Journal of King Saud University – Computer and Information Sciences (2012) **24**, 71–81

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

ORIGINAL ARTICLE

An adaptive meta-search engine considering the user's field of interest

Hamid Hassanpour [a,*], **Farzaneh Zahmatkesh** [b]

[a] *Shahrood University of Technology, Iran*
[b] *Mazandaran University of Science and Technology, Iran*

Received 4 May 2011; revised 17 August 2011; accepted 12 October 2011
Available online 7 November 2011

KEYWORDS

Clustering;
Meta-search engine;
Ranking;
Search relevance;
Social information

Abstract Existing meta-search engines return web search results based on the page relevancy to the query, their popularity and content. It is necessary to provide a meta-search engine capable of ranking results considering the user's field of interest. Social networks can be useful to find the users' tendencies, favorites, skills, and interests. In this paper we propose MSE, a meta-search engine for document retrieval utilizing social information of the user. In this approach, each user is assumed to have a profile containing his fields of interest. MSE extracts main phrases from the title and short description of receiving results from underlying search engines. Then it clusters the main phrases by a Self-Organizing Map neural network. Generated clusters are then ranked on the basis of the user's field of interest. We have compared the proposed MSE against two other meta-search engines. The experimental results show the efficiency and effectiveness of the proposed method.

1. Introduction

A meta-search engine is a searching tool that employs the search results of other search engines. They disregard the user's preferences or field of interest in searching. Existing meta-search engines return web search results based on the page relevancy to the query, their popularity and content. Different people may look for different resources whilst they utilize the same query or keywords, hence employing a user-independent approach in ranking the web resources may not satisfy the user. Fig. 1 shows a search example. There are two users, A and B. User A is looking for information about the Mac OS X 10.2, while user B is interested in jaguar cars. They issue the same query, "jaguar" on Ixquick meta-search engine and obtain the default ranked list shown in Fig. 1. This ranking list does not contain needed information for user A. In addition, it is not satisfactory for user B, because it contains documents about the jaguar cats too. Thus, a user-aware search system is desirable for improving search effectiveness.

This research explores how user information can enhance search result of a meta-search engine. User information can be obtained from the social networks. Traditional approaches to search include scoring documents based on a set of

* Corresponding author.
E-mail addresses: h_hassanpour@yahoo.com (H. Hassanpour), zahmatkesh.farzaneh@gmail.com (F. Zahmatkesh).

Peer review under responsibility of King Saud University.
doi:10.1016/j.jksuci.2011.10.004

Production and hosting by Elsevier

Figure 1 First ten search results for query "jaguar" on Ixquick.

keywords or using the link structure across documents to infer quality and relevance. These approaches attempt to optimally match keywords to documents with little or no information about the user and no information about his or her network. In reality, users are involved in different social communities, and are increasingly engaged in social networks through online services like Facebook, Flickr, and YouTube. The social network can be a unique reflection of the user, and can be used to find the users' tendencies, favorites, skills, and interests.

In this paper, we propose a meta-search engine named MSE. The MSE uses a social information-based approach of search, where each user corresponds to a profile containing his or her field of interest. The objective is to leverage on social information to rank the documents. A possible solution to this problem is to online cluster search results into different categories, and to enable users to identify their required category at a glance. In order to evaluate MSE, we conduct a comparative performance study among MSE and two other existing meta-search engines. The results indicate that MSE is an effective and efficient meta-search engine for document retrieval.

In the next section, we review related works. We present the proposed MSE system in Section 3, together with the whole algorithm description and report results of experimental study in Section 4. The future work is discussed in Section 5. Finally, the conclusions are drawn in Section 6.

2. Related work

Personalized ranking search results algorithms for search engines have been proposed to include various types of user information in ranking (Micarelli et al., 2007). To enhance ranking performance and improve search results, algorithms use such information as a user's search context, geographical location and searching histories, click-through logs, and personal bookmarks. Some algorithms consider the information needs of the user's friends (Dalal, 2007; Mislove et al., 2006). However, these algorithms largely focus on local activities of the user, and fail to embrace the large social contexts of the user.

CYBER, a CommunitY Based sEaRch engine, was proposed for information retrieval utilizing community feedback information in a DHT network. In CYBER, each user is associated with a set of user profiles that capture his/her interests. Likewise, a document is associated with a set of profiles – one for each indexed term. A document profile is updated by users who query on the term and consider the document as a relevant answer. Thus, the profile acts as a consolidation of users' feedback from the same community, and reflects their interests. In this way, as one user finds a document to be relevant, another user in the same community issuing a similar query will benefit from the feedback provided by the earlier user. Hence, the search quality in terms of both precision and recall is improved (Li and Shou, 2008).

Researchers explored how information contained in the structure of the social graph can improve search result relevance on social networking websites. This study tests whether the macrostructure of an interpersonal social network is of value for improving the ordering of results for profile search queries on a social networking website. The term *search* is used here to mean web search within social networking websites (Haynes, 2009).

A ranking framework, Social Network Document Rank (SNDocRank), was proposed by researchers. SNDocRank considers both document contents and the relationship

between a searcher and document owners in a social network. This method combines the traditional tf-idf ranking for document contents with Multi-level Actor Similarity (MAS) algorithm to measure to what extent document owners and the searcher are structurally similar in a social network (Gou et al., 2010a; Gou et al., 2010b).

3. Research methodology

The search problem is defined as using a set of keywords in a search query, as well as characteristics of the user, to identify the most relevant results. To solve the mentioned problem, we propose designing a meta-search engine for document retrieval utilizing social information of the user. In this approach, each user is associated with a user profile that contains his interests. Given a query and the ranked list of documents returned by underlying web search engines, our method first parses the entire list of titles and short descriptions of results and extracts main phrases from them. Then it calculates a vector for each main phrase with parameters like Phrase Frequency/Inverse Document Frequency, Phrase Length, Phrase Independence, Intra and Inter-Cluster Similarity. Then it clusters the vectors by unsupervised learning algorithm. Generated clusters are then ranked on the basis of the user's field of interest. The more similar cluster label to the user's field of interest gets the higher rank.

3.1. Clustering approach

Clustering methods do not require pre-defined categories as in classification methods. Thus, they are more adaptive for various queries. Nevertheless, clustering methods are more challenging than classification methods because they are conducted in a fully unsupervised way. Organizing web search results into clusters quicken browsing search results. The technique proposed in this research is more suitable for clustering web search results because it emphasizes the efficiency of identifying relevant clusters for web users. The clusters are ranked according to the similarity between clusters' labels and user's field of interest, thus the more likely clusters required by users are ranked higher.

In this research, we have employed self-organizing feature map (SOM) competitive neural network to cluster main phrases' vectors. SOM is the type of learning algorithms where a system is provided with sample inputs only during the learning phase, but not with the desired outputs. The aim of the system is to organize itself in such a way to find correlation and similarities between data samples (Beale and Jackson, 1990).

In (Georgakis and Li, 2005; Lagus et al., 2004), the researchers have used an ensemble of SOMs to boost the performance of the document organization and retrieval. The SOMs were used to partition the document repository into clusters of semantically related documents. It has been reported that this approach has a low error rate in document clustering. In another research, it has been demonstrated that SOM neural network clustering methodology is superior to the hierarchical popular clustering methods. These results have been conceived following a test performed on 252 data sets with various levels of imperfections that include overlapped dispersed data, outliers, irrelevant variables, and nonuniform cluster densities (different sized populations (Mangiameli et al., 1996).

3.2. Data

Facebook is a large popular social networking site. Since the data in this site tends to be rich on social networks, we have chosen it to collect the training and test dataset. The MSE obtains web search results from three underlying search (source) engines including Bing, Google and Yahoo.

3.3. Phases of MSE algorithm

3.3.1. Receiving web search results from source engines

The designed MSE accepts username and query inputs from user and passes the query to three underlying search engines. First fifty results are retrieved from each search engine. Each result is considered as an object with six attributes including Title, Description, URL, Source Engine Name, Result Rank and Result's Set of Main Phrases. Fig. 2 shows an example of a retrieved web search results. Title, Description, and URL are set using the corresponding tags. Source Engine Name is name of the source engine which the result is retrieved from. Result Rank is the rank of result in order of results. Result's Set of Main Phrases is further explained.

3.3.2. Removing duplicate results

Since meta-search engine receives results from more than one source engine, it has to eliminate duplicates.

3.3.3. Extracting main phrases from title and short description of any result

Instead of downloading whole page result, the title and short description of each result are parsed in order to filter out stop words and then to extract main phrases. Stop word are a commonly used words such as "the" that are ignored by the majority of search engines when indexing entries for searching or when retrieving them as the result of a search query to save space and time. Stop words are deemed irrelevant for searching purposes. A main phrase is an n-gram, where $n \leqslant 3$, with frequency greater than two. An n-gram is a subsequence of n words from a sequence of words. We apply stemming to each

```
<web:WebResult>
  <web:Title>Welcome to the □Apple□ Store - □Apple□ Store (U.S.)</web:Title>
  <web:Description>Experience the wide world of □Apple□ at the □Apple□ Store. Shop for □Apple□ computers, compare iPod and iPhone models,
    and discover □Apple□ and third-party accessories, software, and ...</web:Description>
  <web:Url>http://store.apple.com/us</web:Url>
  <web:DisplayUrl>store.□apple□.com</web:DisplayUrl>
  <web:DateTime>2011-02-02T21:56:00Z</web:DateTime>
</web:WebResult>
```

Figure 2 An example of a retrieved web search result.

word within a main phrase using Porter's algorithm. For example, consider the statement "Apple spreads whole world" as a title. Then, main phrases that can be extracted from this title are; "Apple", "Apple spread", "Apple spread world", "spread world", and "world".

3.3.4. Calculating main phrases' vectors

We calculate five parameters (Kaifeng et al., 2008) for each main phrase and produce a vector for it.

(1) Phrase Frequency/Inverse Document Frequency (PFIDF): this weight is a statistical measure used to evaluate how important a word is to a document. PFIDF property of a main phrase (mph) is defined as:

$$PFIDF(mph) = PF(mph) \cdot \log \frac{|D|}{DF(mph)} \quad (1)$$

where $PF(mph)$ is the total count of main phrase in all the document set D and $DF(mph)$ is the number of documents containing the main phrase. More frequent phrases are more likely to be better candidates of cluster labels; while phrases with higher document frequency might be less informative to represent a distinct label.

(2) Phrase Length (LEN): the LEN property is simply the count of words in a main phrase. For example the LEN property of the main phrase "Apple spread" is two.

(3) Intra-Cluster Similarity of Phrase (Intra-CS): if a phrase is a good representation of a single topic, the documents which contain the phrase will be similar to each other. We use Intra-CS parameter to measure the content compactness of documents containing the phrase. First, we convert each document into a vector: $d = (x_1, x_2, \ldots)$. Each component of the vector represents a distinct uni-gram and is weighted by PFIDF of this uni-gram. Intra-CS property of main phrase is the average cosine similarity between its associated documents and its centroid (cen). Intra-CS and cen parameters are defined as:

$$Intra - CS(mph) = \frac{1}{PF(mph)} \cdot \sum_{d \in D \& mph \in d} \times \cos(d, cen(mph)) \quad (2)$$

$$cen(mph) = \frac{1}{PF(mph)} \cdot \sum_{d \in D \& mph \in d} \mathrm{d} \quad (3)$$

(4) Inter-Cluster Similarity of Phrase (Inter-CS): inter-CS property of main phrase is the average cosine similarity between its associated documents and the remainder of the documents. Inter-CS parameter is defined as:

$$Inter - CS(mph) = \frac{1}{PF(mph)} \cdot \sum_{d \in D \& mph \in d} \sum_{d \in D \& mph \notin d} \times \cos(d, d) \quad (4)$$

(5) Phrase Independence (IND): a main phrase is independent when the entropy of its context is high. We confirm independence of main phrase when its left and right contexts are random enough. The followings are the equations for IND and INDl (or INDr) which is the independence value for left (or right) context of main phrase. For example left and right contexts of main phrase "Apple spread" are "Apple" and "spread".

$$IND(mph) = \frac{INDr(mph) + INDl(mph)}{2} \quad (5)$$

$$INDl(mph) = -\sum_{t \in l(mph)} \frac{PF(t)}{PF(mph)} \cdot \log \frac{PF(t)}{PF(mph)} \quad (6)$$

3.3.5. Clustering main phrase vectors by SOM

To provide a suitable representation of input data, produced vectors are first normalized. We arrange the neurons of our neural network in a random two-dimensional topology. In this network input layer is two dimensional and output layer is three dimensional. We use the Link distance function in order to calculate distances from a particular neuron to its neighbors. The link distance from one neuron is the number of links or steps that must be taken to get to the neuron under consideration. For example returned web results from Google search engine in case of query "apple" are clustered by SOM. Figs. 3 and 4 show the produced network topology and population of each cluster before and after training process. Fig. 5 shows SOM Neighbor Weight Distances. In Fig. 5, the blue hexagons represent the neurons. The red lines connect neighboring neurons. The colors in the regions containing the red lines indicate the distances between neurons. The darker colors represent larger distances, and the lighter colors represent smaller distances. A band of dark segments crosses from the up region to the down region. The SOM network appears to have the data be clustered into two distinct groups. Here is pseudocode of our implementation of SOM in MATLAB.

```
mymean = mean(mainphrasesvectors);
for each i
mainphrasesvectors(i,:) = mainphrasesvectors(i)-mymean;
dimensions = [23];
positions = randtop(dimensions);
```

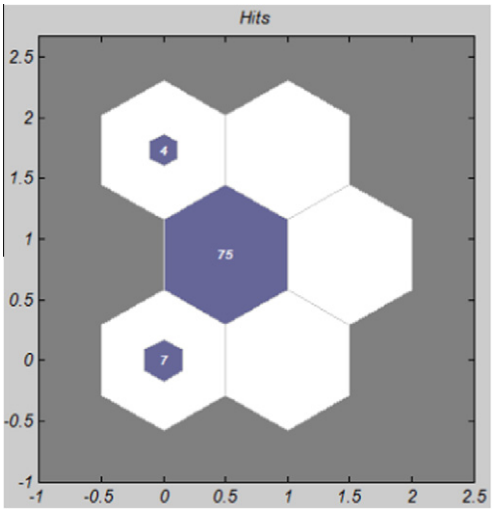

Figure 3 SOM topology and population of each cluster before training process.

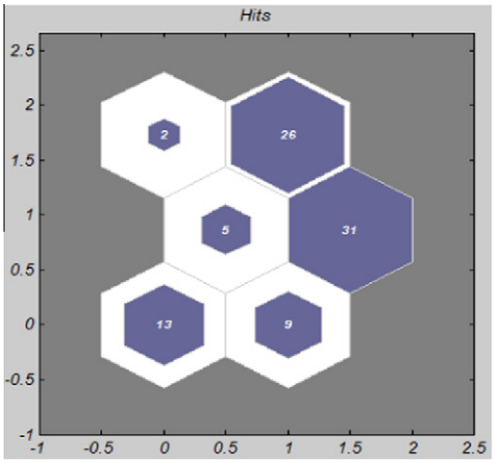

Figure 4 Population of each cluster after SOM training process.

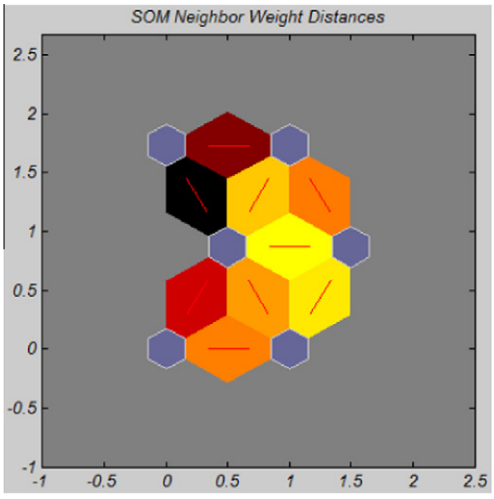

Figure 5 SOM Neighbor Weight Distances.

```
net = newsom(mainphrasesvectors,dimensions);
trainednet = train(net,simpleclassinput);
```

We have earlier demonstrated some reasons for choosing SOM as clustering algorithm. In order to be sure about the correctness of this selection, we cluster returned web results from Google search engine in case of 20 arbitary queries by K-means and GA. Then, we compare SOM, K-means, and GA. Here, we report standard deviations within each generated cluster of main phrases in each clustering approaches in Table 1. In this table, values are the average of standard deviation from 20 clustering process. The lower standard deviation values of SOM in comparison with K-means, and GA, shows the little dispersal of main phrases arranged in one cluster. Therefore, considering the input data, SOM is a reliable clustering algorithm in this research.

Table 1 Standard deviations of SOM, K-means, and GA clustering methods.

STD of the most relevant cluster to query	Clustering method
5.9112	SOM
6.087	K-means
6.9432	GA

3.3.6. Labeling generated clusters

Every generated cluster consists of main phrases. Among the main phrases arranged in a certain cluster, the most frequent one is considered as the label of that cluster. Outlier main phrases are eliminated.

3.3.7. Fetching user's field of interest

The user is able to register in MSE. To do so he is supposed to enter a username and select his field or fields of interest. There is a list of some pre-defined fields of interest available for user to choose. If the list does not contain his field of interest, he can enter his own. After registration he can log in to MSE. Hence, his profile is available to fetch his field of interest.

In case of each field of interest there is a list of pre-defined queries available for user. In case of new queries or new fields of interest MSE is able to update its list.

3.3.8. Computing similarity between each cluster label and the user's field of interest

Similarity between two strings is a confidence score that reflects the relation between the meanings of two strings. The similarity is calculated in three steps:

a. Partitioning each string into a list of tokens (words);
b. Computing the similarity between tokens using a string edit distance matching algorithm; Levenshtein distance is the total cost of transforming one string into another using a set of edit rules, each of which has an associated cost. Edit distance is obtained by finding the cheapest way to transform one string into another. Transformations are the one-step operations of insertion, deletion and substitution. In the simplest version substitutions cost about two units except when the source and target are identical, in which case the cost is zero. Insertions and deletions costs half that of substitutions (Yujian and Bo, 2007).
c. Computing the similarity between two token lists; we capture the similarity between two strings by computing the similarity of those two token lists, which is reduced to the bipartite graph matching problem (Tardos and Kleinberg, 2006). Given a graph $G(V, E)$, G can be partitioned into two sets of disjoint nodes X and Y such that every edge connects a node in X with a node in Y. X is the set of the first list of tokens. Y is the set of the second list of tokens. E is a set of edges connecting between each couple of vertex (X, Y), the weight of each edge which connects a node of X to a node of Y is computed in previous step. The task is to find a subset of node-disjoint edges with maximum total weight. This total weight is considered as the similarity between cluster label and the user's field of interest.

3.3.9. Classifying results on the basis of the generated clusters of main phrases

The results are assigned to relevant main phrases to form final classes of results. A result cannot join more than one class. Hence, if a result is source of two main phrases belonging to

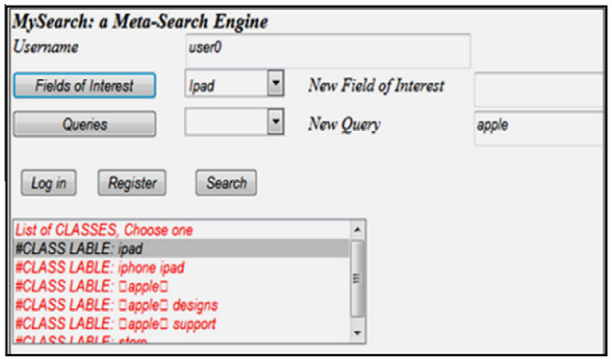

Figure 6 Output of a sample search process in MSE.

two separate clusters, the cluster with higher similarity is preferred.

3.3.10. Ranking results within a class based on the popularity rank of the results

For each result we assign a popularity rank that is the result's rank in its source engine (Result Rank attribute). If the result is obtained from more than one source engine, the Result Rank is average of the result's ranks in its multi-source engines.

3.3.11. Ranking classes of results on the basis of similarity parameter

The produced classes are ranked on the basis of the descending order of computed similarity.

3.3.12. Showing results to the user

At last the sorted classes and their labels are listed for user. There may be users with multi fields of interest. In case of such users MSE repeats the eights and eleventh phases of its algorithm and returns distinct directories of result classes. Classes within each directory are ranked on the basis of one of the user's fields of interest. User is able to select the required directory, and go through it. Each directory label is the same as the first class's label in that directory.

Fig. 6 shows the graphical user interface and output of MSE. Consider that one user who is interested in "Ipad" enters the query "apple" into MSE. As you can see in Fig. 6, six classes of web results (clusters of main phrases) including "ipad", "iphone ipad", "apple", "apple designs", "apple supports", and "store" are shown to the user in red, and he can choose one of them to see the results within each class in blue.

This example can help us to be convinced about the performance of MSE. As Fig. 6 shows the first class lable is "ipad". We have gone through all classes and have examined all results within them. The results in first class, "ipad" are all about apple ipads. Each result within second class named "iphone ipad" contains information about both iphone and ipad. The third class, "apple" only contains results speaking about all of apple products. The results within fourth and fifth classes talk about different designs of apple products and support conditions. All results in last class are about apple fruit.

4. Experimental evaluation

We conduct experiments to evaluate MSE against two meta-search engines, Ixquick and Seekky. Unfortunately, not all meta-search engines are accessible in all countries because of political issue. In order to choose comparative meta-search engines with MSE, we asked fifty different people with different jobs. Most of them use Ixquick and Seekky.

4.1. Experimental Setup

We have considered 200 queries containing general terms or entity names whithin 1 day query log from Google search engine. We have chosen 100 queries as training dataset. After extracting main phrases and clustering them using SOM algorithm the neural network was trained.

We used breath-first search strategy to crawl the social network data. We first start with five users randomly chosen with different fields of interest, and obtained their friends information. Then we used these friends as the new centers and fetch the friends from these centers. Information contained in the 'Info' page of members was collected from facebook. This page contains a part named "Activities and Interests". This process was iterative and stopped once a set of 100 different fields of interest was prepared.

We used the rest of 100 queries as test dataset. These queries were listed in Table 2. We call them normal queries against

Table 2 One hundred queries selected from Google's query log.

Entity names				General terms			
Sony	Nokia	World War2	Moon	Trip	Joke	Map	Teacher
Porsche	Egypt	Hollywood	Rhine	Chat	TV	Health	College
Disney	Canada	Panasonic	Libya	Resume	Game	Sport	Story
Obama	Sun	Himalaya	Earth	Time Zone	Music	Planet	Pain
Gandhi	Niagara	Persepolis	Prof. Zadeh	Design	Flower	Friend	Analysis
Nike	Europe	Real Madrid	LG	Implement	Test	Worker	Diagram
Al Pacino	Baikal	Federer	D&G	Student	Body	Tree	Radio
Bruce Li	Troy	Ronaldinho	Mourinho	Equipment	Thing	Exam	Toy
Titanic	China	Ferdowsi	Pixar	Novel	Help	Play	Business
Gaddafi	Casio	Aboureihan	NBA	Message	City	Phone	Country
Bachchan	Eiffel	Zakaria Razi	Jurassic	Backup	Lake	War	Actor
Beckham	Hafez	Schumacher	Lamborghini	Competition	Watch	Star	Waterfall
				Bollywood	MIT	Clock	Tower

challenging queries. A challenging or ambiguous query is a query that is common in two distinct contexts. For instance, ''apple'' is a fruit name and a brand name for laptops.

We have initialized one hundred user profiles. Initializing a user profile is straightforward. A user profile contains username and his/her field or fields of interest. To test MSE in case of ambiguous queries we prepare a list of 20 challenging queries. For these queries two different fields of interest and hence two distinct user profiles can be considered. So, another 40 user profiles were initialized. The challenging queries and their corresponding distinct fields of interest are listed in Table 3.

We use these 140 user profiles to evaluate our proposed MSE against Ixquick and Seekky.

4.2. Evaluation measures

We use two standard metrics to evaluate the three meta-search engines: precision and recall. In information retrieval, precision is the fraction of returned instances that are relevant, while recall is the fraction of relevant instances that are returned.

Precision = #relevant-documents-returned/#returned-documents = P(relevant|returned)

$$\mathrm{P} = \frac{\mathrm{TP}}{(\mathrm{TP} + \mathrm{FP})} \tag{7}$$

Recall = #relevant-documents-returned/#relevant-documents = P(returned|relevant)

$$\mathrm{R} = \frac{\mathrm{TP}}{(\mathrm{TP} + \mathrm{FN})} \tag{8}$$

TP and FP rates are described in Table 4. In even simpler terms, a high recall means you have not missed anything but you may have a lot of useless results to sift through (which would imply low precision). High precision means that everything returned was a relevant result, but you might not have found all the relevant items (which would imply low recall). We use precision (P) at top N results (P@10).

Another metric, priority is suggested here. This metric is trying to emphasize to the place of relevant returned documents in the list of returned documents. Given a query, let the set of returned documents be K and K set of relevant returned documents (K is a subset of K). Let k be a relevant returned document and r(k) be the rank of k in the returned list of documents, then the priority (Pri) metric is defined as:

$$\mathrm{Pri} = \sum_{k \in K} (M - r(k) + 1) \tag{9}$$

If k is the first returned result, then $r(k) = 1$. We assume M as the number of returned documents, among which the priority is calculated.

To justify priority metric we offer an example here. In case of a search process, first 50 returned documents ($M = 50$) are considered to calculate priority. Assume 20 of these 50 are relevant, and the remained 30 are irrelevant to the query of search. The order of these documents is important to user. In best order, all relevant ones place the first rank to 20th rank. This way time is saved for user. This way the priority is:

$$\mathrm{Pri} = (50 - 1 + 1) + (50 - 2 + 1) + (50 - 3 + 1) + \ldots + (50 - 20 + 1) = 810$$

In worst order, all relevant ones place the 31st rank to 50th rank. This way the priority is:

$$\mathrm{Pri} = (50 - 31 + 1) + (50 - 32 + 1) + (50 - 33 + 1) + \ldots + (50 - 50 + 1) = 210$$

The priority helps to find out, how up relevant documents are placed. Upper place for a relevant document (and higher priority value) shows the effectiveness of the search engine. If there is no relevant document among M documents, priority equals to 0. If all of M returned documents are relevant priority equals to $\frac{\mathrm{M} \cdot (\mathrm{M}+1)}{2}$.

Table 3 Twenty ambiguous queries and distinct fields of interest.

Search ID	Query	Field of interest
A1	Jaguar	Mac OS X 10.2
A2	Jaguar	Cars
B1	Apple	Trees
B2	Apple	Ipods
C1	Saturn	Astronomy
C2	Saturn	Mythology
D1	Jobs	Careers
D2	Jobs	Inventors
E1	Jordan	Geography
E2	Jordan	Fashion
F1	Tiger	Golf
F2	Tiger	Cats
G1	Trec	Research
G2	Trec	Environment
H1	Ups	Chain Management
H2	Ups	Stereography
I1	Quotes	Commercial Offer
I2	Quotes	Famous Sayings
J1	Matrix	Movies
J2	Matrix	Hairstyles
K1	Bush	Forest
K2	Bush	Policy
L1	Dell	Laptop
L2	Dell	Valley
M1	Orange	Color
M2	Orange	Fruit
N1	Clinton	Town
N2	Clinton	Presidents
O1	Bar	Wine
O2	Bar	Lawyer
P1	DI	Police
P2	DI	Military
Q1	Lincoln	Town
Q2	Lincoln	Presidents
R1	Bat	Baseball
R2	Bat	Animals
S1	Kitty	Girls
S2	Kitty	Cats
T1	Ford	River
T2	Ford	Policy

Table 4 Description of TP and FP rates.

	Relevant	Irrelevant
Returned	True positive (TP)	False positive (FP)
Not returned	False negative (FN)	True negative (TN)

Table 5 Average values of precision, recall, and priority among 100 normal queries.

Average value	MSE	Ixquick	Seekky
Precision	0.855	0.857	0.801
Recall	0.79	0.73	0.66
Priority	46.15	46.01	40.99

4.3. Experimental results

Normal queries listed in Table 2 were entered into MSE, Ixquick, and Seekky. In case of these queries, the precision, recall and priority parameters calculated for MSE were almost equal to those of Ixquick and Seekky. The average of precision, recall, and priority are listed in Table 5. The difference is noticeable in case of 20 challenging queries. The difference confirms the preference of MSE over Ixquick and Seekky as the calculated average values are listed in Table 6. The values listed in Tables 5 and 6 are calculated among first ten results (only the first result page) returned by the three meta-search engines. This way the performance and time-saving properties of the three meta-search engines can be comparable. Users would rarely go through the next pages returned by a search engine. So assuming $M = 10$, the range of priority is $0 \leqslant \text{Pri} \leqslant 55$. Higher precision, recall, and priority values of MSE against Ixquick and Seekky confirm the efficiency and effectiveness of this meta-search engine.

Given challenging queries, Ixquick and Seekky have problem figuring out the desired information context of user who initiates the query. Hence, they only try to return most popular relevant documents. They work well only in case of a user aiming these popular documents. Using Ixquick or Seekky, user may be forced to go through next pages to get wanted data, whilst using MSE user does not need to go through even the second page and it is one of the benefits of considering user's field of interest in search process.

Table 6 Average values of precision, recall, and priority among 20 challenging queries.

Average value	MSE	Ixquick	Seekky
Precision	0.4995	0.3655	0.321625
Recall	0.458	0.34275	0.301
Priority	24.825	18.85	17.95

Figs. 7–9 show respectively the precision, recall, and priority values among first 10 returned results by MSE, Ixquick, and Seekky per each 20 challenging queries listed in Table 3. They are the average values calculated among ten search processes for each search ID. For example consider Search IDs J1 and J2. Ixquick and Seekky return the same result set in case of query, "Matrix" in response to two users, one interested in "Movies" and the other interested in "Hairstyles". But MSE first fetches the user's interest from his profile, clusters the received results and then ranks the generated clusters on the basis of user's field of interest. Almost in all, bars of MSE are taller than the other two ones (the diagram of MSE is placed upper).

Table 7 contains the average value of calculated precision, recall, and priority parameters for normal and challenging queries (the whole test dataset) returned by MSE, Ixquick, and Seekky. Table 7 confirms that MSE can satisfy user more than Ixquick and Seekky.

In second phase of our evaluation, we choose two famous meta-search engines, Dogpile and Vivisimo. It needs to be mentioned that the personalization approach used in Dogpile and Vivisimo is Adaptive Result Clustering. These two meta-search engines organize results into folders by grouping pages with the same topics together. In fact the pages are grouped in the meta-search engines' database. This approach first needs a complete topic classification. The users are able to customize the results by navigating and choosing selected clusters based on their needs (Micarelli et al., 2007).

Using test dataset, we calculated three above mentioned metrics for Dogpile and Vivisimo in comparison with MSE. The results show a very small and ignorable difference between MSE and those two meta-search engines. We think Dogpile and Vivisimo can be improved if they merge their approach with ours, namely a Content Result Clustering approach.

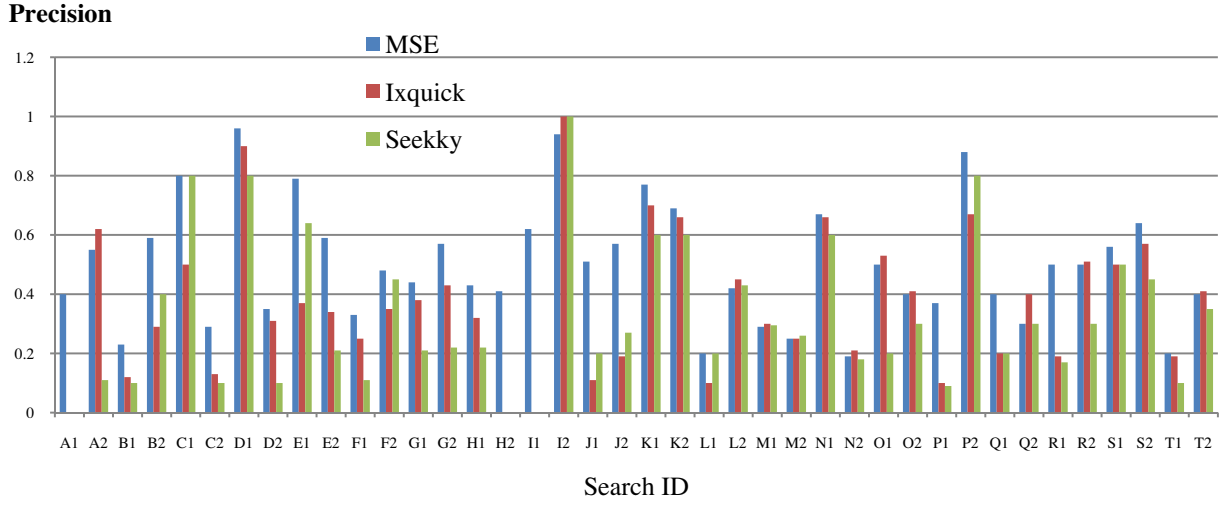

Figure 7 The precision values of three meta-search engines per 20 challenging queries.

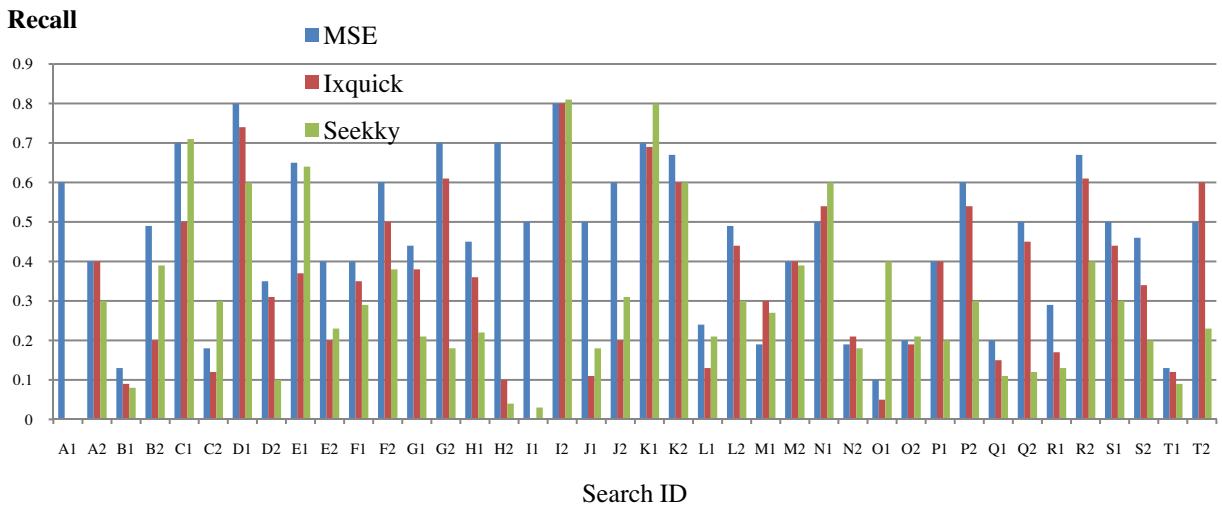

Figure 8 The recall values of three meta-search engines per 20 challenging queries.

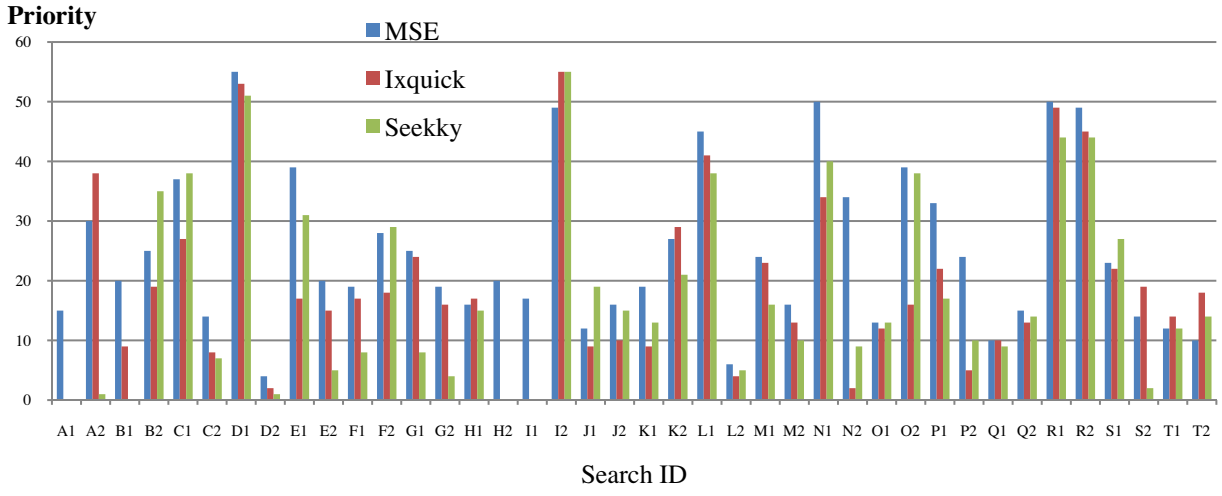

Figure 9 The priority values of three meta-search engines per 20 challenging queries.

Table 7 Average precision, recall, and priority of 20 challenging and 100 normal queries.

	MSE	Ixquick	Seekky
Average precision	0.753429	0.716571	0.664036
Average recall	0.695143	0.619357	0.557429
Average priority	40.0571	38.25	34.40714

It is to be noted that the "higher ranking" values returned by the MSE might sometimes be a deceiving measure since in some cases user interests can fall in a lower ranking results returned by the "lower" search engines (used in the MSE).

4.4. Input document number

We have used the average precision of arbitrary 10 queries on Google to explain the reason why we used the top 50 returned

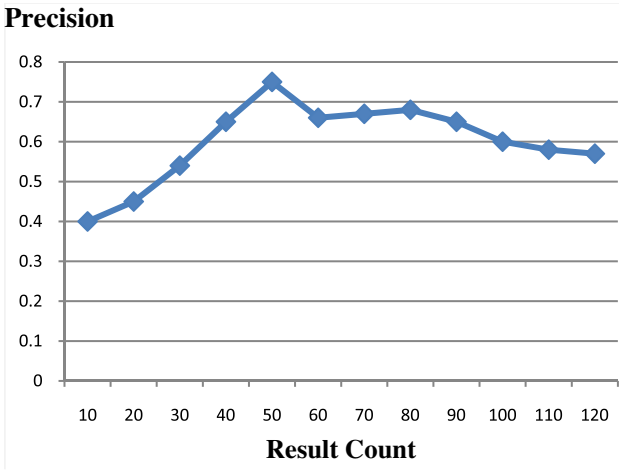

Figure 10 The precision value (P@50) along with result number.

Table 8 Precision value (P@50) along with a single parameter of main phrase.

Parameter	Precision (P@50)
PFIDF	0.42
LEN	0.22
Intra-CS	0.1
Inter-CS	0.11
IND	0.48

results of each source engine, as the experiment result shown in Fig. 10. It is clear that the precision measure arrives at peak when the result count equals to 50. Fig. 10 shows that our algorithm requires a small number of input document to achieve fairly good performance.

4.5. Parameter comparison

To measure the effect and role of each single parameter of a main phrase in result classifying, we use each single parameter to cluster main phrases, and then evaluate the precision of returned results for arbitrary 10 queries. We have executed MSE algorithm five times. Each time, the algorithm is executed on main phrase vectors containing only one of five parameters mentioned in forth phase of the proposed algorithm. The average precisions among top 50 (P@50) results returned by MSE after five executions are shown in Table 8. For example, PFIDF attribute of main phrases can achieve 0.42 precision by itself.

Note that many phrases have the same LEN value, so it cannot be effective enough (P@50(LEN) = 0.22). From Table 8, we can see that each parameter does not work very well alone, but IND and PFIDF are better indicators. Intra and Inter-Cluster are not good indicators. The reason might be that documents are composed of short titles and descriptions, so that the vector space model-based similarity has a high error rate.

5. Future work

The accuracy rate of MSE increases by employing the hierarchical SOM as the clustering algorithm. Hierarchical self-organizing feature maps have two benefits over SOM. First, hierarchical feature maps entail substantially shorter training times than the standard SOMs. The reason is that, there is the input vector dimension reduction on the transition of one layer to the next. Shorter input vectors lead directly to reduced training times. In addition, the SOM training is performed faster. The reason is that, the spatial relation of different areas of the input space is maintained by means of the network architecture rather than by means of the training process. Second, hierarchical feature maps may be used to produce fairly isolated, i.e. disjoint, clusters of the input data than can be gradually refined when moving down along the hierarchy. In its basic form, the SOM struggles to produce isolated clusters. The separation of data items is a rather tricky task that requires some insight into the structure of the input data. Metaphorically speaking, the standard SOM can be used to produce general maps of the input data, whereas hierarchical feature maps produce an atlas of the input data. The standard SOM provides the user with a snapshot of the data; as long as the map is not too large, this may be sufficient. As the maps grow larger, however, they have the tendency of providing too little orientation for the user. In such a case, hierarchical feature maps are advisable as models for data representation (Vicente and Vellido, 2004).

Another recommendation is to consider the ''friendship'' relation between one person and his friends in the social network, to derive some interested topics which have not been explicitly given by the user himself but can be derived based on his friendship information. This would make the current work more powerful and applicable.

6. Conclusions

This research was aimed to design a meta-search engine capable of returning web search results considering user's interests. The presented meta-search engine employs a user-dependent approach in ranking the web resources. The proposed method incorporates the similarity between query and document, and the similarity between the user's interest and the documents to rank search results. The meta-search engine classifies and then ranks the returned results from underlying search engines on the basis of user's field of interest. The results of our experiments show that our approach returns more relevant and higher ranked information in comparison to those ranking methods not considering the user's field of interest. Experimental results verify our method's feasibility and effectiveness.

References

Beale, R., Jackson, T., 1990. Neural Computing: an Introduction. IOP Publishing Ltd., Bristol, UK.

Dalal, M., 2007. Personalized social & real-time collaborative search. In: Proceedings of the 16th International World Wide Web Conference (WWW '07), Alberta, Canada, pp. 1285–1286.

Georgakis, A., Li, H., Gordan, M., 2005. An ensemble of SOM networks for document organization and retrieval. In: Proceedings of International Conference on Adaptive Knowledge Representation and Reasoning (AKRR'05), Espoo, Finland, pp. 141–147.

Gou, L., Zhang, X.L., Chen, H.H., Kim, J.H., Giles, C.L, 2010. Social network document ranking. In: Proceedings of the 10th International annual Joint Conference on Digital Libraries (JCDL'10), Queensland, Australia, pp. 313–322.

Gou, L., Chen, H.H., Kim, J.H., Zhang, X., Giles, C.L, 2010. SNDocRank: a social network-based video search ranking framework. In: Proceedings of the 11th International ACM Conference on Multimedia Information Retrieval (MIR'10), Pennsylvania, USA, pp. 367–376.

Haynes, J., Perisic, I., 2009. Mapping search relevance to social networks. In: Proceedings of the 3rd Workshop on Social Network Mining and Analysis (SNA-KDD'09), Paris, France, pp.1–7.

Kaifeng, X., Li, R., Bao, S., Han, D., Yu, Y., 2008. SEM: mining spatial events from the web. In: Proceedings of the 12th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'08), Osaka, Japan, 393–404.

Lagus, K., Kaski, S., Kohonen, T., 2004. Mining massive document collections by the WEBSOM method. Information Sciences 163, 135–156.

Li, Y., Shou, L., Tan, K.-L., 2008. CYBER: a community-based sEaRch engine. In: Proceedings of the 8th International Conference on Peer-to-Peer Computing (P2P'08), Aachen, Germany, pp. 215–224.

Mangiameli, P., Chen, S.K., West, D., 1996. A comparison of SOM neural network and hierarchical clustering methods. In European Journal of Operational Research 93 (2), 402–417.

Micarelli, A., Gasparetti, F., Sciarrone, F., Gauch, S., 2007. Personalized search on the World Wide Web. In: Lecture Notes in Computer Science, the Adaptive Web: Methods and Strategies of Web Personalization, vol. 4321. Springer, Berlin, pp. 195–230.

Mislove, A., Gummadi, K.P., Druschel, P., 2006. Exploiting social networks for internet search. In: Proceedings of the 5th Workshop on Hot Topics in Networks (HotNets'06), Irvine, Canada.

Tardos, E., Kleinberg, J., 2006. Algorithm Design. Pearson Education, Addison Wesley Publishing, Boston.

Vicente, L., Vellido, A., 2004. Review of hierarchical models for data clustering and visualization, In: Giráldez, R., Riquelme, J.C., Aguilar-Ruiz, J.S., (Eds.), Tendencias de la Minería de Datos en España. Red Española de Minería de Datos.

Yujian, L., Bo, L., 2007. A normalized levenshtein distance metric. In IEEE transactions on pattern analysis and machine intelligence 29 (6), 1091–1095.