Automatic Discovery of Association Orders between Name and Aliases from the Web using Anchor Texts-based Co-occurrences

#V.REDYA JADAV[1], Associate Professor, Computer Science and Engineering Department
#M.NAGESWARA RAO[2], Assistant Professor Computer Science and Engineering Department
#M.LAXMANKUMAR[3] M.Tech, Computer Science and Engineering Department
Bomma Institute of Technology and Science, Khammam, A.P State, INDIA

Abstract— Now a days, searching for people on web is the most common activity done by most of the users. When we give a query for person search, it returns a set of web pages related to distinct person of given name. For such type of search the job of finding the web page of interest is left on the user. In this paper, we develop a technique for web people search which clusters the web pages based on semantic information and maps them using ontology based decision tree making the user to access the information in more easy way. This technique uses the concept of ontology thus reducing the number of inconsistencies. The result proves that ontology based decision tree and clustering helps in increasing the efficiency of the overall search.

An individual can be referred by multiple name aliases on the web. Extracting aliases of a name is important in information retrieval, sentiment analysis and name disambiguation. We propose a novel approach to find aliases of a given name using automatically extracted lexical pattern based approach. We exploit set of known names and their aliases as training data and extract lexical patterns that convey information related to aliases of names and extract large set of candidate aliases from text snippets returned by web search engine. We define numerous ranking scores to evaluate candidate aliases using three approaches: lexical pattern frequency, word co-occurrences in an anchor text and page counts on the web. We introduce notion of a word co-occurrence graph to represent mutual relations between words that appear in anchor text, words in anchor text are represented as nodes in the co-occurrence graph and edge is formed between nodes which link to the same url. The drawback of the existing method is the extracted alias names may be a original of some other person. So we introduce Email id extraction, by this we can overcome the problem. To construct a robust alias detection system, we integrate ranking scores through support vector achines using a single ranking function. Moreover, the aliases extracted using the proposed method are successfully utilized in information retrieval task to improve recall by 20 percent in a relation detection task.

Finding information about people on the Web using a search engine is difficult because there is a many-to-many mapping between person names and specific persons (i.e. referents). This paper describes a person resolution system, called **Web Hawk**. Given a list of pages obtained by submitting a person query to a search engine, **Web Hawk** facilitates person search in three steps: First of all, a *filter* removes those pages that contain no information about any person. Secondly, a *cluster* groups the remaining pages into different clusters, each for one specific person. To make the resulting clusters more meaningful, an *extractor* is used to induce query-oriented personal information from each page. Finally, a *namer* generates an informative description for each cluster so that users can find any specific person easily. The architecture of **Web Hawk** is presented, and the four components are discussed in detail, with a separate evaluation of each component presented where appropriate. A user study shows that **Web Hawk** complements most existing search engines and successfully im-proves users' experience of person search on the Web.

Index Terms— Anchor Text mining, Graph Mining, Word Co-occurrence Graph, *Clustering*, *Semantic information*, *Efficiency*, Web mining, information extraction, Relation Extraction

1. Introduction

Searching for information about people in the web is one of the most common activities of internet users. Around 30 percent of search engine queries include person names [1], [2]. However, retrieving information about people from web search engines can become difficult when a person has nicknames or name aliases. For example, the famous Japanese major league baseball player Hideki Matsui is often called as Godzilla on the web. A newspaper article on the baseball player might use the real name, Hideki Matsui, whereas a blogger would use the alias, Godzilla, in a blog entry. We will not be able to retrieve all the information about the baseball player, if we only use his real name. Identification of entities on the web is difficult for two fundamental reasons: first, different entities can share the same name (i.e., lexical ambiguity); second, a single entity can be designated by multiple names (i.e., referential ambiguity). For example, the lexical ambiguities consider the name Jim Clark. Aside from the two most popular namesakes, the formula-one racing champion and the founder of Netscape, at least 10 different people are listed among the top 100 results returned by Google for the name.

1..1 Information retrieval

This paper mainly deals with information retrieval system. Information retrieval is the area where users might search for documents, information within documents and metadata from documents on the web. Many users query might include retrieval of documents for personal names. Many celebrities and experts from various fields are referred by their original names on web. Most of the queries to web search engines include person names [1] [2]. For example, people might use *"Michel Jackson"* as a query on search engine to know about him. The search engine might give the relevant documents met the information need of the user's query. Apparently celebrities and experts might also be referred by their aliases on the web. Many web pages about person names might also be created by aliases. For example, a newspaper article might refer the persons using their original names, whereas a blogger might refer them using their nick names. The user will not be able to retrieve all information about a person if he only uses his personal name. To retrieve complete information about a person name, one might know about his aliases on the web. Various types of words are used as aliases on the web. Identifying aliases will be helpful in information retrieval. The aliases are extracted using previously proposed alias extraction method. The search engine expands the query on person names by tagging the extracted aliases to retrieve relevant web pages those are referred by original names as well as aliases thereby improving recall and MRR.

1..2 Outline of the proposed approach

The proposed method will work on the aliases and get the association orders between name and aliases to help search engine tag those aliases according to the orders such as first order associations, second order associations etc so as to substantially increase the recall and MRR of the search engine while searching made on person names. The term recall is defined as the percentage of relevant documents that were in fact retrieved for a search query on

search engine. The mean reciprocal rank of the search engine for a given sample of queries is that the average of the reciprocal ranks for each query. The term word co-occurrence refers to the temporal property of the two words occurring at the same web page or same document on the web. The anchor text is the clickable text on web pages, which points to a particular web document. Moreover the anchor texts are used by search engine algorithms to provide relevant documents for search results because they point to the web pages that are relevant to the user queries. So the anchor texts will be helpful to find the strength of association between two words on the web. The anchor texts-based co-occurrence means that the two anchor texts from the different web pages point to the same the URL on the web. The anchor texts which point to the same URL are called as inbound anchor texts [3]. The proposed method will find the anchor texts-based co-occurrences between name and aliases using co-occurrence statistics and will rank the name and aliases by support vector machine according to the co-occurrence measures in order to get connections among name and aliases for drawing the word co-occurrence graph. Then a word co-occurrence graph will be created and mined by graph mining algorithm so as to get the hop distance between name and aliases that will lead to the association orders of aliases with the name. The search engine can now expand the search query on a name by tagging the aliases according to their association orders to retrieve all relevant pages which in turn will increase the recall and achieve a substantial MRR.

Our approach exploits the task of searching the people in more precise manner. Web people search clusters the web pages based on the query fired. Clustering is the process of gathering similar objects in one cluster

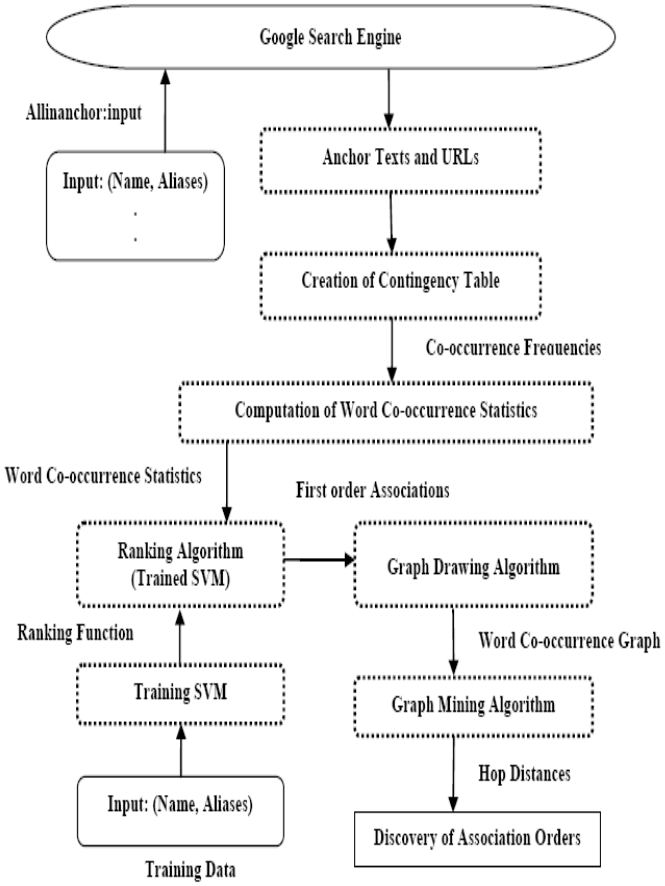

Fig : Outline of the proposed method

which are different to the objects in another cluster. Web pages having semantic information are grouped in a single cluster. Later, these clusters are presented to the user in form of ontology. Ontology is a set of concepts such as things and relations that are specified in some way in order to exchange information. Firstly input is given to the search engine in form of query. The search engine returns the top k relevant pages as soon as the query is fired. Web pages are retrieved are processed. The pre processed web pages are clusters and then presented using decision tree. The system can be made more generalized for the search of people with the help of filters.

The main objective of this paper is to understand the basic concepts of ontology

with particular emphasis on its application to people search problem.

2. Review of the Past Work

Initially, for searching the Page Rank algorithms were used for ranking the search query results. The page rank algorithm was described by Lawrence Page and Sergey Brin. Working of the Page Rank algorithm depends upon link structure of the web pages. The Page Rank algorithm is based on the concepts that if a page contains important links towards it then the links of this page towards the other page are also to be considered as important pages. The Page Rank considers the back link in deciding the rank score. But the limitation is that if new page is inserted between two pages then the crawler should perform a large calculation to calculate the distance vector which is a time consuming process and decreases the performance. Taher Haveliwala in 2002 proposed a Topic Sensitive Page Rank as compared to the original Page Rank for improving the search-query results where a single Page Rank vector is computed using the link structure of the web to compute the relative importance of the page [4]. Hearst and Pedersen showed that relevant documents tend to be more similar to each other, thus the clustering of similar search results helps users find relevant results. In addition, Vivisimo is a real demonstration of this technique. Vivisimo was founded in 2000 by three Carnegie Mellon University scientists who decided to tackle the problem of information overload in web search. Rather than focusing just on search engine result ranking, we realized that grouping results into topics, or "clouds," made for better search and discovery. As search became a necessity for web users, Vivisimo developed a service robust enough to handle the variety of information the everyday web user was after. The result was Clusty: an innovative way to get more out of every search. Clusty was acquired by Yippy, Inc. in May 2010. Yippy queries several top search engines, combines the results, and generates an ordered list based on comparative ranking. Jargon Went and Jian-Yun Nie proposed a new approach to query clustering using user logs[3]. The principles are as follows. 1) If users clicked on the same documents for different queries, then the queries are similar. 2) If a set of documents is often selected for a set of queries, then the terms in these documents are related to the terms of the queries to some extent. These principles are used in combination with the traditional approaches based on query contents.

Our proposed local-cluster algorithm considers linkage structure and content generation of cluster structures to produce a ranking of the underlying clusters with respect to a user's given search query and preference. The rank of each document is then obtained through the relation of the given document with respect to its relevant clusters and the respective preference of these clusters.

3. Related Work

Alias identification is closely related to the problem of cross-document co reference resolution in which the objective is to determine whether two mentions of a name in different documents refer to the same entity. Bagga and Baldwin [10] proposed a cross-document co reference resolution algorithm by first performing within document co reference resolution for each individual document to extract co reference chains, and then, clustering the co reference chains under a vector space model to identify all mentions of a name in the document set. However, the vastly numerous documents on the web render it impractical to perform within document co

reference resolution to each document separately, and then, cluster the documents to find aliases

In personal name disambiguation the goal is to disambiguate various people that share the same name (namesakes) [3], [4]. Given an ambiguous name, most name disambiguation algorithms have modeled the problem as one of document clustering in which all documents that discuss a particular individual of the given ambiguous name are grouped into a single cluster. The web people search task (WePS)1 provided an evaluation data set and compared various name disambiguation systems. However, the name disambiguation problem differs fundamentally from that of alias extraction because in name disambiguation the objective is to identify the different entities that arereferred by the same ambiguous name; in alias extraction, we are interested in extracting all references to a single entity from the web.

The Web People Search challenge is closely related to the well-studied Entity Resolution problem. In our previous work we also have developed interrelated techniques to solve various Entity Resolution challenges, e.g. [14–16]. The approach covered in the paper, however, is not related to those techniques. The key algorithm for training the skyline-based classifier is new. In fact, we are unaware of any Entity Resolution technique that would use a similar approach under any context. There are several research efforts that address specifically Web Person Search and related challenges [1–3, 5, 6, 8,20,21, 23–25]. The approach of [5] is based on exploiting the link structure of pages on theWeb, with the hypotheses that Web pages belonging to the same real person are more likely to be linked together. Three algorithms are presented for disambiguation; the first is just exploiting the link structure of Web pages and forming clusters based on link analysis, the second algorithm is based on word similarities between documents and does clustering using Agglomerative/Conglomerative Double Clustering (A/DC), the third approach combines link analysis with A/DC clustering. The work in [19] clusters documents based on the entity (person, organization, and location) names and can be applied to the disambiguation of semi-structured documents, such as Web pages. The primary new contribution is the development of a document generation model that explains, for a given document how entities of various types (other person names, locations, and organizations) are "sprinkled" onto the document.

3.1 Keyword Extraction Algorithm

Matsuo, Ishizuka [4] proposed a method called keyword extraction algorithm that applies to a single document without using a corpus. Frequent terms are extracted first, and then a set of co-occurrences between each term and the frequent terms, i.e., occurrences in the same sentences, are generated. Co-occurrence distribution showed the importance of a term in the document**.** However, this method only extracts a keyword from a document but not correlate any more documents using anchor texts-based co-occurrence frequency.

3.2 Transitive Translation Approach

Lu, Chien and Lee [5] proposed a transitive translation approach to find translation equivalents of query terms and constructing multilingual lexicons through the mining of web anchor texts and link structures. The translation equivalents of a query term can be extracted via its translation in an intermediate language. However this method

did not associate anchor texts using the definition of co-occurrences.

3.3 Feature Selection Method

Liu, Yu, Deng, Wang, Bian [6] proposed a novel feature selection method based on part-of-speech and word co-occurrence. According to the components of Chinese document text, they utilized the words' part-of-speech attributes to filter lots of meaningless terms. Then they defined and used co-occurrence words by their part-of-speech to select features. The results showed that their method can select better features and get a more pleasant clustering performance. However, this method does not use anchor texts-based co-occurrences on words.

3.4 Data Treatment Strategy

Figueiredo et al. [7] proposed a data treatment strategy to generate new discriminative features, called compound-features for the sake of text classification. These c-features are composed by terms that co-occur in documents without any restrictions on order or distance between terms within a document. This strategy precedes the classification task, in order to enhance documents with discriminative c-features. This method extracts only a keyword from a document but not correlate any more documents using anchor texts.

3.5 Alias Extraction Method

Bollegala, Matsuo, and Ishizuka [3] proposed a method to extract aliases from the web for a given personal name. They have used lexical pattern approach to extract candidate aliases. The incorrect aliases have been removed by page counts, anchor text co-occurrence frequency, and lexical pattern frequency. However, this method considered only the first order co-occurrences on aliases to rank them but did not focus on the second order co-occurrences to improve recall and achieve a substantial MRR for the web search engine.

4. APPROACH OVERVIEW

1http://www.clusty.com
2http://www.kartoo.com
3http://www.zoominfo.com/
4http://www.spock.com/

The main task of a WePS system is to accurately cluster the webpages in D = {d1, d2. . . dk}, such that each resulting cluster corresponds to a namesake. There are several possible ways of implementing a WePS engine, such as client-side, third party proxy, or server-side approaches. In a third party proxy approach, the user query is issued first to a proxy, which in turn quires a web search engine and then clusters the returned results. The advantage of such an approach is that a third (independent) party could implement it. A client-side solution is similar, except for the
software installed on the client acts as the proxy.

In a server-side approach, the user queried a web search engine directly. The server also does the clustering and returns the result to the user. The advantage of such an approach is that it is likely to be more efficient, as many webpage preprocessing steps can be done before any user query is issued. In addition, web querying is done internally instead of querying over the Internet. The disadvantage is that only a web search company could do that, or alternatively a third party should maintain a snapshot of the entire Web, e.g. using technology similar to that of Web Base developed at Stanford.

Preprocessing(AllWebPages,m)
1 for each webpage d 2 AllWebPages
2 P Extract-PeopleNE-Set(d)
3 P Clean-PeopleNE-Set (P)
4 Pd Pick-M-PeopleNEs (P, d,m)
5 O Extract-OrgNE-Set(d)
6 O Clean-OrgNE-Set (O)
7 Od Pick-M-OrgNEs(O, d,m)
8 TFd Precompute-TF/IDF-Factors(d,P,O)
9 Train-Classification-Skyline(TrainingData)

Figure 1: Webpage Preprocessing.

The solution proposed in this paper can potentially work with any of these architectures. However, currently it is more feasible as a server-side approach.

4.1 Preprocessing

The pseudo code demonstrates the webpage preprocessing steps carried out on the server. They are performed in advance for the entire web collection, before the system starts accepting user queries. For each webpage, its Named Entities (NEs) are extracted and processed. The TF/IDF factors are precomputed. This is done to speed up the future computations of TF/IDF similarities between pairs of webpages. TF/IDF will be computed during actual query processing and will use these precomputed values. Finally, the classifier is trained on the training data via supervised learning. The latter is a key step which will be covered. The pseudo code demonstrates that the extracted NEs are first cleaned. This is done using a set of filters. The idea is that NEs will be used by the framework as the context that identifies a namesake to some degree. However, certain types of NEs are too ambiguous for that purpose, in which case they are filtered out from further consideration. Given that the goal is to minimize the number of queries to the web search engine, the filters are specifically designed not to use any extra queries.

Rule 1. For instance, location names have been found to be too ambiguous to be used as context, as too many Namesakes may be mentioned in the context of the same location. Therefore, the algorithm does not use location information as one of the social network components. Moreover, NE

```
PROCESS-QUERY(Q, k)
 1  D ← GET-TOPK-WEBPAGES(Q, k)
 2  for i ← 1 to k − 1 do
 3     for j ← i + 1 to k do
 4        if AlreadyMerged(d_i, d_j) = true then
 5           break
 6        s ← COMPUTE-TF/IDF(TF_{d_i}, TF_{d_j})
 7        if s > τ then
 8           MergedInOneCluster(d_i, d_j)

    // All pairs are "unprocessed" initially.
 9  for each distinct unmerged unprocessed pair d_i, d_j ∈ D do
10     c_ij ← GET-WEB-COUNTS(d_i, d_j)
11     f_ij ← CONVERT-TO-FEATURES(c_ij)
12     if CLASSIFY-FEATURE(f_ij) = merge then
13        MergedInOneCluster(d_i, d_j)
14     MarkAsProcessed(d_i, d_j)

15  VISUALIZE-RESULTS-TO-USER(D)
```

Figure: Online User Query Processing.

extractors sometimes wrongly extract the location names as organization names. This also creates the same problem mentioned above. Thus, to eliminate the ambiguity, the preprocessing step filters out the locations extracted as organizations. It does so by performing a lookup in a locally stored gazetteer with the organization name as the query. If there is a matching location in the gazetteer, then the organization name is simply filtered out.

Rule 2. The second filter deals with the NEs that consist of one-word person names, such as “John”. Such NEs are highly ambiguous since they can appear in the context of many

namesakes on the Web. Consequently, the algorithm
prunes away the NEs consisting of one-word names. This filter works by performing a lookup into the dataset that store first names.

Rule 3. Similarly, the third filter handles NEs that are common English words. For example, word "defense"might be extracted from a webpage as an organization by the extraction software. However, it is a commonly used word, which can appear in the context of many namesakes. To detect common words the algorithm selects the most frequent 5000 terms from Wikipedia5 as common English words. If an NE is a common English word, it is filtered out.

Rule 4. Supposed that we are disambiguating web pages for "Jack Smith". It is not rare to find out that two or more distinct "Jack Smith" namesakes are related to two distinct namesakes"John Smith". These two Jacks might for example be relatives of the two Johns. Thus, "John Smith" cannot serve as a good context to identify a particular "Jack Smith" namesake. To capture this intuition, the algorithm filters out people NEs whose last name is the same as the last name specified in the original query.

4.2 User Query Processing

A user query processing consists of three logical steps illustrated in pseudo code in Figure 2: (1) TF/IDF Clustering (Lines 1–8), (2) WebFeature Clustering (Lines 9–14), and (3) Visualizing the results to the user (Lines 15). The first step of the algorithm merges all pairs of web pages whose TF/IDF similarities exceed a threshold. The threshold value is learned during the supervised learning process. TF/IDF is computed only on NEs extracted from the webpages, using the standard cosine similarity formula [22]. This initial clustering achieves two purposes: it decreases the number of queries to the search engine [17], while at the same time it improves the quality of the final clustering.

The second step employs the Web to gain additional data about the interactions between the webpages in D. For each distinct unprocessed and unmerged pair of webpages di, dj 2 D, it utilizes the search engine to collect the co occurrence information cij of the social networks of di and dj . This will be explained in more detail in Section 4.1. The algorithm then transforms the co-occurrences into the corresponding similarity features (Section 4.2). It then uses the
Skyline-based classifier on those features to predict whether the di, dj pair should be merged (Section 4.3). The algo rithm continues such iterations until no unprocessed pairs are left to be considered. After the second step, the final clustering is ready. The third step presents the computed final clustering results to the user.

GET-WEB-COUNTS(d_i, d_j)
Let:
N be the queried name
$\mathcal{P}_{i1}, \mathcal{P}_{i2}, \ldots, \mathcal{P}_{im}$ be the people NEs extracted from d_i
$\mathcal{P}_{j1}, \mathcal{P}_{j2}, \ldots, \mathcal{P}_{jm}$ be the people NEs extracted from d_j
$\mathcal{O}_{i1}, \mathcal{O}_{i2}, \ldots, \mathcal{O}_{im}$ be the org. NEs extracted from d_i
$\mathcal{O}_{j1}, \mathcal{O}_{j2}, \ldots, \mathcal{O}_{jm}$ be the org. NEs extracted from d_j

1 $\mathcal{P}_i \leftarrow (\mathcal{P}_{i1}$ OR $\mathcal{P}_{i2}$ OR $\cdots$ OR $\mathcal{P}_{im})$
2 $\mathcal{P}_j \leftarrow (\mathcal{P}_{j1}$ OR $\mathcal{P}_{j2}$ OR $\cdots$ OR $\mathcal{P}_{jm})$
3 $\mathcal{O}_i \leftarrow (\mathcal{O}_{i1}$ OR $\mathcal{O}_{i2}$ OR $\cdots$ OR $\mathcal{O}_{im})$
4 $\mathcal{O}_j \leftarrow (\mathcal{O}_{j1}$ OR $\mathcal{O}_{j2}$ OR $\cdots$ OR $\mathcal{O}_{jm})$

5 $c_{ij1} \leftarrow$ GetWebCount(N AND $\mathcal{P}_i$ AND $\mathcal{P}_j$)
6 $c_{ij2} \leftarrow$ GetWebCount($\mathcal{P}_i$ AND $\mathcal{P}_j$)
7 $c_{ij3} \leftarrow$ GetWebCount(N AND $\mathcal{P}_i$ AND $\mathcal{O}_j$)
8 $c_{ij4} \leftarrow$ GetWebCount($\mathcal{P}_i$ AND $\mathcal{O}_j$)
9 $c_{ij5} \leftarrow$ GetWebCount(N AND $\mathcal{O}_i$ AND $\mathcal{P}_j$)
10 $c_{ij6} \leftarrow$ GetWebCount($\mathcal{O}_i$ AND $\mathcal{P}_j$)
11 $c_{ij7} \leftarrow$ GetWebCount(N AND $\mathcal{O}_i$ AND $\mathcal{O}_j$)
12 $c_{ij8} \leftarrow$ GetWebCount($\mathcal{O}_i$ AND $\mathcal{O}_j$)

Figure : Algorithm for Querying the Web.

5. QUERY PROCESSING ALGORITHM

5.1 Queries to the Web Search Engine

To demonstrate that for each unprocessed pair of web pages di and dj the algorithm forms queries to the Web Search engine to compute co-occurrence statistics. This section explains the motivation behind using the queries as well as the procedure for forming such queries. The purpose of using Web queries is to evaluate the degree of interaction of the social networks for two namesakes represented by web pages di and dj . If there is evidence on the web that two social networks are closely related, then two web pages are merged into one cluster. The guiding principles in formulating the queries are:

• Quality. The queries should be chosen such that their results should allow creating a set of features that would enable high quality disambiguation.

• Efficiency. The overall number of the required queries should be minimized for efficiency reasons. The pseudo code in Figure 3 illustrates the procedure for formulating the queries. It demonstrates that two major types of queries are utilized:

1. N AND C_i AND C_j
2. C_i AND C_j.

Here, Ci represents the context for di. It can be either the set of people NEs Pi, or organization NEs Oi. Context Cj is defined similarly for document dj . Since Ci and Cj can have two possible assignments each, this creates 4 context combinations. Given that there are 2 types of queries, this leads to 8 queries in total. For example, assume that the user searches for the web pages related to "William Cohen". Suppose that the algorithm extracts 2 namesakes of each type per webpage, that is, m = 2. Assume that webpage di contains names "Jamie Callan"and"Tom Mitchell", and webpage dj contains names "Andrew McCallum"and"Andrew Ng". Then the first query will be:

"William Cohen" AND ("Jamie Callan" OR "Tom Mitchell") AND ("Andrew McCallum" OR "Andrew Ng").

The web search engine API has a function call that computes the number of web pages relevant to the query, without actually retrieving those web pages. Observe that dataset D alone might not have any evidence to merge di and dj . For instance, the TF/IDF similarity between di and dj might be low. Also, among the web pages in D, names "Jamie Callan" and "Tom Mitchell"might be only mentioned in di, whereas"Andrew McCallum" and"Andrew Ng"only in dj , and otherwise D might not contain any information revealing interactions among these people. However, querying theWeb allows the algorithm to gain additional information to support the merge. In this case, the counts will be high enough to indicate that the people mentioned in the query are closely related.

5.2 Creating Features

To estimate the degree of overlap of two contexts Ci and Cj for webpages di and dj for the queried name N we can compute the co-occurrence count |N · Ci · Cj | for query N · Ci · Cj . However, it might be difficult to interpret this absolute value without comparing it to certain other values. For instance, if this count value is high, does it mean the contexts overlap significantly and thus di and dj should be merged? Or, is it simply because N is a common name and thus there are lots of webpages that contains it under many contexts? Or, is it because contexts Ci and Cj are too unspecific, and thus too many webpages contain them?

Similar questions have been studied in the past [7]. The solution proposed there advocates normalizing such values, based on either Jaccard or Dice similarities. The Jaccard
similarity between two sets A and B computes the fraction of common elements in A and B among all the distinct ele ments

$$Jaccard(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

The Dice similarity between two sets A and B computes the fraction of common elements in A and B among all the
elements (including non distinct) in A and B, and normalizes it to [0, 1] interval:

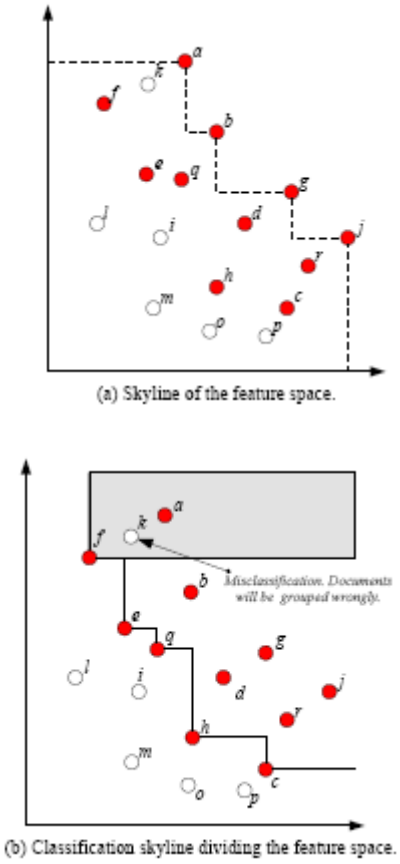

Figure: Example of a SkyLine.

$$Dice(A,B) = \frac{2|A \cap B|}{|A| + |B|}.$$

There have been studies showing that the differences in retrieval quality, when using these measures, is insignificant and furthermore these measures are monotone with respect to each other [18]. The proposed algorithm uses the Dice similarity to get the normalized version of |N · Ci · Cj | count. Two ways to normalize it have been examined:

$$Dice_1(N \cdot C_i, N \cdot C_j) = \frac{2|N \cdot C_i \cdot C_j|}{|N \cdot C_i| + |N \cdot C_j|},$$

And

$$Dice_2(N, C_i \cdot C_j) = \frac{2|N \cdot C_i \cdot C_j|}{|N| + |C_i \cdot C_j|}.$$

The algorithm employs the second formula, as it has proven to capture the ambiguity of context better. The following example provides just one type of scenario to illustrate the choice of the formula. Assume that the namesakes mentioned in two webpages di and dj are different. Suppose that the extractor wrongly extracts “This” as the only person NE from di, and “That” as the only person NE from dj . Assume that all (or, most of) the webpages of the two namesakes contain both “this” and “that”. Then, Dice1 similarity will be 1 (or, very high), causing the wrong merge of the webpages. However, Dice2 similarity will be low, since |Ci · Cj | will be large. That is, Dice2 will automatically capture that “this” and “that” is not a good choice to be used as the contexts.

5.3 Skyline-Based Classification

Observe that the features are chosen such that there is dominance in data in terms of merge decisions. First, let us make a few auxiliary definitions. We will say point f =(f1, f2, . . . , f8) up-dominates point g =

(g1, g2, . . . , g8), if f1 <_ g1, f2 <_ g2, . . . , f8 _<g8, and will denote it as f _ g. Similarly, we will say f down-dominates g, if f1 _ g1, f2 _g2. . . f8 <_ g8, and will denote it as f _ g. Similarly, there is a notion of strict domination where '_' and '_' are substituted with '<' and '>'. Assume that di, dj pair is characterized by the feature vector f = (f1, f2, . . . , f8). Suppose that based on f the algorithm decides that di and dj should be merged. Assume that there is another pair of web pages dk and d`, which is characterized by vector g = (g1, g2. . . g8). Then, if f _ g, then dk and d` should also be merged, since their social networks contain even more evidence that the two namesakes are the same person. Thus, there is dominance in feature data in terms of merge decisions.

```
GET-BESTQUAL-POINT(P_lsterr, S_clas, P_actv)
1  Q_best ← 0 // best quality observed
2  p_best // best point observed
3  for each p ∈ P_lsterr
4     S ← UPDATE-SKYLINE(S_clas, p)
5     Q ← QUAL(S, P_actv)
6     if Q > Q_best then
7        p_best ← p
8        Q_best ← Q
9  return p_best
```

Figure 7: GET-BESTQUAL-POINT.

```
UPDATE-SKYLINE(S_clas, p)
1  S ← S_clas
2  remove from S all points dominated by p // use indexing
3  S ← S ∪ {p}
4  return S
```

Figure 8: Update-Skyline.

6. Ontology

Ontology specifies linked concepts and terms and relations among these terms and concepts. Concepts are nothing but the entities which are language independent. Ontology shows how each concept is related what properties it has. The main purpose of ontology based decision tree is to give a more meaningful, descriptive and a readable view of concepts. Mathematically ontology can be defined Yang *et al*., 2008 [5] as follows:

"An ontology can be defined as an Vector O: = (C, V, P, H, ROOT), where C is the set of concepts, V contains a set of terms and is called the vocabulary, P is the set of properties for each concept, H is the hierarchy and ROOT is the topmost concept. Concepts are taxonomically related by the directed, acyclic, transitive, reflexive relation H belongs to C * C. H(c1, c2) shows that c1 is a subclass of c2 and for all c belongs to C it holds that H(c, ROOT)."

Our goal is to utilize the user context to the search results by re-ranking the results returned from the given query of search engine. The representation of the tree depends upon the user's information access behavior. Semantic information is fundamental part of user context. The best example of ontological approach has proven to be successful in the recommender system which does not consider the domain knowledge. Ontology consists of hierarchy of classes and sub-classes for object-entity [1]. The clusters will be taken as input which is formed using the lingo algorithm, later arranged in hierarchy. This could be achieved as follows:

- The topmost concept is the root having intermediate and leaf concepts. If a user wishes for a leaf concept then each cluster starting from the topmost concept will be traversed.
- The probability for each cluster will be calculated and depending on that clusters of user's interest will be accessed. Long with this a threshold value will be maintained.

• If the user heating ratio for a given cluster is greater and also the probability for each cluster is greater than the threshold value then parent is renamed by child name.
Mathematically it can be defined as follows:
Pb= (100*HR)/Total no.of clusters viewed
Where, Pb=Probability
HR=hitting ratio
An example of basic ontology is shown below in the Figure
Ontology explains a hierarchy of classes, sub classes and their relationships. The above example describes the "HIERARCHY" for a human class. It shows that a human is a vertebrate where the man and woman are synonym and they are hyponym for class human.

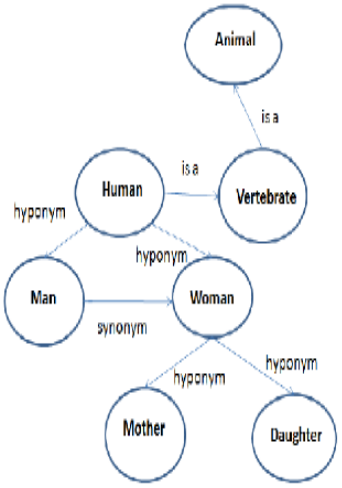

Figure: Example of Ontology

7. PROPOSED WORK

The system mainly works in different phases as:

7.1 Search result fetching:

The user submits the query to the search engine. The filters are used to find whether the given query is related to person search or not. Then we get the WebPages of search result lists returned by a Goggle web search engine. So the first search is the conventional met search based on these keywords. These WebPages are analyzed by an HTML parser and the result items are extracted.

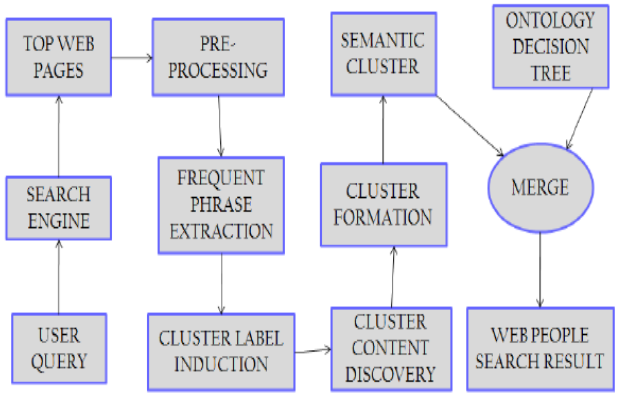

Figure: Proposed system

Generally, there are only titles and query-dependent snippets available in each result item. We assume these contents are informative enough because most search engines are well designed to facilitate users' relevance judgment only by the title and snippet, thus it is able to present the most relevant contents for a given query. Each extracted phrase is in fact the name of a candidate cluster, which corresponds to a set of documents that contain the phrase.

7.2 Ontology based Web People Search:

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. There are various algorithms such as K means, K-medoid, Suffix tree clustering, Hierarchical clustering. There are various drawbacks of these algorithms and to overcome these we use the Lingo algorithm. Lingo reverses the process—first attempt to ensure that we can create a human-perceivable cluster label and

only then assign documents to it. Specifically, extract frequent phrases from the input documents, hoping they are the most informative source of human-readable topic descriptions. Finally, match group descriptions with the extracted topics and assign relevant documents to them.
Our novel algorithm, Lingo clusters the search results. Unlike other algorithms lingo first discovers the name of the clusters and then evaluate each cluster with appropriate name. Basically lingo consists of five phases. The first phase deals with preprocessing to data. It detects the word boundaries and applies the stemming and stop word removal. The second phase deals with the frequent phrase extraction here a term appearing certain number of times is discovered and combined in all set of documents. Phase three is the cluster label induction. SVD is used to extract orthogonal vectors of the term document matrix, believed to represent distinct topic in the input data [6]. The fourth phase is the cluster content discovery phase. In this phase a Vector Space Model is applied to put the input documents under the cluster label discovered in the previous phase. Highest scoring documents for each cluster are assigned as that cluster's content [7]. Last phase is the i.e. the fifth phase is final cluster formation. Later, ontology based decision tree is referred for rearranging the clusters formed. The steps that will be followed are:

a. The ontology tree is formed according to the hitting ratio. Each object forms here the separate cluster.
b. After the cluster formation, weight is given to each node of the tree and threshold value is maintained.
c. Rearrange the clusters considering the weight given to each cluster and then display them to the users. Procedure that calculates the similarity between objects and clusters that estimate the similarity between clusters and ontology objects are used for this purpose.

d. If desired number of clusters are obtained, then stop else goto step a.

8. CONCLUSIONS

This section deals with experimental results when applied on web people search using ontology based decision tree. We have searched for the different persons with their relations like Abdul Kalam as the president of India, and listed the results obtained from various search engines The reslts obtained from our search engine where compared with other search engine likes Yippy, Wink People search engine, etc. Then these results were used to find the overall efficiency and accuracy of the search engines to give relevant results. We found that the accuracy for our search engine i.e. Web People Search Engine is good than other search engines.

The proposed method will compute anchor texts-based co-occurrences among the given personal name and aliases, and will create a word co-occurrence graph by making connections between nodes representing name and aliases in the graph based on their first order associations with each other. The graph mining algorithm to find out the hop distances between nodes will be used to identify the association orders between name and aliases. Ranking SVM will be used to rank the anchor texts according to the co-occurrence statistics in order to identify the anchor texts in the first order associations. The web search engine can expand the query on a personal name by tagging aliases in the order of their associations with name to retrieve all relevant results thereby improving recall and achieving a substantial MRR compared to that of previously proposed methods.

9. REFERENCES

[1] J. Artiles, J. Gonzalo , and F. Verdejo, " A Testbed for People Searching Strategies in the WWW," Proc. SIGIR '05, pp. 569-570, 2005.

[2] R. Guha and A. Garg, " Disambiguating People in Search," technical report, Stanford Univ., 2004.

[3] D.Bollegala, Y. Matsuo, and M. Ishizuka , "Automatic Discovery of Personal Name Aliases from the Web," IEEE Transactions on Knowledge and Data Engineering, vol. 23, No. 6, June 2011.

[4] Y. Matsuo, and M. Ishizuka," Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information", International Journal on Artificial Intelligence Tools, 2004.

[5] W. Lu, L. Chien and H. Lee, "Anchor Text Mining for Translation of Web Queries: A Transitive Translation Approach," ACM Transactions on Information Systems, Vol. 22, No. 2, Aprill 2004, Pages 242-269.

[6] Z. Liu, W. Yu, Y. Deng, Y. Wang, and Z. Bian," A Feature selection Method for Document Clustering based on Part-of-Speech and Word Co-occurrence," Proceedings of 7th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 10), pp. 2331-2334, Aug 2010.

[7] F. Figueiredo, L. Rocha, T. Couto, T. Salles, M.A. Gonclaves, and W. Meira Jr, "Word Co-occurrence Features for Text Classification", Vol 36, Issues 5, Pages 843-858, July 2011.

[8] G. Salton and C. Buckley," Term-Weighting Approaches in Automatic Text Retrieval," Information processing and Management, vol. 24, pp. 513-523, 1988.

[9] T. Dunning, "Accurate Methods for the Statistics of Surprise and Coincidence," Computational Linguistics, vol. 19, pp. 61-74, 1993.

[10] K. Church and P. Hanks, "Word Association Norms, Mutual Information and Lexicography," Computational Linguistics, Vol. 16, pp. 22-29, 1991.

[11] T. Hisamitsu and Y. Niwa, " Topic-Word Selection Based on Combinatorial Probability," Proc. Natural Language Processing Pacific-Rim Symp. (NLPRS '01), pp.289-296, 2001.

[12] F.Smadja, "Retrieveing Collocations from Text: Xtract," Computational Liguistics, Vol. 19, no 1, pp. 143-177, 1993.

V.Redya received the B.E, degrees in computer science and engineering from Osmania University, Hyderabad, AP, and India. He is received M.Tech in Computer Science and Engineering at Jawaharlal Nehru Technological University, Hyderabad, AP and India. He is pursuing Ph.D program in computer science and Engineering.

M.NageswarRao received the Msc, Master degrees in Mathematics from Andhra University, AP, and India. He is received M.Tech in Computer Science and Engineering at Bhrathi University, Chennai and India.

M.LaxmanKumar received the B.Tech, degrees in computer science and engineering from Jawaharlal Nehru Technological University, Hyderabad, AP, and India. He is Pursuing M.Tech in Computer Science and Engineering at BOMMA INSTITUTE OF TECHNOLOGY and SCIENCE from Jawaharlal Nehru Technological University, Hyderabad, AP and India.