similarity is calculated between the two tag vectors, and recorded as a Tg_score.

Then, the final score of the web page is:

TotalScore=google_score+(Tg_score*IDFscore*weighting) (3)

Score=Tg_score*IDFscore*weighting (4)

Re – rank the google results according to this score.

Here, google_score represents the original google results score when the query is applied.

Google_score=(p-q+1)/p (5)

Here, p represents the total no. of documents, which is 100 in the experiment; q represents the location of the document on search engine's result list. So, google_score for the 6th result is (100- 6 + 1) / 100 = 0.95.

In (3), Tg_score is calculated by matching the tags of the user with the tags of the result page. The match between the two vectors is based on the following factors.

1) The similarity between the user tag vector and web page tag vector. The high value is obtained by high similarity between the two vectors.

2) The other factor being the weight of the tags on the result pages. Weight refers to the frequency of the the tags in the result pages which match with the tags of the user.

Tg_score is defined as given below based on the factors considered:

$$\text{Tg\_score}=\frac{\sum_{i=1}^{|\text{V\_usrt}|}\sum_{k=1}^{|\text{V\_rest}|}(freq(\text{V\_rest[i]})*\text{sim}(\text{V\_usrt[i]}),\text{V\_rest[k]}))}{\sum_{k=1}^{|\text{V\_rest}|}freq(\text{V\_rest[k]})} \quad (6)$$

In the above equation, freq (tag) represents the frequency or weight of the particular tag on the result page. $\text{sim}(\text{V\_usrt[i]}),\text{V\_rest[k]})$ represents the similarity between the user tag vector V_usrt[i] and the result page tag vector V_rest[k] and similarity is defined as given below:

$$\text{sim}(\text{V\_usrt[i]}),\text{V\_rest[k]})$$

= 1, V_usrt[i] and V_rest[k] have the same root,
= 1, V_usrt[i] and V_rest[k] have the same meaning,
= 0, V_usrt[i] and V_rest[k] does not have a semantic relation,
=0.5, even if half of the V_usrt[i]tag resembles with the V_rest[k]tag. (7)

,e.g. let us say in the Query = {resources, information retrieval} , resources is the keyword and information retrieval is the tag, then in the tags of the result pages even if information or retrieval appears , we have taken the similarity score as 0.5.

Next, ,e.g. consider the query , Query = {artificial intelligence, pdf} to Google, The tenth result has the tags as "pdf", "pdfs", "research" and the frequency of the tags is 10, 9, 4 respectively. Then, the value of the Tg_score = (10*1+9*1+4*0)/ (10+9+4) = 19/23 and google_score = (100-10+1)/100=0. 91.

Next in (3) is the IDF score multiplied by weighting. We know from the TFIDF algorithm.

Given a document collection D, a word w, and an individual document d Є D, we calculate

$$w_{d=}f_{w,d}*\log(|D|/f_{w,D}), \quad (8)$$

Where $f_{w,d}$ equals the number of times w appears in d, |D| is the size of the corpus, and $f_{w,D}$ equals the number of documents in which w appears in D. Words with high w_d imply that w is an important word in d but not common in D.

Here, if the above equation is analyzed properly, we see that if we replace words with tags, the (8) can be used in the context of semantic web. So, $f_{w,d}$ has already been considered

as the Tg_score. Now remains the log ($|D|/f_{w,D}$), (which is IDF score). Here, for each query, we have taken the 100 Google results. So, for a particular query, D is 100 and $f_{w,D}$ equals the number of documents in which the particular tag of the query appears.

Now, why we have included this IDF score?

Suppose that Tg_score is large and fw,D score is small. Then log ($|D|/f_{w,D)}$ will be rather large, and so in (3), the score will be large. This is the case we are most interested in, since tags with high score imply that this tag is important for the document d but not common in D. This tag is having a large discriminatory power. Therefore, when a query contains this tag, returning a document d where score is large will very likely satisfy the user.

Now, we are multiplying this IDF score with the weighting factor. As, we have already mentioned the significance of this weighting factor .Let us take an example. Let us replace the terms with the tags in (8).If the values of (Tg_score * IDF score) is similar for the different tags, then weighting factor is used to differentiate the results. The tags with the higher weighting will be preferred as they have higher discriminating power for the category Ci in comparison to the tags having less weighting factor. The tags having less weighting may be rare tags in the category Ci.

Now, calculating the (IDF score * weighting factor) for the Query = {books, artificial intelligence}, let us say that the documents in which the tag artificial intelligence appears is 30 and the value of D is 100. So, the IDF score is log (100/30) and weighting factor is (30/70).

In the above (6), we are using java functions to calculate the similarity between user tags and result tags. The database is created using MYSQL.

For example, user submits the query "research papers, mobile computing", to Google, the 4th result of Google is having the tag's values, mobile computing = 37, mobile devices = 35, mobile interaction = 27, pedestrian navigation = 23, navigation = 12. And, the tag mobile computing appears in 37 documents. So, according to the above algorithm, the total score = (0.97) + (0.507) * log (100/37)*(37/63).

V. Significance of the JEKS Algorithm

The JEKS algorithm developed above is effective in the case when (Tg_score*IDF) score is similar for the different tags in a category Ci. Through the values of the proportion of Ai and Ci, it can be easily found that the three tags show different discriminating power to TC (Refer TABLE 2). The weighting factor can be used to differentiate the results. The tags with the higher weighting will be preferred as they have higher discriminating power for the category Ci in comparison to the tags having less weighting factor. The tags having less weighting may be rare tags in the category Ci. For example, take a class Ci as research papers and the three different tags as mobile computing, data mining and semantic web. Corresponding to this, the three different queries are {research papers, mobile computing}, {research papers, data mining} and {research papers, semantic web}. Now for a particular case when Tg_Score and IDF Score is similar for the three different tags of the class Ci, then Ai/Ci will be used to produce three different TotalScore values (Refer (3))., and hence different rankings.

TABLE II. THREE TAGS WHICH SHARE THE SAME (TG_SCORE*IDF SCORE) BUT HAVE DIFFERENT PROPORTION OF AI AND CI IN A CATEGORY CI

Tag	Tg_Score	IDF Score	Ai/Ci
Tag1(mobile computing)	.507	Log(100/37)	2:10
Tag2(data mining)	.507	Log(100/37)	1:1
Tag3(semantic web)	.507	Log(100/37)	10:2

The TABLE 2 shows that the tag3 gives higher discriminating power to the category Ci from other categories than the tags tag1 and tag2. The tag1 may be a rare tag in the category Ci, and makes little contribution to the category Ci. So, the TotalScore will be highest for the tag3, lowest for tag1 and for tag2, it lies in between.

VI. Experiments and Analysis

The experiments are performed as follows:

1) Initially, submit the query to Google, and obtain the original Google search results.

2) Now, submit the Google search results to CiteUlike to obtain the relevant tags.

3) Re-rank the search results according to our algorithm.

4) Compare the Google results with our algorithm.

A. Data Set

Query Set: Initially, we determine the queries which we input to the search engine. We determine a total of fifty queries. The queries are a combination of keywords and tags. These queries are submitted to Google. The queries are from academic domain as CiteUlike provides tags for the academic database.

Result Set: Now, submit each query to Google and record the first 100 results. This way, the result set of 50 queries become 5000 results.

Results Tag Set: Now, we submit the 5000 results to CiteUlike and the resulting tag vector is recorded. We obtain lots of tag values for a result.

For example, user submits the query "resources, genetic algorithm", to Google, the 4th result of Google is having the tag's values, genetic algorithm = 37, genetic = 35, algorithm = 27, pedestrian navigation = 23, navigation = 12. And, the tag genetic algorithm appears in 40 urls. So, according to the above algorithm, the total score = (0.97) + (0.507) * log (100/40)*(40/60).

We have chosen the following queries.

Q1 = {books, artificial intelligence}
Q2 = {books, grid computing}
Q3 = {books, information retrieval}
Q4 = {books, java programming}
Q5 = {books, software engineering}
Q6 = {pdf, artificial intelligence}
Q7 = {pdf, cloud computing}
Q8 = {pdf, data structure}
Q9 = {pdf, deep web}
Q10 = {pdf, digital image processing}
Q11 = {pdf, distributed computing}
Q12 = {pdf, parallel algorithm}
Q13 = {pdf, semantic web}
Q14 = {research papers, communication}
Q15 = {research papers, compiler}
Q16 = {research papers, data mining}
Q17 = {research papers, genetic algorithm}
Q18 = {research papers, mobile computing}
Q19 = {research papers, pharmacology}
Q20 = {research papers, quantum cryptography}
Q21 = {research papers, semantic web}
Q22 = {research papers, software engineering}
Q23 = {research papers, statistics}
Q24 = {research papers, ubiquitous computing}
Q25 = {research papers, web mining}
Q26 = {research papers, wireless communication}
Q27 = {resources, electronics engineering}
Q28 = {resources, grid computing}
Q29 = {resources, information retrieval}
Q30 = {resources, semantic web}
Q31 = {resources, ubiquitous computing}
Q32 = {books, automata}
Q33 = {books, data mining}
Q34 = {books, power electronics}

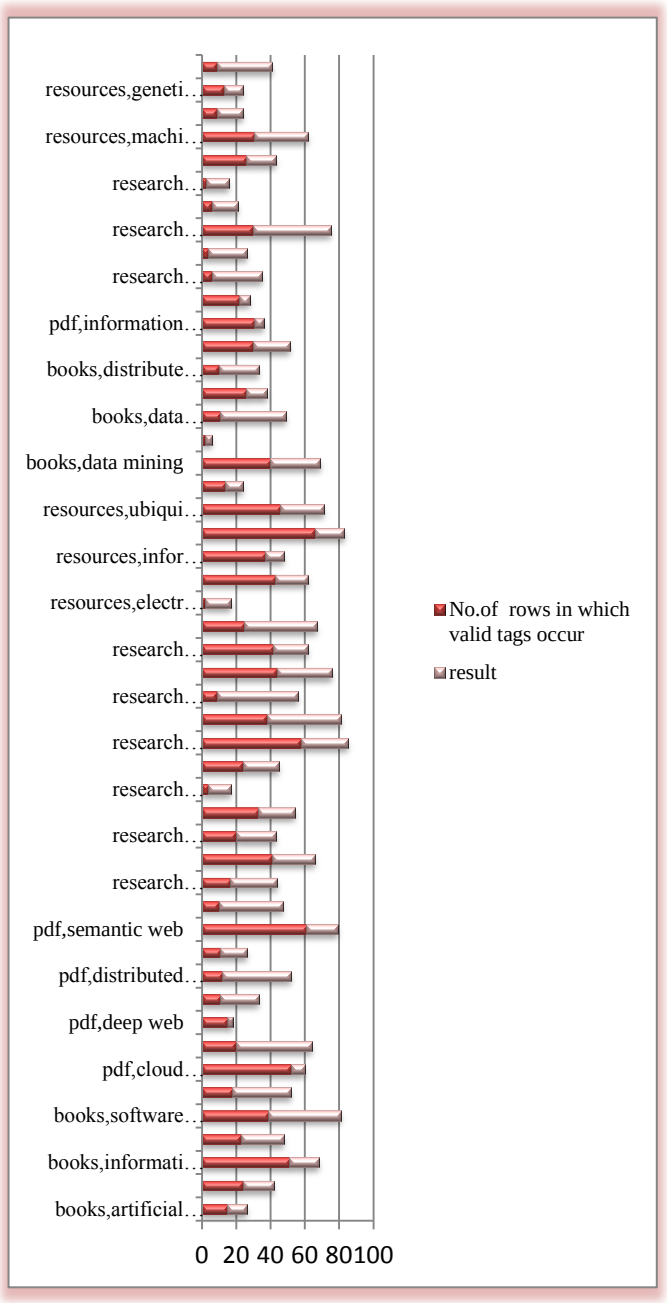

Fig. 1. The number distribution of specific tags versus difference tags in a result set

Rows in which valid (specific) tags occur = A
Rows of total tags = B, Difference Tags (result) = C = (B-A)

e.g., for the query = {research papers, data mining} A = 41, B = 66, C = 25.

Q35 = {books, data structure}
Q36 = {books, deep web}
Q37 = {books, distributed computing}
Q38 = {books, web mining}
Q39 = {pdf, information retrieval}
Q40 = {pdf, genetic algorithm}
Q41 = {research papers, digital signal}
Q42 = {research papers, fluid mechanics}
Q43 = {research papers, machine learning}
Q44 = {research papers, molecular electronics}
Q45 = {research papers, power electronics}
Q46 = {resources, database}
Q47 = {resource, machine learning}
Q48 = {resources, molecular electronics}
Q49 = {resources, genetic algorithm}
Q50 = {resources, structure analysis}

B. Experimental Results

First, we determine the relevance between each query intent and each result page. Each result is assigned a relevance score according to its relevance, which ranges between 0 to 3 (totally irrelevant, basically irrelevant, basically relevant, and totally relevant).

We obtain normalized DCG values for our algorithm and Google as given in the Table 3.

TABLE III. COMPARISION OF NORMALIZED DCG (*NDCG*) VALUES FOR OUR ALGORITHM AND GOOGLE

QUERY NO.	nDCG(A)	nDCG(G)
q1	0.957424	0.970031
q2	0.888747	0.913824
q3	0.877744	0.862172
q4	0.938299	0.934294
q5	0.854472	0.881374
q6	0.887192	0.885906
q7	0.975138	0.97113
q8	0.86662	
q9	0.834386	0.796038
q10	0.920252	0.942012
q11	0.959862	0.953069
q12	0.995585	0.995332
q13	0.982126	0.981987
q14	0.897661	0.84126
q15	0.881929	0.848669
q16	0.933084	0.894468
q17	0.975616	0.983474
q18	0.908892	0.85308
q19	0.805438	0.801738
q20	0.929742	0.91508
q21	0.945845	0.938982
q22	0.92802	0.913109
q23	0.879856	0.770643
q24	0.956999	0.945143
q25	0.83687	0.760944
q26	0.934957	0.92141
q27	0.928905	0.928905
q28	0.994868	0.994253
q29	0.957072	0.964861
q30	0.993879	0.992997
q31	0.986664	0.984467
q32	0.877298	0.837017
q33	0.911324	0.905407
q34	0.934142	0.934407
q35	0.91458	0.902485
q36	0.900831	0.940498
q37	0.87344	0.941972
q38	0.887338	0.854466
q39	0.983348	0.976529
q40	0.982363	0.981797
q41	0.907483	0.851692
q42	0.840634	0.790858
q43	0.905059	0.883763
q44	0.856174	0.818381
q45	0.969632	0.969562
q46	0.961921	0.943682
q47	0.986109	0.979038
q48	0.986892	0.983111
q49	0.98629	0.981852
q50	0.855997	0.882663

We obtained normalized DCG values for the 50 queries for our algorithm as well as for Google results. We observed that Fig. 2 shows the normalized DCG values of 50 queries. The graph compares our algorithm with Google. It can be seen that our algorithm acquires higher values of DCG for 40 queries when compared to Google.

Next, we use Precision@k curve for various Relevance levels.

The following conclusion can be drawn from the Fig. 3 to Fig. 5.Our algorithm acquires higher precision in comparison

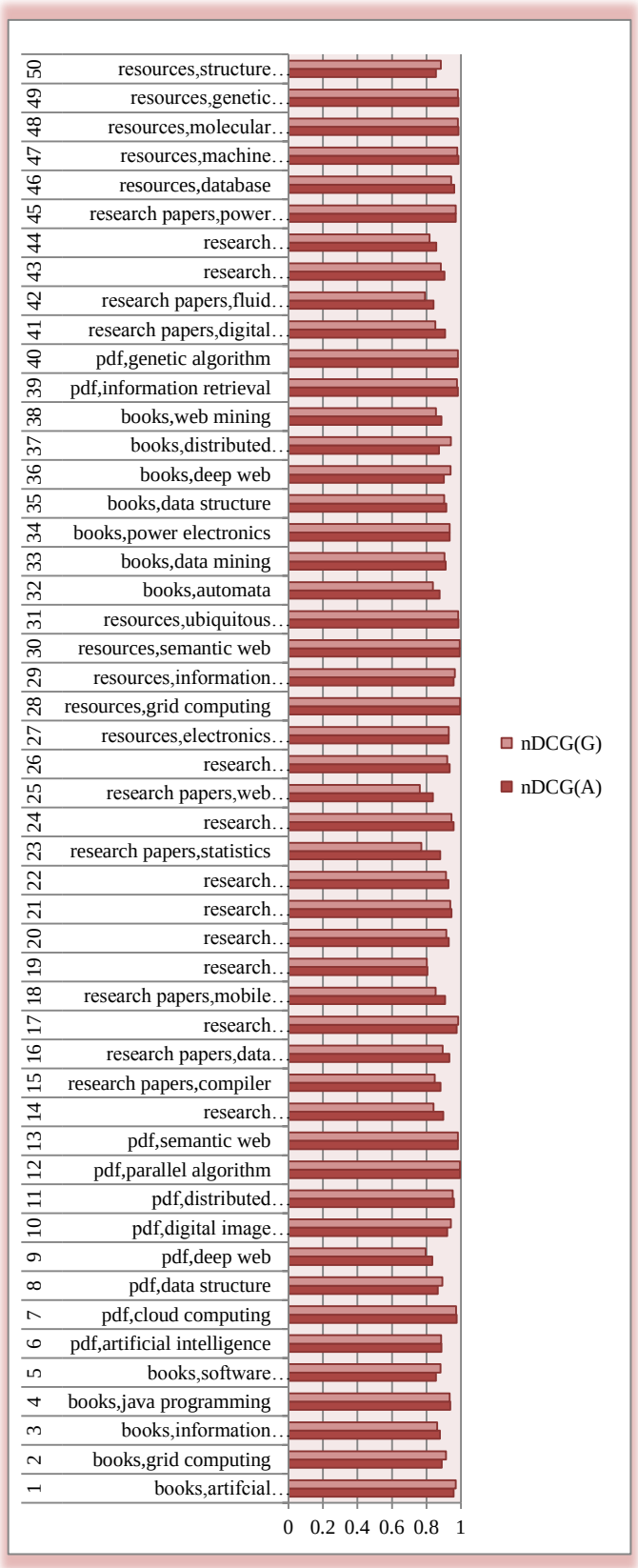

Fig. 2. The average DCG value of 50 queries

to Google throughout the varying levels of K for all the 50 queries. The results obtained for Rel>=1 are the best as expected. The precision for Rel>=1 are better than Rel>=2, which is better than Rel>=3.Only, when the Rel>=3, initially Google results are better as can be seen from Fig. 5.

We computed the values for precision, recall and F1-score for our algorithm and Google (Table 4.). These values are calculated for all the queries. These values are calculated for their corresponding top 50 results for Rel>=2 for all the 50 queries. We observed that the value of recall for our algorithm and Google remain at 1 as we have re-ranked the top 100 results of Google for each query. The value of precision and F1-score are calculated and it has been observed that we are getting better results.

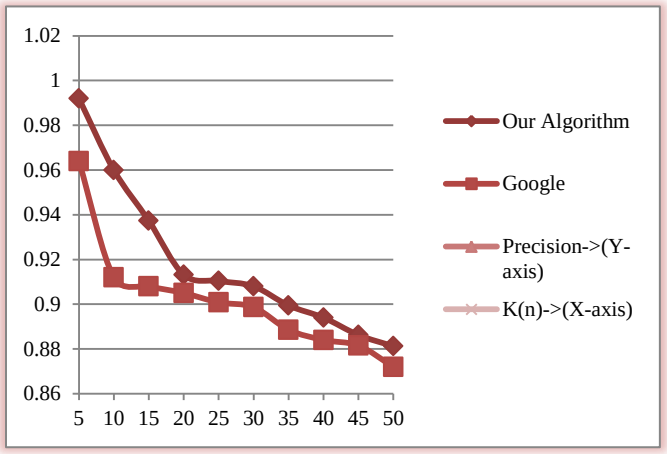

Fig. 3. The Precision@k curve of 50 queries when Rel>=1

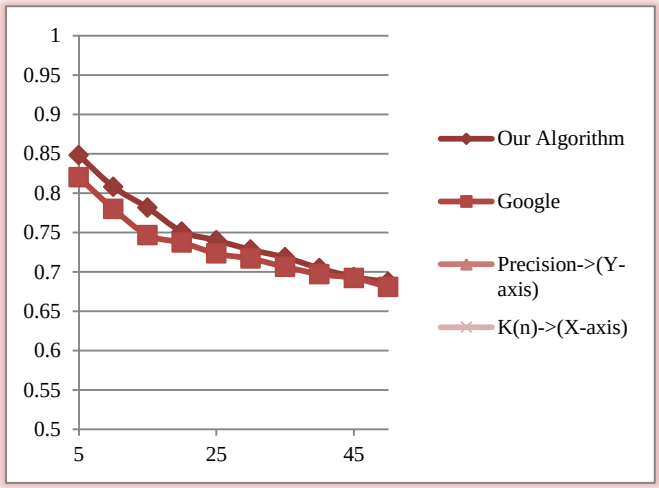

Fig. 4. The Precision@k curve of 50 queries when Rel>=2

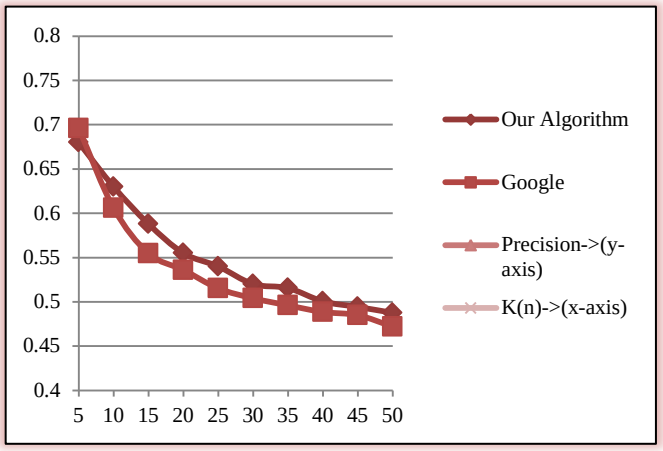

Fig. 5. The Precision@k curve of 50 queries when Rel>=3

TABLE IV. PRECISION AND F1-SCORE FOR OUR ALGORITHM AND GOOGLE

	JEKS algo		Google	
Query	*PRECISION*	*F1-score*	*PRECISION*	*F1-score*
q1	0.94	0.969	0.96	0.98
q2	0.5	0.667	0.5	0.667
q3	0.34	0.507	0.38	0.551
q4	0.72	0.837	0.72	0.837
q5	0.72	0.837	0.7	0.824
q6	0.8	0.889	0.8	0.889
q7	0.96	0.98	0.94	0.969
q8	0.5	0.667	0.5	0.667
q9	0.32	0.485	0.3	0.462
q10	0.9	0.947	0.9	0.947
q11	0.88	0.936	0.86	0.925
q12	0.96	0.98	0.96	0.98
q13	0.98	0.99	0.98	0.99
q14	0.56	0.718	0.56	0.718
q15	0.5	0.667	0.48	0.649
q16	0.64	0.78	0.62	0.765
q17	0.92	0.958	0.88	0.936
q18	0.72	0.837	0.72	0.837
q19	0.26	0.413	0.24	0.387
q20	0.7	0.824	0.68	0.81
q21	0.86	0.925	0.86	0.925
q22	0.66	0.795	0.62	0.765
q23	0.42	0.592	0.42	0.592
q24	0.8	0.889	0.78	0.876
q25	0.48	0.649	0.48	0.649
q26	0.74	0.851	0.68	0.81
q27	0.6	0.75	0.6	0.75
q28	1	1	1	1
q29	0.84	0.913	0.84	0.913
q30	1	1	1	1
q31	0.94	0.97	0.94	0.97
q32	0.54	0.701	0.5	0.667
q33	0.6	0.75	0.62	0.765
q34	0.42	0.592	0.42	0.592
q35	0.16	0.276	0.18	0.305
q36	0.42	0.592	0.42	0.592
q37	0.72	0.837	0.72	0.837
q38	0.46	0.63	0.5	0.667
q39	0.96	0.98	0.96	0.98
q40	0.96	0.98	0.96	0.98
q41	0.44	0.611	0.42	0.592
q42	0.42	0.592	0.42	0.592
q43	0.6	0.75	0.54	0.701
q44	0.52	0.684	0.54	0.701
q45	0.92	0.958	0.92	0.958
q46	0.8	0.889	0.76	0.864
q47	0.92	0.958	0.92	0.958
q48	0.98	0.99	0.98	0.99
q49	0.94	0.969	0.94	0.969
q50	0.42	0.592	0.42	0.592

VII. SIGNIFICANCE OF THE RESEARCH WORK

Being an academician, I preferred to work in the Academic Domain. I have selected some 50 queries applicable in the Academic Domain. The queries are focused on retrieving the books in different fields of computer science, research papers in different fields of electronics and computers, resources in the respective fields and pdf in various fields of computers. I have retrieved Google results for those queries. For a single query, I have retrieved first 100 results. Those 100 urls were submitted to CiteUlike for retrieving metadata (i.e. tags). In totality, I have retrieved 5000 urls and the tags corresponding to those urls with their weights. The Google results were re-ranked corresponding to those queries using my algorithm.

After this, I had applied JEKS algorithm on 5000 urls(corresponding to 50 queries). My results of JEKS algorithm for normalized DCG for 40 queries (out of 50 queries) were higher than Google. Our algorithm acquires higher precision in comparison to Google throughout the varying levels of K for all the 50 queries.

We computed the values for precision, recall and F1-score for our algorithm and Google .These values are calculated for all the queries. These values are calculated for their corresponding top 50 results for Rel>=2 for all the 50 queries.

We observed that the value of recall for our algorithm and Google remain at 1 as we have re-ranked the top 100 results of Google for each query. The value of precision and F1-score are calculated and it has been observed that we are getting better results.

So, the significance of my research work is that a better ranking system has been developed using my algorithm for retrieving the results in academic domain. The results can be extended to include more queries.

VIII. Relevance of My Research Work

The relevance of the research work is that the entire work has been done using semantic tags from CiteULike(which provides tags in a fully uncontrolled environment). The algorithm is entirely based on tags, which are the essence of semantic web. So, it can be taken as an application or a web service in Academics Domain using semantic web. The algorithm can be extended for more queries.

IX. Conclusion

In this paper, we have analyzed some existing ranking methods and proposed a new algorithm based on the previous methods. Semantic tag of a web page is the metadata information associated with it and depicts a lot about the information associated with it. The match degree between user's real query intent and web page content is determined by calculating the similarity between query and web page tag.

We have proposed the new algorithm using the already existing semantic web algorithm which basically calculates the weighted score of the tags. We have utilized the IDF feature of TFIDF algorithm to improve the semantic web which uses tags. In addition to this, we have used a weighting score. In experiments, we have collected the data from Citeulike and implemented the above algorithm. The relevance scores to the different web links have been given by a group of users. Comparing with Google search results, we find that JEKS algorithm acquires better ranking results, and can put more relevant results in front. Our algorithm acquires higher values of DCG for 40 queries when compared to Google. Our algorithm acquires higher precision in comparison to Google throughout the varying levels of K for all the 50 queries.

In the future work, we will further improve the algorithm. We will consider combining with the search engines user logs, and mining out information repeated to user's query, such as the click information, the browse information and so on. The algorithm can be further enhanced by adding these effects. The algorithm can be extended to include more queries.

References

[1] Cilibrasi, R.L., & Vitanyi, P.M.B. (2007) The Google similarity distance, in IEEE Transactions on Knowlegde and Data Engineering, vol. 19, no.3.

[2] Farooq, U., Kannampallil, T.G., & Song, Y. (2007) Evaluating Tagging Behaviour in Social Bookmarking Systems: Metrics and design heuristics, *in the international ACM Conference on Supporting Group Work*.

[3] Gautam, J., & Kumar, E. (2012) An Improved Framework for Tag-Based Academic Information Sharing and Recommender System, in *Proc. of the World Congress on Engineering*, Vol. 2, 2012, 845-850.

[4] Gracia, J., & Mena, E. (2012) Semantic Heterogeneity Issues on the Web, *IEEE Internet Computing*, pages 60-67.

[5] Halpin, H., & Lavrenko, V. (2011) Relevance feedback between hypertext and Semantic Web search, *Journal of Web Semantics*, vol. 9, 2011, pages 474-489.

[6] Jiang, H., Hu, X., Li, P., & Wang S.(2009) An improved method of term weighting for text classification, in *International Conference on Intelligent Computing and Intelligent Systems*, IEEE, Vol.1, 2009, pages 294-298.

[7] Jin Y., Lin Z., & Lin H., The Research of Search Engine Based on Semantic Web, in *proc. of International Symposium on Intelligent Information Technology Application Workshops (IITAW), IEEE*, 2008, pages 360-363.

[8] Jomsri P., Sanguansintukul S. & Choochaiwattana W., A Framework for Tag-Based Research Paper Recommender System: An IR Approach, in *proc. of the 24th International Conference on Advanced Networking and Applications Workshops, IEEE*, 2010, pages 103-108.

[9] Kaczmarek, A.L. (2011) Interactive Query Expansion with the Use of Clustering-by-Directions Algorithm, *IEEE Transactions on Industrial Electronics*, VOL. 58, No. 8, pages 3168-3173.

[10] Lee, M., Kim, W., & Park, S. (2012) Searching and ranking method of relevant resources by user intention on the Semantic Web, Expert Systems with Applications, vol. 39, pages 4111- 4121.

[11] Leung, K.W.T., & Lee, D.L. (2010) Deriving concept-based user profiles from search engine logs, in *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 7.

[12] Lu C., Hu X., & Park J. (2011) Exploiting the Social Tagging Network for Web Clustering, *(Systems, Man, and Cybernetics – Part A: Systems and Humans)*, vol. 41, pp. 840-852.

[13] Maredj A., & Tonkin N. (2013) Semantic Adaptation of Multimedia-Documents, International Arab Journal of Information Technology, vol. 10, No. 6, pages 579-586.

[14] Osinski, S., & Weiss, D. (2005) A Concept-Driven Algorithm for Clustering Search Results, *IEEE Intelligent Systems*, Volume 20, Issue 3, pp. 48-54.

[15] Shaikh, F., Siddiqui, U.A. & Shahzadi, I. (2012) Semantic Web based Intelligent Search Engine, in *proc. of International Conference on Information and Emerging Technologies*, pp. 1-5.

[16] S. Lu, X. Li, S. Bai & S. Wang., (2000) An improved approach to weighting terms in text. *Journal of Chinese Information Processing*, 14(6), pp. 8-13.

[17] Shenliang X., Shenghua B. and Fei, B., *Exploring Folksonomy for Personalized Search*, in proc. of the 31st annual international ACM SIGIR conference on Research and Development in information retrieval, 2008, pp. 155-162.

[18] Yoo, D. (2012) Hybrid Query Processing for Personalized Information Retrieval on the Semantic Web, *Knowledge-Based Systems*, vol 27, pages 211-218.

[19] Wu, X., Zhang, L., & Yu Y. (2006) Exploring Social Annotations for the Semantic Web, in *proc. of the 15th International Conference on World Wide Web (WWW 06), ACM*, pages 417-426.

[20] Zhanguo, M., Jing, F., Liang, C., Xiangyi H., & Yanqin, S. (2011) An improved approach to terms weighting in text classification , in *proc. of the International Conference on Computer and Management*, IEEE, pages1-4.

[21] Zhao, C., & Zhang, Z. (2010) A New Keywords Method to Improve Web Search, in *12th International Conference on High Performance Computing and Communications, IEEE*, pages 477-484.

[22] Zhong, N., Li, Y., & Wu. S.T. (2012) Effective Pattern Discovery for Text Mining, IEEE Transactions on Knowledge and Data Engineering, vol. 24, no1.