SoftBotSearch : SoftBot Approach to information retrieval from Web

A. Shubham Agarwal
ssa@ug.iiita.ac.in

Abstract-In this paper, we describe SoftBotSearch, an Internet softbot which organizes information retrieved from search engines in a form that is more useful to humans. SoftBotSearch meta-searches the web using three popular search engines, retrieves the web pages from the Internet and groups them according to their theme. Thus a search for "operating system" automatically organizes the results into categories like Microsoft, open source resources, distributed operating systems, memory management etc. We use the K-Means and Subtractive Clustering algorithms to find clusters in the document vector space. In this paper we also define methods used for clustering including suffix tree clustering algorithm used. SoftBotSearch can be easily adapted to work on any information source.

Keywords: "Intelligent Information retrival", "SoftBotSearch" ,"Web Searching" **.**

I. Introduction

A. *Motivation*

Finding things is easier when they are organized. This applies equally well to the web. Considering the enormous volume of information available on the Internet today, and the rate at which it is growing, we are motivated towards building a system that would bring order to this chaos. Almost all search engines in wide use today accept a set of words as the search query and return hundreds of documents that contain those words. For a person who is *researching* in some particular field, say *"face recognition"*, such ranked list of documents makes little sense. He must sift through many such documents before he is able to grasp the nuances in the field. This suggests moving up the *information food chain* by the use of softbots which harness the conventional search engines of today to produce more informative results from queries. Etzioni [1] draws an interesting metaphor in which he compares the vast amount of information on the web with grass, and the search engines with cows that graze on this grass. Softbots like SoftBotSearch are at the top of the information food chain, and represent a higher level of intelligence over search engines. SoftBotSearch aims to solve the problem of making sense out of thousands of documents available on the Internet on any conceivable topic by arranging the results into groups containing documents with similar content. Each group is assigned a topic - a set of phrases that best describes the documents in that group. Thus, a query for "e-commerce" would return back topics such as "how to start", "credit card processing", "logo design, website design", "credit card, merchant account" etc. Similarly, a search for "face recognition" would organize the results under "surveillance systems", "can face recognition systems keep airports safe", "neural networks", "link matching techniques" etc. Such a grouping would help the user to choose his topic of interest and then sift through the documents in that group, thus greatly reducing the burden of going though numerous documents returned by a typical search engine. Each cluster must be assigned a topic that best describes the documents in that group. This requires extracting representative phrases from the documents in each group. Finally assigning phrases in document is done through suffix tree clustering described in section 7.

September 12, 2005 This work was done as part of project work at IIIT-A Allahabad, India under guidance of Dr.S.Sanyal (email:ssanyal@iiita.ac.in)

Shubham Agarwal is with Indian Institute of Information Technology Allahabd, India doing B-Tech in 7th semester.(phone:09839619893,email:ssa@ug.iiita.ac.in) .

B. *Previous Work*

One of the most popular meta-searching tool is MetaCrawler [2] which provides an expressive query language and queries through nine popular search engines in parallel. MetaCrawler frees users from having to remember the intricacies of individual search engines but does not use any "intelligence" of its own. I merely aggregates the results obtained from difference search engines and produces a single ranked list of documents.

Grokker [3] is a commercial product that closely resembles SoftBotSearch. It sends the query to multiple search engines and uses the text snippets from search engine results to organize the documents in a hierarchical fashion. Then it applies linguistic and statistical methods to assign topics to each cluster, with the help of a partial parser. Unlike Grokker, SoftBotSearch is light-weight in the sense that no parts-of-speech tagging and parsing is required to discover clusters, and to assign appropriate topics to them.

iBoogie [8] is a web based interface, which produces a list of documents like a typical search engine, and also creates groups which are relevant to the given query. Both Grokker and iBoogie are based on the Clusterizer Technology [4].

C**.** *Organization of the paper*

The rest of the paper is organized as follows. In Section 2, we describe the architecture of SoftBotSearch. In section 3, we describe the pre-processing steps applied before performing clustering. Section 4 explains the Document Vector space model. In section 5, we touch upon Principal Component Analysis (PCA). Section 6 describes the clustering step. In section 7, we present details of our topic extraction that is suffex tree clusturing. We report the results obtained by

SoftBotSearch in section 8. Finally, we present our conclusions and scope for further improvement in section 7.

II. Architecture

A. *How SoftBotSearch Works*

1) Fetching Documents

Given a query, SoftBotSearch passes the query terms to multiple search engines – Google, Yahoo and MSN Search. The web pages corresponding to the top 100 results returned by each search engine are then fetched from the Internet. Since web pages are in HTML format, they are parsed and only the important sections – title, keywords, description, sub-titles, and bold items on the web page are retained, as they are sufficient to characterize the theme of the document.

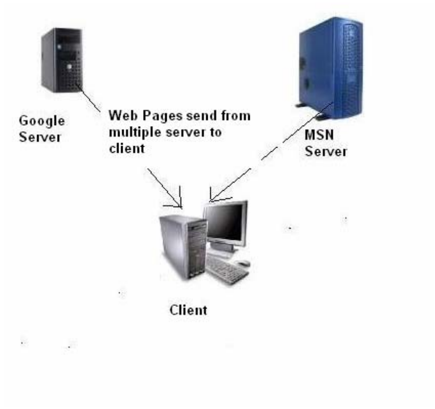


Fig 1: Fetching Documents

2) Preprocessing

Pre-processing involves removing stop words and then stemming the remaining words so that they are reduced to their root form.

3) Processing

After preprocessing, a document matrix is created, and words appearing too rarely or too frequently are removed. Then the word frequency scores are converted to TFIDF scores and PCA is applied to the matrix for dimensionality reduction..

4) Clustering

Subsequently, clustering is used to produce non-overlapping groups of documents. K-Means is one of the most popular clustering algorithms mentioned in literature, mainly because of its simplicity and low computational cost.

5) Topic Extraction

Each group is assigned a set of phrases which best describes the documents in that group.

6) Display

Once topics are assigned to each group, the results are shown to the user in a graphical form. The user can click on

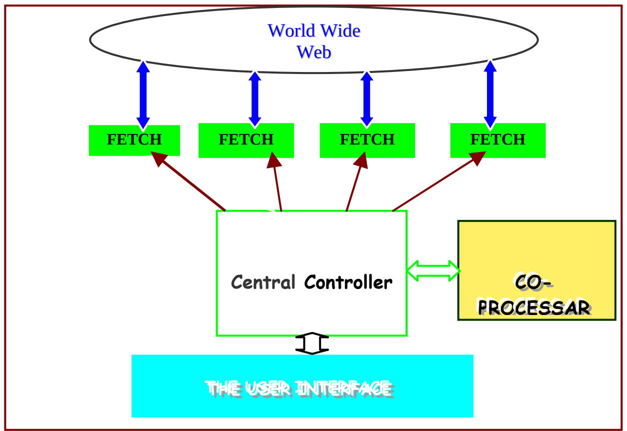


Fig 2:System Architecture

any particular group to find more information about browse Documents in that group .

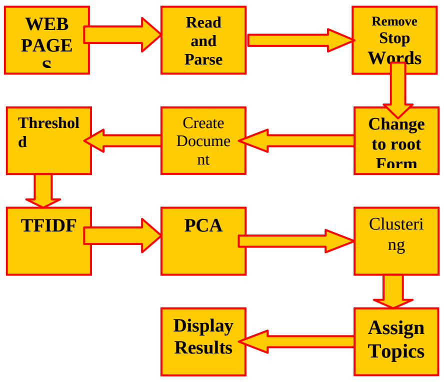


Fig 3:Logical Steps.

III. Preprocessing

A. Stop words removal

Words like articles, prepositions, conjunctions, common verbs (e.g. ‘know’, ‘see’, ‘do’, ‘be’), auxiliary verbs, adjectives (e.g. ‘big’, ‘late’, ‘high’), pronouns, and other such words which do not carry much semantic value are removed, leaving only content words likely to have some contribution towards the theme of the document. This process condenses the vocabulary and thus reduces computational cost of further processing on it. Stop words removal helps to enrich the vocabulary with “good” words which semantically play an important role in determining the theme of the document which we frequently come across on websites but do not help much in conveying the theme of the document.

B. Stemming

The words are passed through a stemmer which reduces various instances of a single word to the root form. e.g. flying and flied are reduced to fly. Words in their different morphological forms (past tense, singular, plural, etc.) do not modify the theme of the document. Hence it makes sense to reduce such words to their “roots” to obtain better similarity measures between documents. Stemming involves suffix stripping while following certain rules which take care of various forms of plurals, verb formation etc. We use the Porter Stemming algorithm [5] to stem words before putting them into the vocabulary.

Stop words removal and stemming drastically affect the quality of the clusters and results obtained by clustering.

IV. Vector Space Model

A. *Term Document Matrix*

The most commonly used method for representing documents is the vector space model [9]. Each document is represented as a vector of length N, which is the size of the vocabulary. The ith element of the vector denotes the frequency of occurrence of the ith word of the vocabulary in the document. We construct a term document matrix which depicts the occurrence of each word in each document.

TABLE 1
A SAMPLE TERM DOCUMENT MATRIX

D1	D2		D3
computer	3	8	2
network	7	7	5
neural	4	3	3
security	4	3	4

The rows represent words from the vocabulary and the columns represent each document, three in this case. It is evident from the matrix in Table 1 that the word computer appears 3 times in the first document, 8 times in the second, and so on. documents D, and number of documents d in which the given word occurs at least once. Using TFIDF approach has the effect of giving more weight to terms that occur multiple times in the given document, but which are not so common as to occur in too many.

Term Frequency of a word denotes the number of times that word occurs in the document. Inverse document frequency is the log of the ratio of total number of documents. Thus TFIDF gives higher values to terms that have more discriminatory power.

TFIDF = TF * IDF

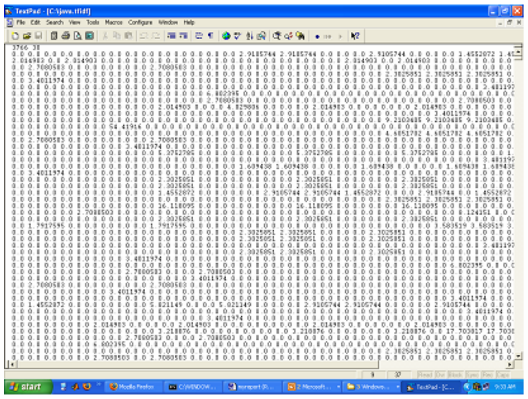

[illegible]

Fig 4:Document Matrix Screen Shot.

B. *Thresholding*

Once the term document matrix is created, we reject those words which occur below or above a certain threshold. Terms occurring in less than 2% documents are rejected since they are too unique to help in classification. Similarly, terms found in more than 30% documents are also rejected since they would be too common to help is discriminating documents.

These values – 2% and 30%, have been chosen manually by analyzing the words that comprised the vocabulary. A lower bound less than 2% invariably caused noise words to be included in the vocabulary. Similarly, an upper bound greater than 30% included unique words that led to the inclusion of undesirable words. E.g. when searching for "operating system", the term "operating" occurred in 39% of the documents. Note that we do not desire the presence of the word "operating" while creating the clusters, since it is part of the search query itself and hence does not help in distinguishing between documents.

C. *TFIDF*

Instead of using binary values representing presence or absence of words, we use the Term Frequency – Inverse Document Frequency or TFIDF scores [9].

The features that make up the document vector can be strongly correlated with each other. It is generally desirable to find or reduce the feature set to one that is minimal but sufficient. This can be done by judicious elimination or by the Principle Components Analysis (PCA) technique amongst others. PCA can be used to reduce the feature vector dimension while retaining most of the information by constructing a linear transformation matrix. The transformation matrix is made up of the most significant eigenvectors of the covariance matrix.

V. PCA

PCA is a useful statistical dimensionality reduction technique that has found application in fields such as face recognition and image compression, and is a common technique for finding patterns in data of high dimension. A preliminary introduction to PCA can be found in [6].

The document vectors constructed in the previous step are used to compute the eigenvectors by Singular Value Decomposition. The top k eigenvectors (having the greatest eigenvalues) are selected which are then used to transform document vectors to the reduced vector space.

VI. Clustering

Once we have obtained the document vectors in the reduced space, We are ready to discover groups such that documents in one group are similar to each other. Although various clustering methods are available, I have used the following algorithms, since they suited our requirements of fast and accurate Clustering.

A. *Key requirement of clustering Steps Involved*

i) Identify groups of documents that are similar to each other more than they are similar to the rest of the collection.

ii) Measure of similarity.

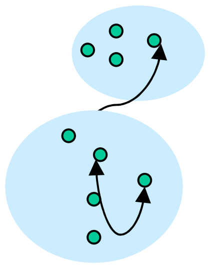


Fig 5. Kmeans Representation.

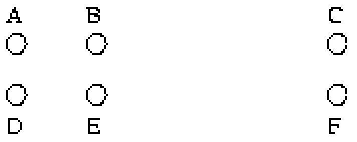
B. *K-Means Clustering*

The K-Means clustering algorithm produces K clusters in the data, by simply assigning each point to the cluster whose centroid is closest to it. Centroids are recomputed as soon as a new point is added to the cluster. Initially, K randomly chosen

points are designated as the seed points, or initial clusters. These seed points are initialized using a linear time algorithm called Subtractive Clustering.
The main benefit of K-Means clustering is that it runs in linear time unlike agglomerative clustering algorithms which run in quadratic or cubic time, depending on the linkage used. The linear time of K-Means is a huge advantage when I consider the fact that thousands of documents may need to be clustered in a high-end clustering program.
ALGO:-

```
K-means-cluster (in S : set of vectors : k : integer)
{  let X[1] ... X[k] be k random points in S;
   repeat {
         for j := 1 to N {
           X[q] := the closest to S[j] of X[1] ... X[k]
           Add S[j] to C[q]
          }
         for i := 1 to k {
           X[i] := centroid of C[i];
           C[i] := empty
          } }
   until the change to X is small enough.
}
```

Have to guess K.
Local minimum. Example: In diagram below, if K=2, and you start with centroids B and E, converges on the two clusters {A.B.C}, {D,E,F}



Disjoint and exhaustive decomposition.
Starvation: Complete starvation of C[j], or starvation to single outlier.
Assumes that clusters are spherical in vector space. Hence particularly sensitive to coordinate changes (e.g. changes in weighting)

VII. Suffix Tree Clustering

. For assigning topics I have used STC for extracting relevant phrases from the cluster of documents.Their are two ways of extracting topics simple keywords based and phrase extraction.The first approach requires one to extract most relevant word from the set of document .In second approach I need to extract phrases from set of documents .As phrases can more clearly distinct the documents hence I have used STC for phase extraction.

Steps Involved

Document phrasing
Phrase cluster identification
Phrase cluster merging

Step 1 – Document Phrasing

- Each document is transformed into a sequence of words and phrase boundaries are identified
    - Perform stemming
        - Cleaning clean, items item
    - Mark sentence boundaries
        - Punctuation . and HTML tags
    - Maintain word ordering
        - Football player not player football !

Step 2 – Phrase Cluster Identification

- The STC algorithm identifies all maximal phrase clusters

    1. Build a suffix tree
    2. The identification of phrases can be viewed as the creation of an inverted index of phrases
    3. The phrase clusters are scored

VIII. Results

A. SoftBotSearch Results

We present some of the prominent cluster topics produced by SoftBotSearch when presented with the following queries:
"operating system"
1) microsoft windows
2) red hat, open source, linux operating system, gnu
3) parallel and distributed operating systems
4) extremely reliable operating system .
5) memory management
3) credit card processing
4) logo design, graphic design, website design, domain name
5) credit cart, merchant account

"clustering"
1) high availability clustering .
2) Beowulf training .
3) search engine, document clustering
4) software for clustering technologies .
6) threading technology

"e-commerce"
1) how to start, guide, want to start
2) open source

B. *Screenshot*

The SoftBotSearch(SoftBotSearch) in action for the query made by user operating system we get different clusters . These clusters formed are at runtime and thus may change when query is made at some other time.

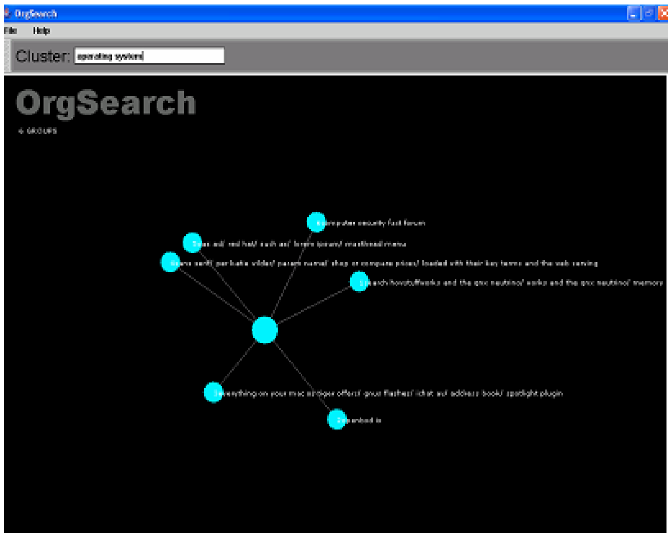


Fig 6.Query made is operating system.

C. *Effect of preprocessing*

Various combinations of stemming, stop words removal, and different values of thresholds not only affect the size of the vocabulary but also the quality of the clusters. e.g. although the quality of clusters was found to be invariant to increase in the upper threshold, decreasing the lower threshold to 1% added noise into the vocabulary, and many spurious clusters were formed. Table 2 shows the vocabulary size under different configurations.

TABLE 2
VOCABULARY SIZE UNDER DIFFRENT CONFIGRATIONS

Stemming	Stop words removal	Threshold	vocab
Yes	Yes	2%, 25%	2
Yes	Yes	1%, 20%	669
Yes	No	2%, 25%	232
No	No	2%. 25%	219

IX. Conclusion

In this paper, we described Web Clustering as a means to present organized information to the user. A proper application of preprocessing techniques and clustering methods can lead to good quality clusters of web documents. For assigning topics to clusters, we described a novel algorithm that generates sufficiently good descriptive phrases without requiring parsing techniques. The most important thing currently lacking in SoftBotSearch is Hierarchical Clustering. Hierarchies of documents make more sense to humans than flat groupings. But hierarchical clustering requires more semantic information to assign appropriate topics to nodes that are higher in the hierarchy. E.g. in a hierarchical clustering program, a search for "face recognition" should create one of the clusters as "techniques" which in turn will contain "neural networks", "eigenfaces" etc. To discover that "neural networks" and "eigenfaces" represent techniques of face recognition would require additional semantic processing.

Secondly, our Internet softbot is not personalized. It is not able to adapt to the likes and dislikes of individual users. Using relevance feedback techniques, a user could be asked to rate each cluster, and this information could be used for enhancing future results. Such functionality would surely make the bot look more intelligent.

References

[1] O. Etzioni. Moving up the information food chain: Deploying softbots on the world-wide web. In Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI96) , Portland, OR, 1996.
[2] E. Selberg, and O. Etzioni, The MetaCrawler Architecture for Resource Aggregation on the Web, IEEE Expert, January/February 1997
[3] Grokker, http://www.grokker.com
[4]Clusterizer Technologies, http://www.clusterizer.com
[5] M..Porter.An Algorithm for Suffix Stripping. Program (Automated Library and Information Systems), 14(3). 1980. 130-137.
[6] L. Smith, A tutorial on Principal Component Analysis http://www.cs.otago.ac.nz/cosc453/student_tutorials/ principal_components.pdf
[7] S. L. Chiu. Method and software for extracting fuzzy classification rules by subtractive clustering. In Proceedings of North American Fuzzy Information Processing Society Conf. (NAFIPS'96), June 1996.
[8] iBoogie Search Engine, http://www.iboogie.com
[9] A. Singhal, and G. Salton, Automatic Text Browsing Using Vector Space Model. In Proceedings of the Fifth Dual-Use Technologies and Applications Conference, 318-324, Utica/Rome