Clustering Search Results

Prashanth Reddy Madi

(prm0080@unt.edu)

Abstract:

Web creates new challenges as amount of information on the web is growing rapidly, as well as the number of new users inexperienced in the art of web research. People are likely to surf the web using its link graph, often starting with high quality human maintained indices such as search engine. Many of times, Each output link from search engine depicts different topic, Ex: Apple (company, fruit). In order to direct user with one of the topic, Showing User's Similar pages (Requires Clustering) at regular interval's for their query (reformulated based on cluster in use) may reduce this effect. Common method of clustering includes BOW(bag of words) model. These are based on co-occurrence statistics which ignores semantic relatedness. Usage of semantic relatedness involve computational complexity. With growing web pages, It's almost became practically impossible with limited resources. In this paper, we present a naive implementation of Clustering method. I had also eliminated totally similar pages, as they represent same information.

Introduction:

Information overload is a popular problem today. This problem could be solved partially with Search Engine: a tool helps find needed information from the whole web. However, even though some Search Engines work very well, users still cannot avoid information overload problem. There are so many returned results. Post processing search result is a step to further reduce information overload problem by organizing search results such that minimizing effort for examining them. One of the common ways to post processing search results is clustering. In this project, we propose a novel technique for clustering search results which does it on the fly. Organizing Web search results into clusters facilitates users' quick browsing through search results. Traditional clustering techniques are inadequate since they don't generate clusters with rank for representing them. In this paper, we re-formalize clustering problem as a salient cluster ranking problem.

Automatic query expansion is used to add related terms to the user’s query. In the field of IR, Expansion techniques based on statistically derived associations have proven useful, while other methods using thesauri with synonyms obtained less promising results . This is due to the ambiguity of query terms and its propagation to their synonyms. The resolution of term ambiguity (Word Sense Disambiguation) is still an open problem in Natural Language Processing. Nevertheless, in the case of geographical terms, the resolution of ambiguity is usually less difficult and therefore better results can be obtained by use of effective query expansion techniques based on Ontologies. I had used meronym, holonymn and synsets in wordnet to expand the query during pre-processing step. I had worked on address resolution of geographical terms in query. As, there should be difference between what are the places in texas and where is denton, Texas ? First, should result all the places in Texas. which can be retrieved by using meronym in Wordnet, While the later should be limited to use meronym. As the query would expand to where is denton, Texas followed by all other places returned by meronym for Texas. My method would return results from meronym based on existence of its results in Query.

Clustering of document involves grouping similar type of documents based on some criteria. In the methodology presented in [2] I had used semantic relatedness . The benefit of using such concept is to group documents in the same cluster even if they do not contain common keywords, but still possess the same sense. Existing clustering techniques cannot perform this sort of discovery or do this work only to a limited degree. Developing algorithm that discover all frequently occurring subgraphs in a large graph database is particularly challenging. I had used dbscan to discover the subgraphs. After discovering the subgraphs the documents are clustered using kmeans algorithm.

Related Work:

Many works has been done in the problem of clustering search results. Most of them use traditional clustering algorithms as Hierarchical algorithms (e.g. agglomerative or divisive algorithms) and Partitional algorithms. Similar work was done by Hieu Khac Le[1], They had used Induction Clustering, where each cluster is formed based on a summary from the result but still this doesn't solve the problem of ranking clustered documents. In another Research, Prashanth[2], had performed clustering using semantic relatedness. In this method, I had used Wikipedia miner.

During the pre-processing step, Usage of wordnet for query expansion was performed by David and Bariero[7]. As there would be ambiguity of using word-net due to presence of more than one meaning Davide, Paolo and emilo [6] had confined usage of word-net only for geographical terms.

Method:

I had collected 3000 web pages from UNT domain using a web crawler. Than, I had applied pre-processing steps such as

1. Html tag removal
2. Tokenization
3. Stop-word removal and
4. Stemming.

I had applied tfidf on each of the document after pre-processing step. A typical weighting is *tf-idf weighting* :

$$w_{ij} = tf_{ij}\, idf_i = tf_{ij} \log_2 (N/ df_i)$$

A term occurring frequently in the document but rarely in the rest of the collection is given high weight. Experimentally, *tf-idf* has been found to work well

In order to compensate distance calculation (square root the sum of square's of each word's weight in document) during run-time for cosine similarity, I had stored them in a static file while calculating tfidf. I had created inverted index for each existing unique word in corpus. This will reduce time of execution, As we will apply cosine similarity on those documents which have query words.

For an input query, i had used wordnet to incorporate related terms for each word in query. As mentioned in introduction part of the paper, I had induced rules on usage of meronyms for address resolution. Then, i had applied pre-processing steps as mentioned earlier for documents.

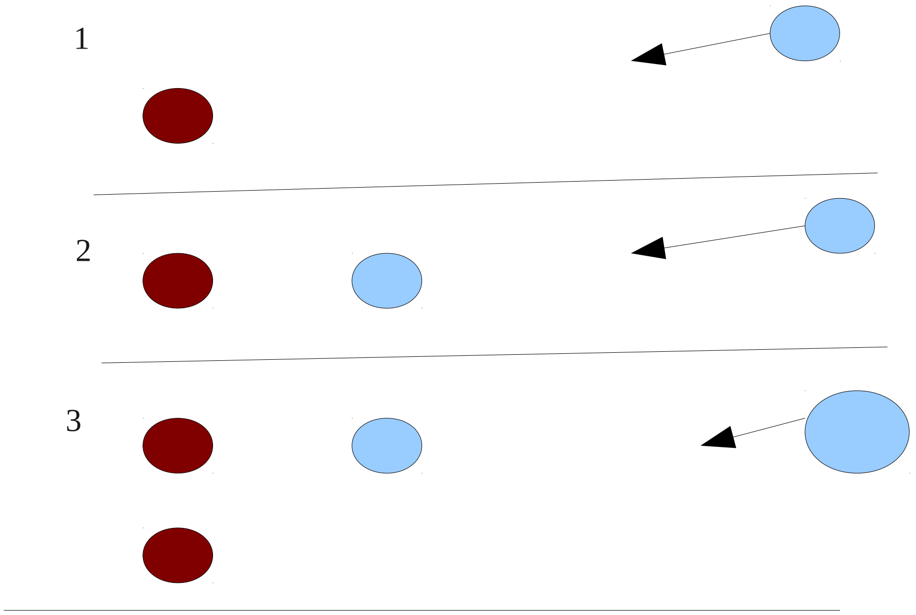
To measure the similarity between query and documents, i had used cosine similarity. Here query is considered as a document and represented as Vector. Similarity between vectors d_1 and d_2 is *captured* by the cosine of the angle x between them.

$$sim(d_j, d_k) = \frac{\vec{d}_j \cdot \vec{d}_k}{|\vec{d}_j||\vec{d}_k|} = \frac{\sum_{i=1}^{n} w_{i,j} w_{i,k}}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2}\sqrt{\sum_{i=1}^{n} w_{i,k}^2}}$$

The denominator involves the lengths of the vectors, So the cosine measure is also known as the *normalized inner product*

$$\text{Length } |\vec{d}_j| = \sqrt{\sum_{i=1}^{n} w_{i,j}^2}$$

For each of the query after applying cosine similarity, i had got ranked set of documents. As a further step for intelligent search, I had grouped the ranked result documents with a novel clustering method. In this, first ranked document is considered as representative and compared with the second. If the dis-similarity between these documents is less than a particular threshold than they are grouped into one cluster, else the second document would represent another group. If two different clusters are formed and the incoming doesn't matches with first cluster representative than it is compared with the second cluster representative, If it still doesn't matches than it becomes another representative for third cluster. This greedy approach is continued untill, all the documents in the retrived result gets clustered. This method is pictorially represented as follows.



Each incoming document is compared only with the representative of the group, represented with red colour in above figure. We find most of the web pages representing same content including the structure or few weblinks redirected to same webpage. In order to resolve this, I had eliminated pages with similarity less than 0.05.

During Clustering process, we need to check for similarity between pair of documents. Before the model i had used for this, I went through following approach

CUT- 1

I had calculated similarity of every single document with remaining. Than, Placed this in a static file. Here the similartiy between doc1 vs doc2 and doc2 vs doc1 are same. So, I had considered only smaller document number vs larger document and i had load this file during clustering process.

CUT-2

Since, I am calculating similarity for only retrieved documents from search results. I felt that there is no point in loading similarity of all the documents. So, i had kept single file for each and every document to represent their similarity with other. Example doc1 has similartiy of doc1 vs all other. This would reduce time complexity.

CUT-3

Even though, previous method had decreased time complexity but still, due to limited memory resources i was forced to search for alternative. Than, at last I got settled with calculating similarity between search results on the fly during clustering stage. This had reduced memory and time complexity.

I had also performed clustering of documents by considering semantic relatedness measure using wikipedia miner. But, the time required for clustering these documents was too long and it became practically impossible for an online demo.

Evaluation:

I had applied few set of queries on the search engine to get its result and also clustered them using above model and cross-checked them manually by using the links provided by search result

Results:

After cautious analysis of clustered search results, I found that most of the links belonging to a particular cluster are well arranged and takes fraction of seconds to output results. Our experimental evaluation show that this approach leads to reasonably good clusters.

Conclusion and Future Work:

- Using combination of single and complete linkage clustering algorithm's to cluster large data
- Prof. Richard Goodrum had an idea of using message passing concept in c++ to perform clustering using multiple process.
- Incorporate user's (like/dislike) based on Measuring proximity on graphs with side information
- Confine Query expansion of wordnet only to geographical terms.

References:

1. Inductive Clustering: A Technique for Clustering Search Results

   Hieu Khac Le
2. Clustering of documents using wikipedia miner

   Prashanth Reddy Madi
3. eSReC: A News meta-Search Engines Result Clustering Tool

   Hassan Sayyadi, Sara Salehi, Hassan AbolHassani.
4. Learning to Cluster Web Search Results

   Hua-Jun Zeng, Qi-Cai He, Zheng Chen, Wei-Ying Ma, Jinwen Ma.
5. Personalized Concept-Based Clustering of Search Engine Queries

   Kenneth Wai-Ting Leung, Wilfred Ng, and Dik Lun Lee
6. A WordNet-based Query Expansion method for Geographical Information Retrieval

   Davide Buscaldi, Paolo Rosso, Emilio Sanchis Arnal
7. QUERY EXPANSION USING WORDNET WITH A LOGICAL MODEL OF INFORMATION RETRIEVAL

   David Parapar, Álvaro Barreiro