Multi-factor Clustering for a Marketplace Search Interface

Neel Sundaresan
eBay Research Labs
2145 Hamilton Avenue
San Jose, CA 95125
(408)376-8422
nsundaresan@ebay.com

Kavita Ganesan
eBay Research Labs
2145 Hamilton Avenue
San Jose, CA 95125
(408)967-5895
kaganesan@ebay.com

Roopnath Grandhi
eBay Research Labs
2145 Hamilton Avenue
San Jose, CA 95125
(408)376-8439
rgrandhi@ebay.com

ABSTRACT

Search engines provide a small window to the vast repository of data they index and against which they search. They try their best to return the documents that are of relevance to the user but often a large number of results may be returned. Users struggle to manage this vast result set looking for the items of interest. Clustering search results is one way of alleviating this navigational pain. In this paper we describe a clustering system that enables clustering search results in an online marketplace search system.

Categories and Subject Descriptors

H.3.3 [Information Search and Retrieval]: clustering, query formulation, relevance feedback, retrieval models, search process, selection process

General Terms: Algorithms, Performance, xperimentation

Keywords: Algorithms, Clustering, Suffix-Tree, Linear

1. MOTIVATION AND BACKGROUND

The challenge of providing a manageable search experience exists whether the search is over a document repository within a site, or over the World Wide Web. Often users get overwhelmed with the fact that a large number of documents matched their search query and that in turn increases the user drop-off rate. Search engines have used various techniques for helping the user manage the search results. Query refinement, faceted navigation, clustering, and search refinement are a few techniques. Unlike query refinement, faceted navigation or search refinement, clustering tries to group together large number of query results and let the user dive deeper into the cluster of interest. Clustering has the advantage that it does not require the system to dip back into search index as query filtering or faceted navigation do.

Our goal is to provide a search experience based on clustering in an online marketplace. The constraints on the clustering system are as follows: it is real-time; the pieces of information available to the clustering system are a single line of title for each item for the search results based on the query; it should be scalable and incremental and possibly linear in nature; clusters could be overlapping; number of clusters and cluster sizes may be dynamic. Factors like price, category of the item may be available with some confidence. In this paper we discuss the design and implementation of a clustering system that caters to this requirement.

The rest of the paper is organized as follows: In section 2 we describe the challenges of search in a marketplace environment. In section 3 we discuss a linear multi-factor clustering system using STC. Section 4 discusses architecture and implementation. In section 5 we introduce evaluation measures and discuss experimental results. We conclude in section 6 and draw scope for further research.

1. THE SEARCH CHALLENGES

eBay has one of the best known community created content for commerce. This has its own challenges. eBay sellers, while expected to accurately represent their ware for sale, have limited real estate in the title to describe their items at best. Also, sellers use the limited title space for merchandizing. For example, an ‘iPod skin’ item might mostly look like the title for an iPod with the term ‘skin’ added. An IR-based search might return both iPods and iPod skins. While buyers searching on the keyword ‘iPod’ might be looking for mostly iPods, it is not appropriate to filter out iPod skins. Majority of items in eBay are not catalog items and cannot be cataloged as they tend to be one-of-a-kind items. Further, attributes and values cannot be well defined for such items. While items can be de-emphasized based upon knowledge of buyer intent, they cannot be ruled out. In this context, we believe that clustering might provide a useful intermediate solution.

2. MULTI-FACTOR CLUSTERING

We want a linear clustering algorithm where, as documents are incrementally added, massive re-clustering is not required. Our interest in Suffix Tree based clustering algorithm [1] is driven by these goals. We also need a mechanism to identify and use phrases and not mere words in the titles. Suffix tree retains the order of occurrence of terms in the documents which helps in identifying phrases. The use of STC for search result clustering has been demonstrated in Grouper [3].

Clustering search results with the standard STC algorithm does not guarantee the quality of clusters formed. There are other influencing factors that may affect the quality of the clusters. This is especially true for eBay’s search results where each item listing is made up of different components. For example, one factor could be the relevance of the terms in the title to the query. Other factors include the item price, seller information and possibly feedback, categories into which the items belong and so on. If such parameters can be successfully used to influence the cohesiveness of clusters, we will be able to obtain more meaningful clusters.

3. ARCHITECTURE

Our prototype implementation evolves around adding layers to the core STC clustering implementation. The system was built using components that have been made to work together in generating good search based clusters with meaningful labels.

WWW 2007, May 8–12, 2007, Banff, Alberta, Canada.
ACM 978-1-59593-654-7/07/0005.

For a given query, the flow starts with obtaining search results for the particular query. This is achieved through the externally available eBay search API. The relevance weights obtained (upon selection) are normalized for every query. The search results obtained are then clustered on the fly using the standard STC clustering algorithm with influencing factors like the relevance-weighting factor (weighting of the terms according to how relevant it is to the query based on relevance feedback), merge threshold and minimum base cluster score, making this a multi-factor clustering system. The clustered results are then sent to the labeling algorithm for representative label extraction. The labeling algorithm uses a statistical measure to provide descriptive tags that represent the clusters as a whole. Figure 1 is a screenshot of the results interface for the search query "persian rugs".

3.1 *Cluster Label Extraction*

Cluster labels depict compact information about the type of documents present in the cluster. The labels act as pivots for navigation for the user and hence should represent the concepts within the cluster as closely as possible.

We first labeled the clusters using the most frequent phrases in the cluster but the labels formed just using this measure do not accurately represent the documents and would result in more noise phrases. We have investigated two approaches in labeling the clusters, $\mathrm{CLE}^{\mathrm{Tags}}$ and $\mathrm{CLE}^{\mathrm{BD}}$. In $\mathrm{CLE}^{\mathrm{Tags}}$ we extract important multiple tags within the cluster while in the $\mathrm{CLE}^{\mathrm{BD}}$ we pick the best document (centroid) that represents the majority of the documents within the cluster.

4. EXPERIMENTS AND EVALUATION

We took the typical 1000 queries in eBay as typical queries for our experiment. While the ideal measure of correctness would be based on actual user experience we wanted to create some simple automatic evaluation metrics. We used two measures: *Coverage* and *Overlap* (Cluster Independence) to evaluate our system.

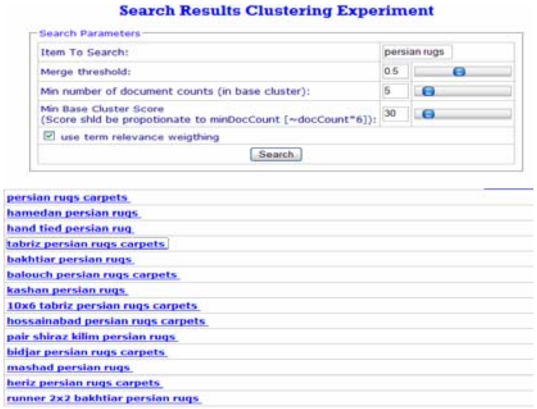
Coverage could be measured using the ratio of all those result items that belong to at least one cluster of size > 1 to the total number of items in the result set. As for overlap, if we can separate out clusters by their distinction from others, we can say that the Overlap (Cluster Independence) measure is high. One easy way to achieve high cluster independence is by putting each item in a separate cluster or by creating random buckets of clusters each of which has independent sets of items.

4.1 Results: Coverage and Overlap

Figures 2 and 3 depict graphs plotted with top 1000 queries against the coverage and overlap values with and without the use of the relevance weighting.

5. CONCLUSION

Here we described the architecture of such an STC based clustering system that clusters in real time. We devised methods for tagging the clusters. We formalized a number of measures like coverage and overlap for measuring the performance of the algorithm and also measured the influence of the relevance factor on the coverage and cluster overlap. Our cluster overlap is a simplification of a precision measure. While overlap and coverage move in opposite directions, we need to further evaluate the trend. We need to further refine this measure to better approximate it to precision and then we can derive the F-measure as a weighted harmonic mean of these two measures. We also need to do further analysis on our cluster quality measures and category distribution measures.


Search Results Clustering Experiment

Search Parameters

Item To Search: persian rugs

Merge threshold: 0.5

Min number of document counts (in base cluster): 5

Min Base Cluster Score
(Score shld be propotionate to minDocCount [~docCount*6]): 30

☑ use term relevance weigthing

persian rugs carpets
hamedan persian rugs
hand tied persian rug
tabriz persian rugs carpets
bakhtiar persian rugs
balouch persian rugs carpets
kashan persian rugs
10x6 tabriz persian rugs carpets
hossainabad persian rugs carpets
pair shiraz kilim persian rugs
bidjar persian rugs carpets
mashad persian rugs
heriz persian rugs carpets
runner 2x2 bakhtiar persian rugs

Figure 1. Screen shot of the Clustering system interface. Merge threshold, Min number of documents, and base cluster score are fed as arguments in to the clustering algorithm. The results shown display the different clusters.

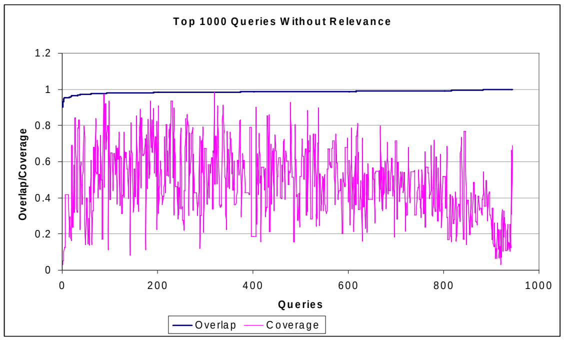


Figure 2. This shows the coverage and overlap measures for the top 1000 queries without relevance query. The algorithm performs well in the overlap measure. Coverage averages to about 50%. This number indicates on an average given a result set about half of them could be clustered into clusters of size > 1.

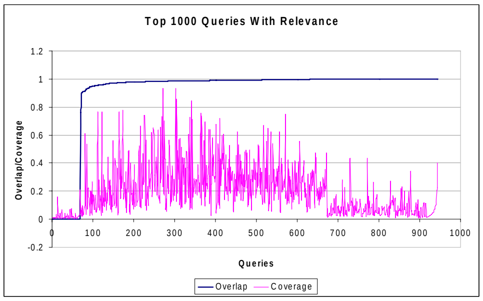


Figure 3. This shows the coverage and overlap measures for the top 1000 queries with relevance query. The algorithm performs well in the overlap measure. Coverage drops as compared to the cases without relevance measure.

6. REFERENCES

[1] S. Kurtz. Reducing the space requirements of suffix trees. Software – Practice and Experience. 29(13), 1149-1171, 1999.

[2] O. Zamir and O. Etzioni. Grouper: A dynamic clustering interface to web search results. Proceeedings of the 8th World Wide Web Conference, Toronto, Canada.

[3] D. Gusfield. Algorithms on strings, trees, and sequences: computer science and computational biology, chap 6. Cambridge University Press, 1997.