Journal of Engineering and Fundamentals
Vol. 2(2), pp. 51-68, December , 2015
Available online at http://www.tjef.net
ISSN: 2149-0325
http://dx.doi.org/10.17530/jef.15.15.2.2

Geospatial Information Retrieval Base on Query Expansion and Semantic Indexing

Omer Sevinc*
Computer Engineering Ondokuz Mayıs University Samsun, Turkey

Lucy Huang
Engineering Technical Dept.Texas A&M University Corpus Christi, USA

Mehrube Mehrubeoglu
Engineering Technical Dept.Texas A&M University Corpus Christi, USA

Li Loughzang
Engineering Technical Dept.Texas A&M University Corpus Christi, USA

Erdal Kilic
Computer Engineering Ondokuz Mayıs University Samsun, Turkey

Article history
Received:
18.08.2015

Received in revised form:
17.11.2015

Accepted:
24.11.2015

Key words:
component; geospatial information retrieval, query expansion, semantic indexing, semantic web

In this paper, a hybrid approach is developed for geospatial information retrieval which combines query expansion (QE) and latent semantic indexing (LSI) methods. The hybrid method uses the advantages of Query Expansion and Semantic indexing methods for improving search results. LSI establishes relations depending on the similarities between queries and the documents where QE helps by adding extra similar terms to the queries. The dataset is populated using data extracted from Wikipedia and USGS (The United States Geological Survey). Automation is programmed to get all the information from the web and to extract the meaningful words. The significant terms are the name of the places, directions, and nearby locations. The dataset includes mountains and places with their latitudes, longitudes, locations and directions. The extracted data is also taken into Protégé for semantically querying. The approach is applied to geographical data (popular search items on search engines), and by querying the data with the combination of LSI and QE more related results are handled than the results of methods when they are applied alone. Many queries are run, and the results are listed and compared. At the end of the study it is understood that the hybrid method considerably improves the search results.

*Correspondence: e-mail: osevinc@omu.edu.tr

INTRODUCTION

Searching geospatial information takes an important part of a total web search. A web search is actually based on information retrieval. Nowadays the practices of information retrieval generally depend on keyword-based searching through full-text data that model is called bag-of-words. That kind of sample exudes the actual semantic knowledge of the text. For dealing with the matter, ontology is recommended [1]. In last ten years, the investigate group has begun a struggle for investigating new infrastructures for the future stages of the Web, named as Semantic Web [2].

Query expansion is another method used to enhance user queries to get better results. Query expansion is required for the ambiguity of natural language, and also the difficulty of using just only word to typify an information notion [16].

Another method is Latent Semantic Indexing (LSI) have a try to solve the challenges of lexical mapping by under cover of statistically gained cognitive indices rather than unique words for retrieval. LSI supposes that there is some core meaning or latent structure in word usage which is partially cloaked by the versatility in term option [32].

In this paper, a new dealing is represented about geospatial searching which combines and compares Query Expansion and Semantic Indexing methods. This hybrid method uses both particular methods' advantageous features to improve search results. As an experiment, New Mexico's mountains data is used, extracted from USGS and Wikipedia. It is aimed to improve previous geospatial search methods which are mostly based on just ontology, query expansion or traditional keyword-based systems. Traditional keyword search methods just try to find out keywords of queries in the documents. The documents on the web are mostly in HTML format which does not allow keeping semantic data sufficiently. [35, 36] Query expansion adds synonyms to user queries to find out more related documents. This alone improves users' search results, but with LSI, it gives better results together because it is seen in our study that, after synonym terms are added to query matrices in LSI and formulas of LSI applied from the beginning, the related documents in the search results get better scores at the end. The semantic web systems are restricted by limited ontology. There should be a common ontology for specific domains. [34] Nowadays information is kept on the web mostly in HTML, XML PDF and office document formats but not in ontology. It is useful to use a huge amount data from common sources, which is more practical.

An XML file is created which describes mountains and places with their detailed information and enables us to index them, run queries with keywords, and also by using query expansion. Most occurring terms in the XML file are taken to create a document matrix to apply semantic indexing. Our hybrid method is applied by annexing new query terms to the query matrices to combine semantic indexing and query expansion. The results are improved, and the hybrid method gives better results than these methods applied singly.

RELATED WORKS

There are many related works done. They use methods like inverted index, vector space model, query expansion, spatial index, and semantic indexing [9, 12,14, 26]. Some of them combine these methods to improve search results. The latest works are concentrated on query expansion and semantic indexing. In our work, it is decided to combine these two methods to create a hybrid method. Query expansion adds new related terms to users' query and provides more search results. Semantic searching requires the data to be structured semantically to be queried by semantic query languages automatically or with user guidance which requires user interaction.

However, semantic indexing provides a user friendly keyword-based search interface. But, semantic indexing needs word disambiguation to clarify what the word means. So, we applied the LSI (Latent Semantic Indexing) on the data which is eliminated from unwanted stop words and suffixes by open source indexer Solr.

Inverted indexes are qualified as the vintage text indexing method. An inverted index reconciles to each term in the text (prearranged as vocabulary) an array of pointers to the situations where the term rises in the document. The collection of that entire list is named as the occurrences [6]. But the dilemma of these indexes is that geographic hierarchies are mostly neglected. This method does not provide the information of which cities are belong to which country and which counties are belong to which cities and so on. Solr indexer uses this method [31].

The classical keyword-based information retrieving dealings are essentially taking on the vector space sample recommended by "Salton" et al. [7]. In this approach, annotation or extraction phases are not required. It is simple to apply, but the precision is subdued. Besides, spatial index structures are used for spatial search. The one of the most known spatial index structures, and an analogous sample, is the R-Tree [8]. The R-Tree is an equable tree reproduced from the B-tree which divides space in hierarchically nested, likely converging, and the lowest bounding rectangles. A dilemma of spatial index structure is that they do not consider the hierarchy of space [9].

In some studies they combine inverted index and spatial index to specify the hierarchy between places to solve geographical reference problems. These are such as Web-a-where [10], Meta Carta [11], and STEWARD [12]. On the other hand, some approaches separate spatial and text indexing.

Ontology can truly define the unique characteristics of a geographic space that is a formal distinctive emphasize of severed conceptualization [13]. Using ontology you can define classes, relations between classes, attributes and instances of the classes. A class specifies the general properties of a thing. Attributes specify properties of things. Relations are predicates between things like belonging to something or being inside another thing.

Another novel approach is query expansion which improves the information retrieving for the queries, it is needed to prosper the actual user query and coat the gap between the user query and needed information. There are some works based on query expansion [14-15]. In this method, the interface can offer related information to the user for specifying search results. Lots of work has been achieved in the field of query expansion [15, 16, 23, 25]. Query expansion is required to overcome the difficulties of natural language, and also the challenges by use of just a word to stand for an information theme. Via query expansion, user is directed to generate queries which allow practical results to be acquired. The primary goal of query expansion is to annex new significant words to the users' very first query. This progression of annexing words can either be manually, automatically or with users' assisting. Hand built query expansions depend on user expertness, weightings are evaluated for all words, and the words which have the maximum weighting are put on to the very first query. Several weighting functions generate several results; by this reason, the retrieval productivity relies on how the weightings have been evaluated. With user guided query expansion, the system produces probable query expansion words, and the user chooses which of those to contain [16].

Another novel approach is semantic indexing. This method is also applied in our work. A paper which focuses on semantic indexing, claims that semantic indexing gives

better results than query expansion [5].

Usually, information is gained by literally mapping words in documents with those of a query. However, lexical mapping techniques can be indecisive when they are utilized to map a user's query. Thenceforward there are generally lots of manners to represent a given theme (synonymy), the literal words in a user's query could not map those of a related document. More, most words have more than one meaning (polysemy), so words in a user's query will literally match words in irrelevant documents. A better deal would enable users for retrieving information on the base of a cognitive subject or denotation of a document [33].

LSI solves two of the most challenging constraints of Boolean keyword queries: multiple words that have same meanings (synonym) and words that have multiple meaning (polysemy) [21].

Semantic indexing is a methodology in which data can be searched by a keyword based method. Where after organized XML files are created like documents and fields related to them, cleaned data is extracted from the indexer handled over the XML files, so extracted data can be indexed and searched. We use LSI for the semantic indexing method. All terms are handled from the indexer and put into the matrix to calculate the Eigen values. Then scores of documents are calculated.

So it is decided to combine the latest approaches and create a novel method to get better search results. Because of the benefit of the keyword based search and synonyms, the query expansion technique will be used, which brings more related results after querying data. Then, semantic indexing will be applied to the same data which are kept in an organized XML file but cleared by the Indexer. To create a hybrid method, first, LSI will be applied, and then, synonyms of some query terms will be added, and scores will be calculated again. It will be possible to search enriched data to get better results.

METHODS

It is/We concentrated mainly on two methods in our work which are Query Expansion, Semantic Indexing and a hybrid method which we proposed by combining these two methods. The framework can be seen in Figure 1.

Our method firstly queries data which are extracted from open and governmental sources over the internet to score the related documents for the first step. At this step, unwanted stop worlds and suffix are eliminated by configuring the indexer. For the purpose of query expansion, the next step is applied by adding synonyms and additional query terms which are consulted to and proved with experts. The additional query terms also have similar meanings to the query keywords which are written by the user. After synonyms and additional keywords are added the number of listed documents in the search results increase, and related documents which include synonym keywords get high scores, so the search results improve. For instance, if the documents have the synonyms keywords but not the initial query terms, they get a higher score than their previous scores. The indexer scores the documents according to the inverted index [31] checks for the adjoined keywords in the user’s query if they exist together in the documents; however, it checks for the keywords separately and gives scores which have more keywords. So, the results demonstrate that when the synonyms are added over the configuration file and similar keywords are added to the users’ queries, then more related documents are listed in the search results with higher scores, and also more documents are listed in total.

In the next step, Latent Semantic Indexing (LSI) is applied to the same data. LSI directly focuses on the keywords written in the users’ queries. If all the keywords are included in the documents without unrelated terms or at

least includes less unrelated terms get high scores.

When the LSI method is compared with the indexer search or QE search it is seen that LSI gives more accurate results because it gives the highest scores to the documents if solely searched keywords are included in them. It gives high scores to the documents which include the searched keywords with less unrelated terms inside them and so on.

At last step, synonym words are added to the query matrices and LSI is applied and scores of the documents recalculated.

The architecture can be seen in Figure-1 for better understanding. Wikipedia and USGS are used as data sources. Necessary information is dragged from the data sources. The information is also structured in XML format with certain tags to stress certain data. By that manner many related information packed into the related document with certain tags. For example, name, latitude, longitude, place (city, county), closer places and mountains are written. So, a document includes many detailed information about a mountain. The frequencies are calculated by the Indexer program to gain values of terms occurrence. On the other hand, the synonyms of the terms included in the documents are also added to the synonyms file to be evaluated when searches are done by queries. The indexer program ignored the stop words, made text tokenizing, lowercased, stemming. Over these enriched data, Latent Semantic Indexing, LSI is applied. The frequencies of the terms are represented as a matrix. Then, the matrix is reduced and similarity values are calculated, and then, ranks of the documents are listed as a result. LSI gives faster response to the queries. It is aimed to benefit from the advantages of query expansion and LSI methods by combining them.

Framework

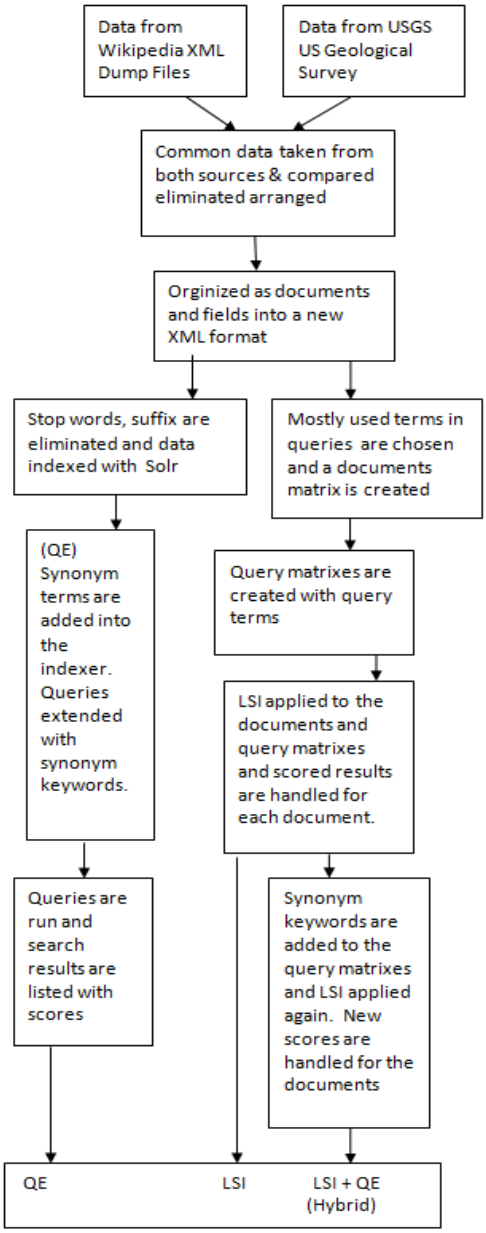

Figure 1. Architecture of the proposed approach

Query Expansion

To apply query expansion to our work, the synonym terms, which are specified by the experts, are added to the users' initial queries, and also added into the indexer's configuration file to take into account the synonyms. So the search results give better results which are more accurate, and the number of the documents returned

after the search increased. To adjust Solr indexer, field names in XML and the type of these fields are specified to be indexed or calculated. One of the important points is that detailed information like directions; locations of the mountains are described in specified tags in the XML file. A sample of the XML file can be seen in Figure 2.

```
<field name="is_in" type="text_general" indexed="true"
stored="true"/>                    <field name="lat" type="text_general"
indexed="true" stored="true"/>
<field name="lon" type="text_general" indexed="true" stored="true"/>
```

Figure 2. Solr XML input file sample.

It is possible to get extended information from new documents to apply one part of the Query Expansion task. It is possible to get mountains' information in certain specified areas by using latitude and longitude values. The latitude and longitude fields are numeric, so, for example, Solr can find out locations and mountains 10 miles far from the queried location. In Figure 3 some field definitions of Solr can be seen.

```
<doc>
 <field name="name"> Wheeler Peak
  </field>
<field name="is_in"> Taos County</field>
<field name="lat">36.0000</field>
<field name="lon">105.0000</field>
<field name="within">null</field>
<field name="direction">northwest</field>
<field name="dir_place">Taos, New Mexico</field>
<field name="direction2">north</field>
<field name="descript">lies to the northwest of Wheeler
Peak, while both the town of [[Taos, New Mexico
located northeast of [[Taos, New Mexico</field>
```

Figure 3. Sample of Solr Schema for indexing Wiki data by Solr

Using a queried mountain's latitude or longitude, certain other mountains' information can be handled if there are any close to the queried mountain. It is possible to teach Solr synonyms adding them to the synonym.txt file of it in the conf directory. However, it is also possible to add synonyms as new query terms to the user's queries in the Solr search bar. This provides query documents with synonym terms too. If a document does not have the query terms but synonyms, it will be listed in the search results after query expansion is applied. The documents are queried by adding synonyms to the user queries and to the Indexer. Certain

mappings for certain terms in Solr Indexer's synonym.txt file are done as below.

Peak => mountain, peak, mountains

Mountain => peak, mountain, mountains

Volcano => crater, volcano

Crater => volcano, crater

Documents' scores are handled over Solr with adding synonyms and without adding synonyms separately. For example, a query is run on Solr's search bar such as; "Sierra black peak" is extended to "Sierra black peak mountain", and also, "mountain = peak" similarity is determined in the synonym.txt before query is run. By adding score parameter to the query, the scores of the documents are handled over Solr Indexer, with and without adding synonyms. It is understood that when query expansion is applied by adding synonyms, precision value increases [23].

Semantic Indexing

To apply LSI, certain terms are specified to be queried, and the number of these terms in the documents is handled. In our work, 11 terms are picked which have high frequencies in whole terms of the documents which proved by Solr Indexer. The occurrences of the indexed data of the mountains and places are handled over the documents. The frequencies are also calculated automatically with Solr Indexer. The occurrences are used to create the document matrix. The document matrix is created with 11 X 41 dimensions. This means that a matrix consists of 11 terms and the occurrence of the terms in 41 documents. A sample of the terms documents matrix and question matrix can be seen in Figure 5. Singular Value Decomposition (SVD) is calculated over the matrix. Eigen values of the document matrix are handled and necessary rows and columns processed for the calculations. Then, more than 80 % of the data, which is presented with Eigen values and Eigen vectors, are kept, and the others are eliminated. The matrix is decomposed to find out the U, S and V matrices, where $A = USV^T$. Rank approximation is implemented by keeping the first six columns of U and V and the first six columns and rows of S. The number of columns is chosen six because this amount of the matrix carries more then %80 of the matrix value. As a result, the matrix is induced.

The new document vector coordinates are handled in the reduced six dimensional spaces. Rows of V carry eigenvector values. These are the coordinates of unique document vectors like:

d1 (-0.2487, 0.8746, 0.5477, …)

d2 (-0.3247, -0.1134, 0.4777, …) etc.

The new query vector coordinates are found in the reduced 6-dimensional space. The applied formula is $q = q^T U_k S_k^{-1}$.

These are the new coordinates of the query vector in six dimensions. The matrices are now different from the original query matrices (q1, q2, …) created at the beginning. The documents are ranked in a decreasing order of query-document cosine similarities. The cosine similarities are calculated with the formula: $sim(q,d) = (q.d)/(|q| . |d|)$.

QE + LSI

A method is proposed which combines query expansion and semantic indexing. The Query expansion method adds new terms (synonyms) to the user's queries to improve search results, where LSI tries to find out related documents by using statistical derived conceptual indices. So we decided to combine both methods to get better results. The synonym terms related with the user queries are added to the query matrix in LSI to recalculate the SVD and the document scores. By adding terms to the query matrices the query expansion method is applied, and then, by recalculating all LSI values again, the LSI method is applied to get

results with the hybrid method. The occurrences are used to create the document matrix. Singular Value Decomposition (SVD) is calculated over the matrix. Eigen values of the document matrix are handled and necessary rows and columns are considered. Then, more than 80 % of the data which is presented with Eigen values and Eigen vectors are kept and the others are eliminated. As a result, the matrix is induced. To achieve this, a matrix is created that has 41 columns for each document, and 11 rows include sample terms which are included in the documents with different frequency values. A sample of the terms documents matrix and question matrix can be seen in Figure 5. The documents include information of mountains and places which are named as D1, D2, and D3 etc. The sample question is "Sierra Black Peak" which is named as Q1. A question matrix (11X1) is also defined. The V matrix is reduced to 6 rows. Then the 6X41 matrices are handled. The cosine similarities of document matrices are also calculated, and the documents are ranked [30]. The ranking results of the documents can be seen in Figure 6. The semantic indexing method makes the searching process faster [22].

To combine query expansion and LSI, the synonyms are added to the query matrices which are created at the beginning, and then all the LSI calculation process is renewed to get new result by the hybrid method. For example, the beginning query matrix is on the left in Table 1, and on the right the mountain term is added to the query.

Q1	**Q1 (After synoym added)**
1	1
0	1
0	0
1	1
1	1
0	0
0	0
0	0
0	0
0	0
0	0

Table 1 . The matrix of some of the documents with chosen terms

Implementations

A common data source is used for obtaining data on the web. A governmental, reliable data source is also used to verify and to get extra information which is not included in the common data source. The most common data source Wikipedia is used for obtaining mountain's name, coordinates, locations such as city or county. USGS is used to verify the data handled from Wiki and to get extra information about mountains and places of New Mexico. USGS directly and clearly contains geographical information of the United States of America. It is a text file and stores places, mountains, and coordinates. After the data is handled, they are eliminated, reorganized by automation, which is programmed by us and imported into an open source indexer program. Our program reads that file and automatically extracts data which is needed, and stores them into a XML file. After related XML dump files of Wikipedia are downloaded, the necessary information is extracted from these files and compared with the data of USGS, then stored into a new XML file with new tags to be imported into Solr. Our java program automatically extracts and stores data for importing XML files into Solr, the schema file of Solr is also predefined by describing the data types of tags, and rules for eliminating some unnecessary terms and to make Solr able to understand tags of the imported XML file. To apply one aspect of Query Expansion task extra information is added. This information extends the documents with new terms. Other aspects of the Query Expansion method are applied by eliminating stop words, and word stemming and adding synonyms. These processes are carried out by the indexer after

it is configured up to the needs by us. The program coded by us automatically extracts and stores data into a XML file. In the XML file, specified tag names are created for each property of the documents. The indexer indexes the documents and provides us a query screen for querying documents. Firstly, initial queries are run and scores of the resulted documents are listed. Then, synonyms are added to the Solr's synonym.txt file and to the initial user queries. After that, documents' scores are recalculated. To apply LSI, the indexed terms are handled from Solr indexer and some of certain terms are picked from whole indexed terms. Then, a document and query matrices are created to be used for applying singular value decomposition to calculate Eigen values and similarities for handling documents' scores according to the LSI. At last step, synonyms are added to the query matrices, and the same processes for LSI are executed again. Then, all results are compared.

EXPERIMENTS

First, the prepared documents are indexed with Solr indexer. The queries are run on the Solr, and the results are handled. The results can be seen in Figure 7, under Keyword Based title where just stop words, unwanted suffixes are eliminated, and the special information such as coordinates of the points or name of the points are categorized with specified tags. The queries are directly written in the search bar of Solr. Search results are listed with the scores of the documents. The results demonstrate weather the documents include the query terms with high frequency, and then get higher scores. In this method it is important how many times the query terms occur and what the frequencies of the terms are. If the frequency and number of occurrence of the term is high, then the container documents get higher scores.

Second, to apply query expansion to the current documents and queries the synonym terms are added both to the user queries as additional keywords and to the Solr indexer's synoym.txt file which makes Solr understand synonyms words. A part of Query expansion task is already completed with enriched data with the categorizing data of entities with specified tags [25]. The other main part is applied to fulfill query expansion by adding synonyms to the indexer and user queries. The synonym terms are specified with experts. To expand the query terms, the synonyms are added into them. Then, queries are run in the search bar of Solr. Results demonstrate that, when synonyms are added, more documents are listed in the search results. If synonyms are not included in the initial query, the documents are listed that only including the query's terms. When synonyms are added, the documents which include synonyms too are also listed in the search results. If a document has a high number of synonyms with initial query terms it gets a high score.

Third, LSI is applied with chosen terms from the same documents used in previous methods. The semantic indexing method makes searching process faster [22]. To achieve this, a matrix is created that has 41 columns for each document and 11 rows include sample terms which are included in the documents with different occurrences values. A sample of the terms documents matrix and question matrix can be seen in Figure 5. The documents include information of mountains and places which are named as D1, D2, and D3 etc. The sample question is "Sierra Black Peak" which is named as Q1. A query matrix (11X1) is also defined. Query matrices can be seen in Figure 6. Singular Value Decomposition (SVD) is calculated over the matrix. Eigen values of the document matrix are handled and necessary rows and columns considered. Then, more than 80 % of the data which is presented with Eigen values and Eigen vectors are kept, and the others are eliminated. The matrix is decomposed to find out the U, S and V matrices. The V matrix is reduced to 6 rows.

Then the 6X41 matrices are handled. The cosine similarities of document matrices are calculated, and the documents are ranked [30]. The ranking results of the documents can be seen in Figure 7 under LSI title. LSI has a different technique than both previous methods because LSI tries to get the concept of the documents by using statistically derived cognitive indices instead of scoring the documents with the number of occurrence of individual query terms. The scores of the documents are calculated with the LSI method. It is seen that if a document solely includes just one of the query terms and another document includes solely the same term but more than one time, they get the same scores. But, if a document includes solely some of or the entire query terms, it gets a higher score than a document with a query term repeated many times inside it. If there are some unrelated terms too, which are not related with the query terms inside the document, the document's score decrease.

Forth, the query expansion method is combined with the LSI method. To achieve that, initial query matrices are expanded by adding synonyms. The LSI calculations are done from the beginning with new values. Similarities are recalculated; new results for the documents are handled. The results demonstrate that, when synonyms are added to the query matrix, the documents which have synonym terms get higher scores than previous. If a document does not have synonyms it may get similar or fewer scores than previous. When this hybrid method is evaluated on our dataset, which includes 41 documents with 11 terms for each, it is seen that 74 % of the results improve and get better scores than the results of LSI. A control can also be added to evaluate the results where documents do not have synonyms and get fewer scores than previous. The previous results can be kept for these kinds of documents instead of changing with the new scores. As a result, it is seen that the hybrid method considerably improves results and related documents, which include synonyms too, get higher scores.

Datasets

The hybrid and LSI methods are applied with more than ten queries and a hundred documents. More than a thousand document scores are handled. But for explanation identical documents are chose which include most frequently occurred terms.

The terms and documents matrix is created. When the matrix is created all documents are used with chosen terms. Some of the documents which contain chosen terms are listed in the matrix in Table 2 below.

Terms /Docs	**D 1**	**D 2**	**D 3**	**D 4**	**D 5**	**D 6**	**D 7**	**D 8**	**D 9**	**D 11**	**D 13**	**D 14**
Peak	2	2	0	0	1	0	2	1	0	2	0	0
Mountain	1	0	0	1	1	1	2	3	1	1	1	0
Baldy	0	0	0	1	0	0	0	0	1	0	0	1
Black	0	0	1	0	0	0	0	0	0	0	0	0
Sierra	0	0	0	0	0	0	0	1	0	0	1	0
Crater	0	0	0	0	0	0	0	0	0	0	0	0
Volcano	0	0	0	0	0	0	0	0	0	0	0	0
Grande	0	0	0	3	0	0	0	0	0	0	0	0
Blue	0	0	0	0	0	0	0	0	0	0	0	0
Cathey	0	0	0	0	0	0	0	0	0	0	0	0
Cerro	0	0	0	0	0	0	0	0	0	0	0	0

Terms /Docs	D 15	D 16	D 20	D 24	D 26	D 28	D 32	D 33	D 38	D 40	D 41	D 42
Peak	0	1	0	0	0	0	0	0	3	0	0	0
Mountain	1	0	1	1	1	0	0	0	1	0	1	0
Baldy	0	0	0	0	0	0	0	0	0	1	0	0
Black	0	0	1	0	0	0	0	0	0	0	0	0
Sierra	0	1	0	0	0	1	0	0	0	0	0	0
Crater	0	0	0	0	0	0	2	0	0	0	0	0
Volcano	0	0	0	0	2	0	0	1	0	0	0	0
Grande	0	0	0	0	0	0	0	0	0	0	0	1
Blue	1	0	0	0	0	0	0	0	0	0	0	0
Cathey	0	0	0	0	0	0	0	0	2	0	0	0
Cerro	0	0	0	0	0	0	0	0	0	0	0	1

Table 2. The matrix of some of the documents with chosen terms

Queries

Queries are specified to apply the LSI method. It is considered that some of the query terms should have synonyms because after LSI is applied synonyms will be added to the query matrices, and LSI formulas will be applied from the beginning to practice our hybrid method. Some queries are created according to our chosen terms. The query matrices can be seen in Table 3.

Terms /Queries	Q1	Q2	Q3	Q4	Q5
Peak	1	0	0	1	0
Mountain	0	1	0	0	0
Baldy	0	1	0	1	0
Black	1	0	0	0	0
Sierra	1	0	0	0	0
Crater	0	0	0	0	0
Volcano	0	0	1	0	0
Grande	0	1	0	1	0
Blue	0	0	1	0	0
Cathey	0	0	0	0	1
Cerro	0	0	0	0	1

Table 3. Sample frequencies of the terms, queries and the documents matrix

Query1 : Sierra Black Peak
Query2: Grande Baldy Mountain
Query3: Blue Crater
Query4: Grande Baldy Peak
Query5: Grande Baldy Sierra Mountain
Query6: Blue Baldy Black Crater
Query7: Grande Blue Baldy Peak
Query8: Cerro Peak Cathey
Query9 : Sierra Black Mountain Peak
Query10: Grande Baldy Cerro Mountain
Query11: Blue Crater Volcano Mountain Cathey
Query12: Crtater Grande Baldy Peak
Query13: Volcano Cerro Cathey Blue

Results

The results of all the methods applied at each step can be seen below. The first three queries have terms that have synonyms. For these queries all methods are applied and results are listed in Table 4. The fourth query includes the peak term, so it is not considered in the results part because the same synonym is considered in the first and second queries. Fifth query does not include a term that has a synonym, so it is also not included in the search results.

LSI (Latent Semantic Indexing)							
Query 1							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,5680	0,5685	0,1213	0,0554	0,4246	-0,1082	0,4246	0,340
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
-0,2954	0,5685	0,5680	NaN	0,4470	-0,2662	-0,105	0,994
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
-0,2954	0,5685	0,5685	-0,0822	-0,0822	-4,e-15	-0,108	-0,10
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
0,5685	-0,7115	-0,266	0,7578	0,5685	0	-0,701	0,500
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
0,5685	0,48371	0,5680	-0,2662	-0,1082	0,1712		
Query 2							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,02031	-0,1096	-0,145	0,7928	0,1359	0,2489	0,1359	0,187
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
0,6923	-0,1096	0,0203	NaN	0,1148	0,6262	0,2323	-0,12
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
0,69234	-0,1096	-0,109	0,20587	0,2058	-1,e-16	0,2489	0,248
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
-0,1096	0,0579	0,6262	-0,0632	-0,1096	0	0,0049	0,340
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
-0,1096	-0,02080	0,0203	0,62629	0,24895	0,4512		
Query 3							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,5680	0,5685	0,1213	0,0554	0,4246	-0,1082	0,4246	0,340
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
-0,2954	0,5685	0,5685	-0,0822	-0,0822	-4,e-15	-0,108	-0,10
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
0,5685	-0,7115	-0,266	0,7578	0,5685	0	-0,701	0,500
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
0,5685	0,48371	0,5680	-0,2662	-0,1082	0,1712		
Query 5	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,02031	-0,1096	-0,145	0,7928	0,1359	0,2489	0,1359	0,187
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
0,69234	-0,1096	-0,109	0,20587	0,2058	-1,e-16	0,2489	0,248
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
-0,1096	0,0579	0,6262	-0,0632	-0,1096	0	0,0049	0,340
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
-0,1096	-0,02080	0,0203	0,62629	0,24895	0,4512		
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
0,3063	0,0256	0,6183	-0,1023	0,3063	0	0,0133	0,325
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
0,3063	0,3730	0,3727	0,6183	0,0587	0,4577		
Query7							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,5680	0,5685	0,1213	0,0554	0,4246	-0,1082	0,4246	0,340
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
-0,2954	0,5685	0,5680	NaN	0,4470	-0,2662	-0,105	0,994
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
-0,2954	0,5685	0,5685	-0,0822	-0,0822	-4,e-15	-0,108	-0,10
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
0,5685	-0,7115	-0,266	0,7578	0,5685	0	-0,701	0,500
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
0,5685	0,48371	0,5680	-0,2662	-0,1082	0,1712		
Query 9							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,02031	-0,1096	-0,145	0,7928	0,1359	0,2489	0,1359	0,187
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
0,6923	-0,1096	0,0203	NaN	0,1148	0,6262	0,2323	-0,12
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
0,69234	-0,1096	-0,109	0,20587	0,2058	-1,e-16	0,2489	0,248
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
-0,1096	0,0579	0,6262	-0,0632	-0,1096	0	0,0049	0,340
D37	**D38**	**D39**	**D40**	**D41**	**D42**		

-0,1096	-0,02080	0,0203	0,62629	0,24895	0,4512		
Query 11							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,5680	0,5685	0,1213	0,0554	0,4246	-0,1082	0,4246	0,340
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
-0,2954	0,5685	0,5680	NaN	0,4470	-0,2662	-0,105	0,994
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
-0,2954	0,5685	0,5685	-0,0822	-0,0822	-4,e-15	-0,108	-0,10
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
0,5685	-0,7115	-0,266	0,7578	0,5685	0	-0,701	0,500
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
0,5685	0,48371	0,5680	-0,2662	-0,1082	0,1712		
Query 13							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,02031	-0,1096	-0,145	0,7928	0,1359	0,2489	0,1359	0,187
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
0,6923	-0,1096	0,0203	NaN	0,1148	0,6262	0,2323	-0,12
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
0,69234	-0,1096	-0,109	0,20587	0,2058	-1,e-16	0,2489	0,248
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
-0,1096	0,0579	0,6262	-0,0632	-0,1096	0	0,0049	0,340
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
-0,1096	-0,02080	0,0203	0,62629	0,24895	0,4512		
Query 15							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,5680	0,5685	0,1213	0,0554	0,4246	-0,1082	0,4246	0,340
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
-0,2954	0,5685	0,5680	NaN	0,4470	-0,2662	-0,105	0,994
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
-0,2954	0,5685	0,5685	-0,0822	-0,0822	-4,e-15	-0,108	-0,10
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
0,5685	-0,7115	-0,266	0,7578	0,5685	0	-0,701	0,500
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
0,5685	0,48371	0,5680	-0,2662	-0,1082	0,1712		
Query 1							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,02031	-0,1096	-0,145	0,7928	0,1359	0,2489	0,1359	0,187
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
0,6923	-0,1096	0,0203	NaN	0,1148	0,6262	0,2323	-0,12
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
0,69234	-0,1096	-0,109	0,20587	0,2058	-1,e-16	0,2489	0,248
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
-0,1096	0,0579	0,6262	-0,0632	-0,1096	0	0,0049	0,340
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
-0,1096	-0,02080	0,0203	0,62629	0,24895	0,4512		
Query 1							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,5680	0,5685	0,1213	0,0554	0,4246	-0,1082	0,4246	0,340
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
-0,2954	0,5685	0,5680	NaN	0,4470	-0,2662	-0,105	0,994
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
-0,2954	0,5685	0,5685	-0,0822	-0,0822	-4,e-15	-0,108	-0,10
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
0,5685	-0,7115	-0,266	0,7578	0,5685	0	-0,701	0,500
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
0,5685	0,48371	0,5680	-0,2662	-0,1082	0,1712		
Query 2							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,02031	-0,1096	-0,145	0,7928	0,1359	0,2489	0,1359	0,187
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
0,6923	-0,1096	0,0203	NaN	0,1148	0,6262	0,2323	-0,12
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
0,69234	-0,1096	-0,109	0,20587	0,2058	-1,e-16	0,2489	0,248
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
-0,1096	0,0579	0,6262	-0,0632	-0,1096	0	0,0049	0,340
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
-0,1096	-0,02080	0,0203	0,62629	0,24895	0,4512		
Query 1							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,5680	0,5685	0,1213	0,0554	0,4246	-0,1082	0,4246	0,340
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
-0,2954	0,5685	0,5680	NaN	0,4470	-0,2662	-0,105	0,994
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**

-0,2954	0,5685	0,5685	-0,0822	-0,0822	-4,e-15	-0,108	-0,10
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
0,5685	-0,7115	-0,266	0,7578	0,5685	0	-0,701	0,500
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
0,5685	0,48371	0,5680	-0,2662	-0,1082	0,1712		
Query 2							
D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**
0,02031	-0,1096	-0,145	0,7928	0,1359	0,2489	0,1359	0,187
D9	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
0,6923	-0,1096	0,0203	NaN	0,1148	0,6262	0,2323	-0,12
D17	**D18**	**D19**	**D20**	**D21**	**D22**	**D23**	**D24**
0,69234	-0,1096	-0,109	0,20587	0,2058	-1,e-16	0,2489	0,248
D25	**D26**	**D27**	**D28**	**D29**	**D32**	**D33**	**D34**
-0,1096	0,0579	0,6262	-0,0632	-0,1096	0	0,0049	0,340
D37	**D38**	**D39**	**D40**	**D41**	**D42**		
-0,1096	-0,02080	0,0203	0,62629	0,24895	0,4512		

Table 4. Scores of the documents handled with Keyword Based, QE, LSI, LSI + QE methods.

Technique	Solr Indexer		LSI	
Documents	**QE**		**LSI**	**LSI +QE**
Query	**Query1**		**Query1**	**Query1**
D13	0.3077696		0,4470	0,8404
D16	0.1415263		0,9949	0,7353
D8	0.5000139		0,3406	0,8465
D20	0.4195109		-0,0822	0,5463
Query	**Query2**		**Query2**	**Query2**
D4	0.4139775		0,7928	0,7557
D9	0.17834		0,6923	0,5936
D17	0.2853565		0,6923	0,5936
D14	0.06054		0,6262	0,6183
Query	**Query3**		**Query3**	**Query3**
D22	0.26327816		0.0000	0.7364
D26	0.23270719		0.9810	0.6637
D33	0.19745861		0.9997	0.6764

Table 5. A sample comparison of some of the document scores.

In Table 5 some of the chosen documents' scores are listed under the applied techniques. For Query 1, when the results of the keyword based, which is a querying method without synonyms, is compared with the results of the query expansion, it is seen that; D13, D20, D8 get higher scores than previous where D16 get fewer. When we check the terms inside the documents D13, D20, D8 all include the term "mountain", which is added to the user query as a synonym to apply query expansion. D13and D20 does not include peak but mountain. When synonym is added these documents get higher scores. D8 includes once the peak term with thrice mountain terms, it also boosts after synonym is added to the query. D16 includes the terms "peak" and "sierra" but not mountain, so adding synonym does not affect D16 positively. At the beginning, D13 has a higher score than D16, even though D16 includes sierra and peak terms where D13 includes sierra and mountain. The reason is that, with whole terms, D16 has more unrelated terms than D13. Occurrence of D16 is better than D13 but because of the higher number of unrelated terms inside D16, it gets a fewer score. D8 boosts when the synonym "mountain" is added, because it includes thrice that term with less unrelated terms inside it. For Query 2; D4 gets the highest score because it has thrice "Grande", once "mountain" and "baldy" terms. It does not increase when query expansion is applied, because the synonym term is not included in D4. Same things happen for D9, D17, and D14 too.

There is a difference with the first two methods and LSI. In the first two methods all terms in the documents are taken into account, but in LSI, eleven terms are chosen which have high frequencies to create the

document matrix. Whole terms are not used when the document matrix is created, because it is not feasible. However, the effect of our hybrid method can be clearly seen from the chosen terms, too.

When LSI is applied with Query 1 to D13, D16, D8, D20, D16 gets the highest score. D16 solely has "peak" and "sierra" terms which are written in the initial Query. That is why D16 gets higher scores at the beginning. Then, a synonym is added to Query 1. "Mountain" term is added to the query matrix, and then scores of the documents are recalculated. When our hybrid method is applied, D8, D13, D20 increase dramatically because all these documents include the mountain term, too. D16 decreases after this process is run because it does not include the mountain term.

When Query 2 is run D4 gets the highest score because it includes the "Grande", "baldy", "mountain" terms but not the "peak" term, which is added later as a synonym to apply the hybrid method. When the hybrid method is applied the score decreases. Same rule works for D9, D17, D15, too. If the added synonym is included with the document, the score increases after the hybrid method is applied. Otherwise scores decrease or stay almost still.

When Query 3 is run similar results are handled as in Query 1 and Query2. D22 does not include the query terms but the synonym "crater", because of that, when the hybrid method is applied the score of D22 increases.

After synonyms are added to the query matrices and the hybrid method is applied, more related documents are listed in the search results. It is evaluated for 41 documents and three querying results to get precision, recall and f-measure values after LSI and our hybrid method is applied. In Table 6 it can be seen that when our hybrid method is applied, all recall, precision and F-measures get better values than the values of the LSI method.

Queries	Precision	Recall	F-Measure
LSI			
Query1	1	0,714	0,833
Query2	0,96	0,774	0,857
Query3	1	0,5	0,666
Query4	1	0,714	0,833
Query5	0,96	0,774	0,857
Query6	1	0,5	0,666
Query7	1	0,714	0,833
Query8	0,96	0,774	0,857
Query9	1	0,5	0,666
Query10	1	0,714	0,833
Query11	0,96	0,774	0,857
Query12	1	0,5	0,666
LSI + QE (Hybrid)			
Query1	1	1	1
Query2	0,967742	1	0,984
Query3	1	1	1
Query1	1	0,714	0,833
Query2	0,96	0,774	0,857
Query3	1	0,5	0,666
Query1	1	0,714	0,833
Query2	0,96	0,774	0,857
Query3	1	0,5	0,666
Query1	1	0,714	0,833
Query2	0,96	0,774	0,857
Query3	1	0,5	0,666

Table 6. Precision, Recall and F-Measure values of LSI and Hybrid Method

As a result, it is clear that when query expansion is applied to a basic query or semantic indexing method, it improves search results. When queries are run without adding synonyms the results include documents that just include the query terms. When synonym is added more related documents are listed in the search results. LSI gives more accurate results because it

focuses on the occurrences of the different query terms at least ones, instead of more occurrences of just one or two terms of the query. Repeating the same terms many times does not increase the documents' score in LSI. When our hybrid method is applied documents which include synonyms get higher scores than previous. More related documents are listed with high scores in the search results.

CONCLUSION AND FUTURE WORK

In this work an approach is proposed by combining query expansion and semantic indexing. With this hybrid method it is aimed to achieve to put on new significant terms to the initial query of users, and enable users to search by keyword, which is simple, and users have already been using this method to search for a long time. The automatically extraction of the data from Wikipedia dump files and USGS text file is achieved, and the data stored into a new XML file which can be indexed by Solr. To have detailed information such as latitude, longitude, certain relations like near, south, north of certain locations, and property data of mountains and places such as name, city, and county are described with specified tag names in the XML file. The method is applied to the mountain and places data, but other types of data such as healthcare or academic information can also benefit from this method. Synonym terms, which are decided upon with experts, are added into the indexer, and also to the user queries to apply query expansion. The queries are run after synonyms are added when the search results considerably improved. Latitude, longitude values are used to find out if there is any close mountain or place in the context of query expansion.

By using most occurring terms a document matrix is created to apply LSI. LSI gets the concept of the documents by use of statistically derived conceptual indices in replacement of ranking the documents with the number of processions of individual query words.

At the end, to apply our hybrid method; LSI is combined with query expansion method by adding synonyms to the query matrices, recalculating LSI and getting document scores again. So, by that manner it is achieved to combine two methods to get better search results. It is obvious that the hybrid method dramatically enhances search results.

On the other hand, Query expansion is a very enhancer method over keyword based search. It improves search results well. It improves user queries and provides handling of more related search results. By adding synonyms to user queries more related documents can be found out.

LSI tries to get the conceptual meaning of the document instead of indexing single query terms. LSI gives accurate results where related documents are listed in search results with high scores.

But if semantic indexing is also applied together with query expansion by adding extended terms, which are synonyms to query matrices in LSI, it gives better results and representation of related documents which are ranked quickly.

As a conclusion, nowadays, most of the documents are not semantically described on the web, so the documents should be enriched with alternative terms, synonyms and extra related information by using data from current web documents, making inferences of new information from handled data over the web, which means query expansion is practically more usable. LSI provides us to get related documents up to the conceptual meaning over the document matrix. After synonyms are added and query expansion is applied to LSI, our hybrid method provides better results because it uses the advantages of both methods.

As a future work, more data will be used in another field to handle considerable results. Synonyms will be specified by using

weighting functions and will be added to the document matrix to apply LSI, automatically.

Acknowledgment

This research is funded with the YOK (The Higher Education Council of Turkey), Computer Science Program of TAMUCC (Corpus Christi Texas A&M University), and Ondokuz Mayıs University Samsun.

References

[1] T.R. Gruber, Toward principles for the design o ontologies used for knowledge sharing, International Journal of Human Computer Studies 43, pp. 907-928, 1995.

[2] J. Berners-Lee, J. Hendler and O. Lassila. The Semantic Web, Scientific American, vol 184, no. 5, pp. 34-43, 2001.

[3] J. Egenhofer. Toward the Semantic Geospatial Web. ACM-GIS 2002. 10th ACM International Symposium on Advances in Geographic Information Systems. McLean (USA). 2002.

[4] V. Uren, Y. Lei, V. Lopez, H. Liu, E. Motta, M. Giordanino, The usability of semantic search tools: areview, Knowledge Engineering pp. 361-377, Review 22, 2007.

[5] K. Soner, A. Ozgur, S. Orkunt, A. Samet, C. K. Nihan, A. N. Ferda, "An ontology-based retrieval system using semantic indexing," Elsevier, Information Systems pp. 294-305, 37, 2012.

[6] Baeza-Yates R, Ribeiro-Neto B, Modern information retrieval. Addison-Wesley, Harlow, 1999.

[7] G. Salton, A. Wong, C.S. Yang, A vector space model for automatic indexing, Communications of ACM 18, pp. 613-620, 1975.

[8] Guttman A (1984) R-Trees: a dynamic index structure for spatial searching. In: Yormark B (ed) SIGMOD'84, proceedings of annual meeting, Boston, Massachusetts, pp. 47-57, June 18-21, 1984. ACM, New York.

[9] B. R. Nieves, L. R. Miguel, P. S. Angeles, S. Diego, Exploiting geographic references of documents in geographical information retrieval system using an ontology-based index, Springer, pp. 309-310, 2010

[10] Amitay E, Har'El N, Sivan R, Soffer A (2004) Web-q-where: geotagging web content, pp. 273-280, In: SIGIR '04: proceedings of the 27th ACM, New York, http://doi.acm.org/10.1145/1008992.1009040

[11] Rauch E, Bukatin M, Baker K, A confidence-based framework for disambiguating geographic terms. In: Proceedings of the HLT-NAACL 2003 workshop pn analysis of geographic refremces. Association for Computational Linguistics, Morristown, USA, pp 50-54,2003.

[12] Lieberman MD, Samet H, Sankaranarayanan J, Sperling J, STEWARD: architecture of a spatio-textual search engine. In: Proceedings of the 15th ACM Int. Symp. on Advances in GIS (ACMGIS'07). ACM New York, pp. 186-193, 2007.

[13] Gruber TR, A translation approach to portable ontology specifications. Knowl Acquis5(2), pp. 199-200.

[14] A. Neda, P. Pallabi, M. Sheetal, K. Latifur, S. B. Steven, T. Bhavani, Ontology-driven query expansion methods to facilitate federated queries, IEEE, 2010.

C. G. V. Guillermo, A. Lylia, C. Nadine, A query expansion method applied to water information system, 5th International Conference on Signal Image Technology and Internet Based Systems, 2009.

[15] J. Bhogal, A Macfarlane, P Smith, A review of ontology based query expansion, Information Processing Management, Science Direct, 2007.

http://en.wikipedia.org/wiki/Apache_Solr, 04.27.2012, 10:54 am.

[16] http://semanticweb.org/wiki/FaCT, 19.05.2013, 12:20 pm.

[17] http://infomesh.net/2002/notation3/, 19.05.2013,12:40pm

[18] http://protegewiki.stanford.edu/wiki/DL_Query, 28.05.2013

[19] http://en.wikipedia.org/wiki/Latent_semantic_indexing, 28.05.2013

[20] http://www.semanticsearchart.com/researchLSA.html, 28.05.2013

[21] F.B. Dian Paskalis , M.L. Khodra, Word Sense Disambiguation In Information Retrieval Using Query Expansion, 2011 International Conference on Electrical Engineering and Informatics 17-19 July, Bandung, Indonesia, 2011.

[22] Tannebaum W.,Rabuer A., Acquiring lexical knowledge from Query Logs for Query Expansion in Patent Searching , IEEE Sixth International

[23] Conference on Semantic Computing,2012.

[24] Ghali B. Quadi A. and friends,Probabilistic Query Expansion Method Using Recommended Past User Queries,2012

[25] Thorleuchter Dirk.,Poel Dirk, Technology classification with latent semantic indexing, Expert Systems with Applications ,2013

[26] Gottlob Georg and friends, Ontological Queries: Rewriting and Optimization,ICDE Conference, 2011

[27] Sevinc Omer, Kilic Erdal, To Retrive Geospatial Information and Query with Query Expansion Method, Inet-tr Conference, Eskisehir, Turkey, 2012

[28] David A. Grossman, Information Retrieval, Algorithms and Heuristics, Latent Semantic Indexing, Springer, pp 70-74, 2004

[29] http://www.solrtutorial.com/basic-solr-concepts.html, 3.10.2013

[30] Deerwester Scoot, Dumais Susan T., Harshman Richard, Indexing by Latent Semantic Analysis 1999

[31] Latent Semantic Indexing: An overview, Rosario Barbara, 2000.

[32] David Vallet, Enrico Motta, and friends, Semantic Search Meets the Web

[33] http://james.padolsey.com/general/semantic-html-is-dying/,26.11.2013

[34] http://rhizomik.net/html/~roberto/thesis/html/SemanticWeb.html,27.11.2013