ISSN No. 0976-5697

Volume 2, No. 6, Nov-Dec 2011

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Clustered approach to Web Search Using SVM as Load Balancing Module

Miss. Rita Shelke*
Computer Department College of engineering
Bharati Vidyapeeth deemed university
Pune, India.
ritashelke@gmail.com

Prof. Devendrasingh Thakore
Computer Department College of engineering
Bharati Vidyapeeth deemed university
Pune, India
deventhakore@yahoo.com

Abstract: The rapid growth of the Internet has made the Web a popular place for collecting information. Today, Internet user access billions of web pages online using search engines. Information in the Web comes from many sources, including websites of companies, organizations, communications etc. Effective representation of Web search results remains an open problem in the Information Retrieval community. To overcome this, the relevant Web pages are often located close to each other in the Web graph of hyperlinks. It presents a graphical approach for entity resolution & complements the traditional methodology with the analysis of the entity-relationship (ER) graph constructed for the dataset being analyzed. It can significantly improve the quality of entity resolution. Using Support Vector Machines (SVMs) as supervised learning methods distributes the workload over the network by assigning the capacity to handle the number of requests at a time. Hence provide the stable system with quality results.

Keywords: Information Retrieval; Web; ER; SVM; Cluster;

I. INTRODUCTION

Searching for entities is a common activity in Internet search today. Searching for web pages related to a person accounts for more than 5 percent of the current Web searches. Currently, it is done using keywords. A search engine such as Google or Yahoo! returns a set of web pages, in ranked order, where each web page is deemed relevant to the search keyword entered (e.g. the person name in this case). [1] A search for a person such as say "*Andrew McCallum*" will return pages relevant to any person with the name *Andrew McCallum*. A next generation search engine can provide significantly more powerful models for person search.

The clusters can be returned in a ranked order determined by aggregating the rank of the web pages that constitute the cluster. With each cluster, we also provide a summary description that is representative of the real person associated with that cluster (for instance, in this example, the summary description may be a list of words such as "computer science, machine learning, and professor"). The user can work on the cluster of interest to him/her and get all pages in that cluster, i.e., only the pages associated with that *Andrew McCallum*.

Effective representation of Web search results remains an open problem in the Information Retrieval community. For ambiguous queries, a traditional approach is to organize search results into groups (clusters), one for each meaning of the query. These groups are usually constructed according to the topical similarity of the retrieved documents, but it is possible for documents to be totally dissimilar and still correspond to the same meaning of the query. The clusters can be returned in a ranked order determined by aggregating the rank of the web pages that constitute the cluster. Such cluster-based people search could potentially be very useful. If the web pages were randomly assigned to clusters, the cluster-based approach could be worse compared to the state of the art.

II. QUERY PROCESSING SYSTEM

Internet search engines have become an indispensable tool for people looking for information on the web. The majority of publicly available search engines adopt the so-called *query-list paradigm*, whereby in response to a user's query the search engine returns a linear list of short document summaries (*snippets*).Despite its great popularity; the query-list approach has several deficiencies. Moreover, especially in case of ill-defined queries, small groups of interesting but low-ranked outlier documents may remain unnoticed by most users. One alternative to ranked lists is *search results clustering*.

In this setting, in response to a query "London", for example, the user would be presented with search results divided into such topics as "London Hotels", "Weather Forecasts", "Olympic Games" or "London Ontario Canada". Users looking for information on a particular subject would be able to identify the documents of interest much quicker, while those who need a general overview of all related topics would get a concise summary of each of them.

[2] Search results clustering involve a class of algorithms called post-retrieval document clustering algorithms. A successful search results clustering algorithm must first of all identify the major and outlier topics dealt with in the results based only on the short document *snippets* returned by the search engine (most users are unwilling to wait for the full documents to download). Secondly, in order to help the users to identify the results of interest more quickly, the algorithm must label the clusters in a meaningful, concise and unambiguous way.

Finally, the clustering algorithm must group the results fully automatically and must not introduce a

noticeable delay to the query processing. Many approaches to search results clustering have been proposed, including Suffix Tree Clustering (STC), Semantic On-line Hierarchical Clustering (SHOC), Tolerance Rough Set Clustering (TRC), and Discover. To overcome the limitations, the goal is to group all the entity descriptions that refer to the same real world entities. A user submits a query to the middleware via a specialized Web-based interface. The middleware queries a search engine with this query via the search engine API and retrieves a fixed number (top K) of relevant web pages.

The result is a set of clusters of these pages with the aim being to cluster web pages based on association to real entity. Each resulting cluster is then processed. A set of keywords that represent the web pages within a cluster is computed for each cluster. The goal is that the user should be able to find the person of interest by looking at the sketch. The proposed work has been divided into four modules which are as follows:

i. Web pages retrieval for the query
ii. Preprocessing of web pages
iii. Clustering & its Processing
iv. Graph Creation

a. Overview of Query Processing:

Web search applications can be implemented in two different settings.

a) Server-side setting
b) Middleware setting

In server-side setting, the disambiguation mechanism is integrated into the search engine directly. On other hand in a middleware approach, build entity search capabilities on top of an existing search-engine such as Google by "wrapping" the original engine. The middleware would take a user query, use the search engine API to retrieve top K web pages most relevant to the user query, and then cluster those web pages based on their associations to real people.

The middleware approach is more common, as it is difficul to conduct realistic testing of the server-side approach due to the lack of direct access to the search engine internal data. The architecture is a pipeline that receives the input query, obtains search results from a search engine, filters the results applying a clustering algorithm and then gets the clusters.

A. Web Page Retrieval for Query:

Web Pages retrieval for query can be implemented in many ways. There are many algorithms to process Top-k retrieval, for example: Fagin's Threshold Algorithm (TA), No Random Access Algorithm (NRA) and Combined Algorithm (CA). All these threshold algorithms work on inverted indices for query terms. Assuming the vector space model, the way to fetch the top-k documents would be to compute the textual similarity of all the documents present in the corpus with the query vector, order them according to this similarity score and then fetch the top-k documents from this ordered list. However taking into consideration the huge size of the web corpus, this process becomes very unfeasible. The HttpServlet component seeks to fill this void by providing an efficient, up-to-date, and feature-rich package implementing the client side of the most recent HTTP standards and recommendations. The features are standards based, pure Java, implementation of HTTP versions 1.0 and 1.1. It is the full implementation of all HTTP methods. The fig.1 shows the process of retrieving the top pages from the search engine.

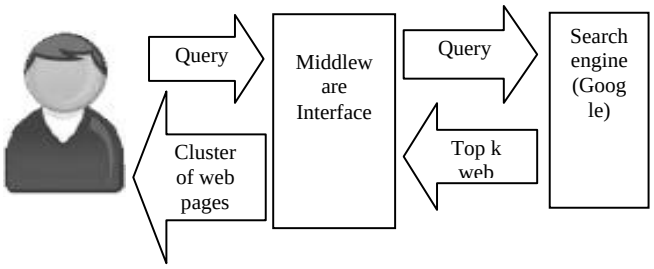


Figure 1: Web pages retrieval

B. Preprocessing of Web Pages:

After retrieving the top pages related to the query, the pages are processed by using IR techniques. There are various algorithms which are simply a set of instructions, usually mathematical, used to calculate a certain parameter and perform some type of data processing. The job is to generate a set of highly relevant documents for any search query, using the available parameters on the web. The task is challenging because the available parameters usable by the algorithm are not necessarily the same as the ones web users see when deciding if a webpage is relevant to their search. The figure 2 shows the preprocessing of the web pages which include the two processes named as stemming & stop word removal.

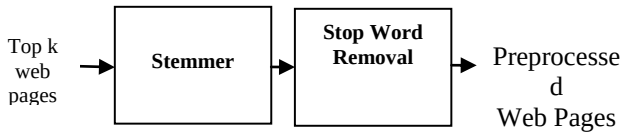


Figure 2: Preprocessing of web pages

C. Stemming:

Stemming algorithms are used to transform the words in texts into their grammatical root form, and are mainly used to improve the Information Retrieval System's efficiency. To stem a word is to reduce it to a more general form, possibly its root. For example, stemming the term interesting may produce the term interest. Though the stem of a word might not be its root, we want all words that have the same stem to have the same root. The effect of stemming on searches of English document collections has been tested extensively. Several algorithms exist with different techniques.

The most widely used is [3] the Porter Stemming algorithm. In some contexts, stemmers such as the Porter stemmer improve precision/recall scores. The stemmer operations are classified into rules where each of these rules deals with a specific suffix and having certain condition(s) to satisfy. A given word's suffix is checked against each rule in a sequential manner until it matches one, and consequently the conditions in the rule are tested on the stem that may result in a suffix removal or modification.

III. CLUSTERING AND ITS PROCESSING

When designing a Cluster Based Web Search, special attention must be paid to ensuring that both content and description (labels) of the resulting groups are meaningful to humans. As stated, "A good cluster—or document grouping—is one, which possesses a good, readable description". There are various algorithms such as K means, K-medoid but this algorithm require as input the number of clusters. A Correlation Clustering (CC) algorithm is employed which utilizes supervised learning. The key feature of Correlation Clustering (CC) algorithm is that it generates the number of clusters based on the labeling itself & not necessary to give it as input but it is best suitable when query is person names.

For general query, the algorithms are Query Directed Web Page Clustering (QDC), Suffix Tree Clustering (STC), Lingo, and Semantic Online Hierarchical Clustering (SHOC).The focus is made on Lingo because the QDC considers only the single words. The STC tends to remove longer high quality phrases, leaving only less informative & shorter ones. So, if a document does not include any of the extracted phrases it will not be included in results although it may still be relevant.

A. Frequent Phrase Extraction:

The frequent phrases are defined as recurring ordered sequences of terms appearing in the input documents. Intuitively, when writing about something, we usually repeat the subject-related keywords to keep a reader's attention. Obviously, in a good writing style it is common to use synonymy and pronouns and thus avoid annoying repetition. The Lingo can partially overcome the former by using the SVD-decomposed term document matrix to identify abstract concepts—single subjects or groups of related subjects that are cognitively different from other abstract concepts.

To be a candidate for a cluster label, a frequent phrase or a single term must:

a. Appear in the input documents at least certain number of times (term frequency threshold),
b. Not cross sentence boundaries,
c. Be a complete phrase (see definition below),
d. Not begin nor end with a stop word.

A complete phrase is a complete substring of the collated text of the input documents, defined in the following way: Let T is a sequence of elements (t_1, t_2, t_3 . . . t_n). S is a complete substring of T when S occurs in k distinct positions p_1, p_2, p_3 . . . p_k in T and Ǝi, j Є 1 . . . k : $t_{pi-1} \neq t_{pj-1}$ (left- completeness) and Ǝi, j Є 1 . . . k : $t_{pi+|S|} \neq t_{pj+|S|}$ (right-completeness). In other words, a complete phrase cannot be "extended" by adding preceding or trailing elements, because at least one of these elements is different from the rest.

An efficient algorithm for discovering complete phrases was proposed in [5], although it contained one mistake that caused the frequency of some phrases to be miscalculated. [6] The space limits make it impossible to discuss details here, for a full overview of the corrected algorithm. It does not affect further discussion of Lingo because any algorithm capable of discovering frequent phrases could be used at this stage. Figure 3 presents the whole phrase extraction phases.

Phase 2: Frequent phrases extraction

Conversion of the representation
For each document
{ Convert the document from the character-based to
The word-based representation;
}
Document concatenation
Concatenate all documents;
Create an inverted version of the concatenated documents;
Complete phrase discovery
Discover right-complete phrases;
Discover left-complete phrases;
Sort the left-complete phrases alphabetically;
Combine the left- and right-complete phrases into a set of complete phrases;
Final selection
For further processing choose the terms and phrases whose frequency exceed the Term Frequency Threshold;

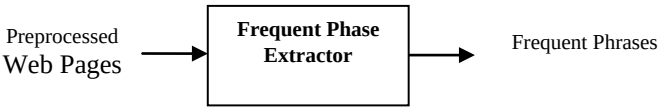


Figure 3: Frequent phrase extraction

B. Cluster label induction:

Once frequent phrases (and single frequent terms) that exceed term frequency thresholds are known, they are used for cluster label induction. There are three steps to this: term-document matrix building, abstract concept discovery, phrase matching and label pruning.

The term-document matrix is constructed out of single terms that exceed a predefined term frequency threshold. Weight of each term is calculated using the standard term frequency, inverse document frequency (tf-idf) formula [7], terms appearing in document titles are additionally scaled by a constant factor. In abstract concept discovery, Singular Value Decomposition method is applied to the term-document matrix to find its orthogonal basis.

The vectors of this basis (SVD's U matrix) supposedly represent the abstract concepts appearing in the input documents. It should be noted, however, that only the first k vectors of matrix U are used in the further phases of the algorithm.

We estimate the value of k by selecting the Frobenius norms of the term-document matrix A and its k-rank approximation A_k. Let threshold q be a percentage-expressed value that determines to what extent the k-rank approximation should retain the original information in matrix A. We hence define k as the minimum value that satisfies the following condition: $||A_k||F / ||A||F \geq q$, where ||X||F symbol denotes the Frobenius norm of matrix X. Clearly, the larger the value of q the more cluster candidates will be induced. The choice of the optimal value for this parameter ultimately depends on the users' preferences. Therefore make it one of Lingo's control thresholds—Candidate Label Threshold.

Phrase matching and label pruning step, where group descriptions are discovered, relies on an important observation that both abstract concepts and frequent phrases are expressed in the same vector space—the column space of the original term-document matrix A. Thus, the classic cosine distance can be used to calculate how "close" a phrase or a single term is to an abstract concept.

Let us denote by P a matrix of size t× (p+t) where t is the number of frequent terms and p is the number of frequent phrases. P can be easily built by treating phrases and keywords as pseudo-documents and using one of the term weighting schemes. Having the P matrix and the i^{th} column vector of the SVD's U matrix, a vector m_i of cosines of the angles between the i^{th} abstract concept vector and the phrase vectors can be calculate d:$m_i = U_i^T P$. The phrase that corresponds to the maximum component of the m_i vector should be selected as the human-readable description of i^{th} abstract concept. Additionally, the value of the cosine becomes the score of the cluster label candidate.

A similar process for a single abstract concept can be extended to the entire U_k matrix—a single matrix multiplication $M = U_k^T P$ yields the result for all pairs of abstract concepts and frequent phrases. On one hand we want to generalize information from separate documents, but on the other we want to make it as narrow as possible at the cluster description level. Thus, the final step of label induction is to prune overlapping label descriptions. Let V be a vector of cluster label candidates and their scores. We create another term-document matrix Z, where cluster label candidates serve as documents. After column length normalization calculates $Z^T Z$, which yields a matrix of similarities between cluster labels. For each row we then pick columns that exceed the Label Similarity Threshold and discard all but one cluster label candidate with the maximum score.

Phase 3: Cluster label induction
Term-document matrix building
Build the term-document matrix A for the input snippet collection.
As index terms use the non-stop words that exceed the predefined
Term frequency threshold. use the tf-idf weighting scheme;
Abstract concept discovery
Perform the Singular Value Decomposition of the term-document
Matrix to obtain U, S and V matrices;
Based on the value of the q parameter and using the S matrix -
Calculate the desired number k of abstract concepts;
Use the first k columns of the U matrix to form the U_k matrix;
Phrase matching
Using the tf-idf term weighting create the phrase matrix P;
For each column of the U_k matrix
{
multiplicate the column by the P matrix;
Find the largest value in the resulting vector to determine
The best matching phrase;
} Candidate label pruning
Calculate similarities between all pairs of candidate labels;
Form groups of labels that exceed a predefined similarity threshold;
For each group of similar labels
{
Select one label with the highest score;
}

Figure 4: LINGO – cluster label induction phase pseudo-code

C. Cluster Content Discovery:

In the cluster content discovery phase, the classic Vector Space Model (VSM) is used to assign the input documents to the cluster labels induced in the previous phase. In a way, re-query the input document set with all induced cluster labels. The assignment process resembles document retrieval based on the VSM model.

Let us define matrix Q, in which each cluster label is represented as a column vector. Let $C = Q^T A$, where A is the original term-document matrix for input documents. This way, element c_{ij} of the C matrix indicates the strength of membership of the j^{th} document to the i^{th} cluster. A document is added to a cluster if c_{ij} exceeds the Snippet Assignment Threshold, yet another control parameter of the algorithm. Documents not assigned to any cluster end up in an artificial cluster called others.

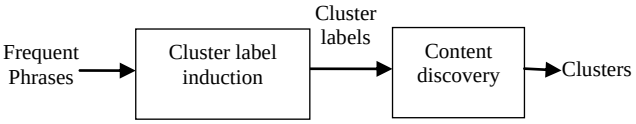


Figure 5: Cluster Formation

D. Final cluster formation

Finally, clusters are sorted for display based on their score, calculated using the following simple formula: C_{score} = label score × ||C||, where ||C|| is the number of documents assigned to cluster C. The scoring function, although simple, prefers well-described and relatively large groups over smaller, possibly noisy ones. For the time being, no cluster merging strategy or hierarchy induction is used for Lingo.

E. Design of Graph Creation:

It is a graphical approach, as it visualizes the dataset as the standard entity-relationship graph. There are other graphical disambiguation approaches, which visualize different graphs: Web Graph, Co-reference dependence graph, Entity-relationship graph (ER graph). Existing techniques are frequently based on probabilistic methodologies, application rely primarily on the mathematical apparatus from the area of Operation Research.

The suitable visualization is the ER graph. By using JGraph class objects and their relations are displayed. A JGraph object doesn't actually contain the data; it simply provides a view of the data. Like any non-trivial Swing component, the graph gets data by querying its data model.

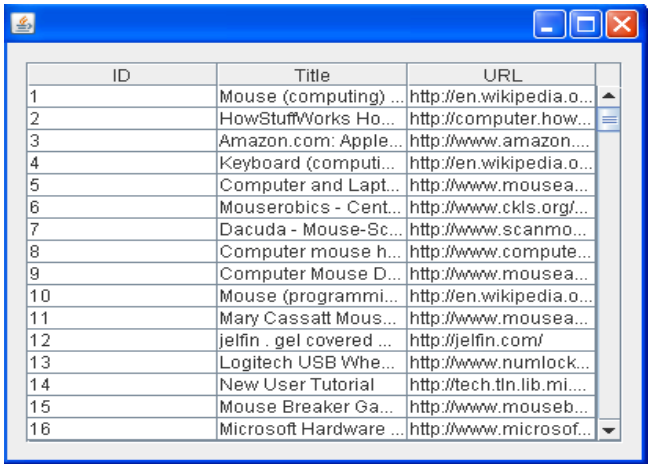
The summary line for above discussion is that this work is to help workers and researchers effectively sift through the large and often complex sets of search

results. Cluster labels reflect both proximity of search terms within and among the documents, and common terms extracted from the metadata. Table 1, Shows the excepted results/cluster names for various categories of queries.

Table 1: Clusters

Type of query	Query	Clusters
Ambiguous	Mouse	Computer mouse, Magic mouse, Cursor, gene, House Mouse, Mickey Mouse
General	Music	New music, music news, pop music, songs, Albums, Games
Compound Query	Travel to shimla	Tourism Travels, Travel Guide, shimla Tour Packages, Map, Hotels, Resorts
People Name	Pratibha Patil	Female President, President of India, Governor of Rajasthan, Photos, Videos, Visit

The Graph is used for visualization of clusters & its relevant pages. In the graph, left & right boxes show the name of clusters whereas middle boxes shows index of pages. The arrow from cluster name to web pages shows that these pages are present in the clusters. The pages are indexed by using the title & URL of results. The Graph 1 shows the graph for the query mouse & the result considered are 200.



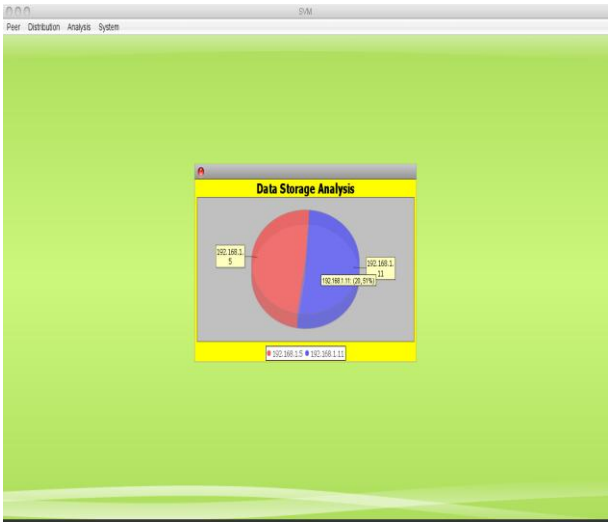
ID	Title	URL
1	Mouse (computing) ...	http://en.wikipedia.o...
2	HowStuffWorks Ho...	http://computer.how...
3	Amazon.com: Apple...	http://www.amazon....
4	Keyboard (computi...	http://en.wikipedia.o...
5	Computer and Lapt...	http://www.mousea...
6	Mouserobics - Cent...	http://www.ckls.org/...
7	Dacuda - Mouse-Sc...	http://www.scanmo...
8	Computer mouse h...	http://www.compute...
9	Computer Mouse D...	http://www.mousea...
10	Mouse (programmi...	http://en.wikipedia.o...
11	Mary Cassatt Mous...	http://www.mousea...
12	jelfin . gel covered ...	http://jelfin.com/
13	Logitech USB Whe...	http://www.numlock...
14	New User Tutorial	http://tech.tln.lib.mi....
15	Mouse Breaker Ga...	http://www.mouseb...
16	Microsoft Hardware ...	http://www.microsof...

Figure 6: Information of web pages

Graph 1: Graph for query "mouse" & results=200

F. Data Storage Analysis:

Graph 2, shows the Pie chart is drawn to show graphically how much workload being executed by different nodes in terms of percentage & number of workloads.



Graph 2: Pie Chart of Data storage analysis as per distribution

IV. INFLUENCE OF THE NUMBER OF RESULTS

The number of outputs processed for a single query is likely to have impact on two major aspects of the results: the quality of groups' description and the time spent on clustering. Having more input documents may eliminate some of the very specific phrases (which in many cases turn out to be misleading) replacing them with their more general counterparts. Inevitably, the increased quality of description will be achieved at the cost of longer processing times (not only will the number of documents increase but also there will be more indexing terms). Below we compare clustering results for the same query but different numbers of input. The table 2 shows the effect of number of results (50 & 200) for the same query "Data mining".

Table 2: Clustering Results for 50 & 200 results (for query "Data Mining")

Query	Data mining	Data mining
Source	Google	Google
Input	50	200
Clustering Results	Data Mining Software, Data Mining Research, Data Mining & Analytics, Tools, Knowledge Discovery & Data Mining, Algorithms, Data Analysis, Extraction,	Data Mining Research, Data Mining Tools, Knowledge Discovery, Algorithms Data Mining, Data Mining Applications, Data Mining Process, Statistical Data Mining, Data Analysis, Computing, Machine Learning, Program, Web Mining, Data Mining Books, Data Mining Related, Data Mining Services, Finding, Text Mining, Data Mining Methods, Information

	Forum, Introduction, Predicative Analytics, Relational Data Mining, Concepts & Techniques, Conference on Data Mining, DATA-MINING-CUP, Data Mining, Notes, Oracle Data Mining, Practical Machine Learning Tools & Techniques, Visualization & Social Media, Wikipedia, Other Topics (735 ms)	Technology, Oracle Data Mining, Solutions, Structure Mining, Workshop, Data Mining, Introduction to Data Mining, IEEE International Conference on Data Mining, Association ,Clustering, Predicative Analytics, Extracting Useful, Privacy, Review, Concepts & Techniques, Google Buchsuche-Ergebnisseite, IBM, Open Source, SQL Server, Microsoft & Central Labs, National Center for Data Mining, Principles of Data Mining, Statoo Consulting, Visualization & Social Media, Other Topics (891 ms)

The most striking difference between the above groupings is that with the increase in the number of results, the number of groups is also significantly larger. The most obvious explanation for this lies in the fact that new input documents simply introduce new topics. Indeed, the results on the right (200) contain a number (e.g." IEEE International Conference on Data Mining"," Data Mining Books ", "National Center for Data Mining") of groups absent from the grouping on the left. Closer examination of the "IEEE International Conference on Data Mining " group reveals that all its members are in the third hundred of snippets and thus were not available in the 50-results setting.

The accuracy of cluster description seems to be better in the 200- results. The "Untangling Text Data Mining" label is too specific, whereas its equivalent in the 200- results setting – "Text Data Mining" – is much better. A similar pair can be "Tools" and "Data Mining Tools". The reason for this may be that in the 200- results grouping more documents match the discussed clusters somehow enforcing a more general description. Finally, increasing the number of input results severely affects the processing times. In the 50- results setting the clustering process took mere 735 ms, while processing of 200 input documents required almost ten times as much time – 891 ms.

a. Evaluation of Search Results Clustering:

The focus is made on the evaluation of usefulness of generated clusters. The term usefulness involves very subjective judgments of the clustering results. For a set of groups created in response to a single query, evaluated the following:

A. Usefulness of Clusters:

For each created cluster, based on its label, decided whether the cluster is useful or not. Useful groups would most likely have concise and meaningful labels, while the useless ones would have been given either ambiguous (e.g. too long) or senseless (e.g. consisting of a single stop word) descriptions.

Scale: useful group | useless group

B. Assignment Precision:

For each cluster individually, for each snippet from this cluster, judged the extent to which the result fits its group's description. A very well matching result would contain exactly the information suggested by the cluster label. A moderately matching result, despite being e.g. more general, would still be somehow related to the group's topic. A non-matching snippet, even though it might contain several words from the group's label, would be completely irrelevant to its cluster. It is important that groups which have previously been assessed as useless be excluded from this step – if a user cannot understand the description of a group they will most likely be unable to evaluate its contents either.

Scale: very well matching | moderately matching | not matching

The table 3 shows the parameters used for the evaluation of queries & their assigned pages. In the table numbers shows the value of parameters for their respective query. Based on the values obtained, the measures are done for queries which are shown in table 4. To decide the cluster label quality the parameters used was u & g. For Assignment coverage, s & O was used. The parameters a & s were used for cluster overlap.

Table 3: Evaluation parameters with results

Query / Parameters	**Mouse & results=200**	**Data mining & results= 200**	**Travel +shimla & results =196**	**No one know what he can do until he tries & results= 200**
g=the total no of created groups	29	43	42	43
u=the total no of groups judged	21	32	35	23
a=the total no of result assignments	160	333	480	318
w=the total no of result judged as very well match	93	171	275	71
m=the total no of result judged as moderately match	15	65	119	34
n=the total no of result judged as non matching	8	21	20	25
S=the total no of results	200	200	196	200
O=the total no of results confined to "Other Topics "	0	26	28	42

Table 4: Measures for cluster Quality

Query **Measures**	**Mouse**	**Data mining**	**Travel+ shimla**	**No one knows what he can do until he tries**
Cluster Label Quality	More than 70% Clusters are useful.	The 74% Clusters are useful.	The 83% Clusters are useful.	The 53% Clusters are useful.
Assignment Coverage	All the resul	87% of results are assigned to the clusters.	86% of results are assigned to the clusters.	79% of results are assigned to the clusters.
Cluster Overlap	80% of input results are assigned to more than one cluster.	All input results are assigned to more than one cluster.	All input results are assigned to more than one cluster.	All input results are assigned to more than one cluster.

V. CONCLUSION

Cluster base web search approach using support vector machine is useful where one can easily, efficiently and effectively get the clustered results. The quality of clusters obtained is more significant than other approaches of web search. Hence we can summaries few points as follows:

a. The use of phrases in the process of cluster label induction guarantees that group descriptions can be easily understood by the users. Also frequent phrases significantly increase the overall quality of clustering, not only of the phrase-based algorithms (such as Suffix Tree Clustering) but also of other approaches such as k-means. Similar effects can be observed also in Cluster Based Web Search.

b. Apart from the general abstract concepts related to fairly large groups of documents, Latent Semantic Indexing discovers narrower, more-specific ones. In this way meaningful clusters can be created whose labels are not necessarily the highest-frequency phrases. Additionally, the orthogonally of the SVD-derived abstract concept vectors makes the diversity among cluster labels even wider.

c. Placing the same document in a larger number of clusters increases the chances that, viewing only selected groups, the user is be able to identify all relevant documents. Moreover, some snippets may be related to two or more topics and thus should be placed in all respective clusters.

d. As all the phases of system are easily separable. Thus, it is possible to provide alternative implementations of some of them, improving the quality or time-efficiency of the algorithm as a whole.

e. Using the concept of SVM the distribution would be easily possibly which helps in getting results more faster and also balances the load of system from being unstable by executing the number of requests at a time.

VI. REFERENCES

[1]. Kalashnikov D.V., Mehrotra S., R.N.Turen and Z.Chen.,"Web People Search via Connection Analysis" IEEE Transactions on Knowledge and data engg.Vol 20, No11 ,in Nov 2008.

[2]. Kalashnikov D.V., Mehrotra S., Z. Chen, Nuray-Turan R., and Ashish N., "Disambiguation Algorithm for People Search on the Web," Proc. IEEE Int'l Conf. Data Eng. (ICDE '07), in April 2007.

[3]. Porter M. F. "An algorithm for suffix stripping", Program Vol. 14, no. 3, paper presented 130-137.

[4]. Osinski Stanis law, Stefanowski,Jerzy and Weiss Dawid.," Lingo: Search Results "Clustering Algorithm Based on Singular Value Decomposition".

[5]. Zhang Dong,"Towards Web Information Clustering". PhD thesis, Southeast University, Nanjing, China, in 2002.

[6]. Osinski S. l, "An Algorithm for Clustering of Web Search Results". Master's thesis, Pozna´n University of Technology, Poland. .2003. 58

[7]. Salton G., "Automatic Text Processing — The Transformation, Analysis, and Retrieval of Information by Computer.", in 1989.

[8]. Zamir Oren E., "Clustering Web Documents: A Phrase-Based Method for Grouping SearchEngine Results". Doctoral Dissertation, University of Washington. , in 1999.

[9]. Stefanowski Jerzy and Weiss Dawid, "Web search results clustering in Polish- Advances in Soft Computing, Intelligent Information Processing and Web Mining", Proceedings of the International IIS: IIPWM´03 Conference, Zakopane, Poland, vol. 579 (XIV), , pp. 209-22, in 2003.