Problem Statement

Given a list of queries to a search engine, we aim to cluster them into groups such that those with common intent end up in the same cluster and hence, constitute a search session. Note that all queries of a search session need not be continuous - a session can return back after user does some queries about another session.

For example, if we have the following searches:

1. new home furniture
2. best sofa 3 seater
3. glass dining table
4. latest gaming laptop
5. Counter strike tactics
6. call of duty tricks
7. Where can I find cheap wardrobe?
8. Narnia movie online

Here, queries 1-3 and 7 are broadly with similar intent (buying furniture) and queries 4-6 are clearly not related to this, but are about Gaming. Query 8 needs a new cluster of its own.

Motivation

Once the search queries get segmented into search sessions, we believe that other information such as click-through data can be used for several useful purposes - personalization, identifying type of the query (navigational, informational etc). The key idea is that if we have the information about what the user did on the search queries which are most similar to the current one, we are in a better position to generate relevant search results.
Proposed Solution

Dataset(s) used

For the purposes of this project, we do not need any search engine logs or click-through data. However, for every search query, we need an (anonymized) identity of the person who ran the query.

There are very few public datasets available that contain search query logs, mainly to protect privacy. However, there have been few (controversial) releases of such datasets on public forums, the most popular one by AOL in 2006, which we use for this project.

Approach

We realized that the proposed clustering of search queries into sessions would have to be performed at a per-user level. We do some basic preprocessing with all search queries such as removing stop-words and punctuations.

Given a stream of queries from a user, we maintain a set of search 'sessions' and an eligibility factor for each session. This eligibility factor is a measure of how recently the session was updated and therefore, ages (or decays) with time. Whenever a new query is added to a session, its eligibility is reset to a maximum value. If the eligibility factor of a session goes below a certain threshold, the corresponding session becomes inactive and is not considered for clustering in the next step. Each session itself is a bag of words containing all words of the queries belonging to that session.

To allocate a cluster to a new query, we iterate over all words in the current query. For each word, we keep track of the session that is 'closest' to. Finally, we put the query in the cluster which has overlap over the largest number of words, tied are broken by the eligibility factor described above. In case less than half of the query words match any current session, we create a new session for the query (more recently updated session is selected).

The closeness of a word from the bag of words of every cluster is calculated based on edit distance. Specifically, we iterate over the list of words belonging to a session and find the one within reasonable (in this case, <= 3) edit distance from the query word. This is one of the solutions for the presence of spelling errors in the search query.
Challenges

Availability of dataset

Yet another dataset that can be created for this purpose using the IIT Bombay network log data.

With help from the CC, we procured the logs for the last two months. While extraction of individual search queries from these logs hasn’t been done, it also looks like the total number of search queries going through the IITB network is fairly small in number. We are taking this up with the CC to confirm the same and find a solution.

Distance metric for short words

In some cases the words in a query are very short. For example, the edit distance between ‘cars’ and ‘card’ is 2, which would be below our threshold. This makes it necessary to find out a distance metric which can adequately assign scores between queries and clusters so that only the keywords are considered when evaluating closeness to a cluster/session.
Results

A quick implementation of the above algorithm on search queries of user #39509 gave the following results (cluster numbers of each query indicated in square brackets).

1. [1] cardomain
2. [1] cardomain .com
3. [2] soundomain .com
7. [3] mtx sledghammer subs
8. [3] cheap mtx sledgehammer
10. [1] cardomain
11. [1] cardomain .com
12. [1] cardomain
15. [6] ps2 game cheats
17. [7] original body panit colors for a ford fairlane
18. [7] original body paint colors for a ford fairlane
19. [7] original ford body paint colors
20. [8] lmc
21. [7] ford body paint colors
22. [7] ford auto paint colors
23. [7] fairlane paint colors
24. [9] pagoda green
25. [9] dynasty green
26. [9] dynasty green ford
27. [7] 1964 ford auto paint

Knowing the complete picture (what cars, woofers, ford, etc are), we can probably conclude that in the snippet of consecutive search queries shown above, the user is interested in cars, looking for audio systems, 1964 ford fairlane paint colours, etc. Some interesting observations:

- Consider cluster #7. We observe that as more and more queries get added to this cluster, a clearer picture forms regarding the intent of the user. We understand that probably the user was looking for 'original body paint colours of a 1964 ford fairlane'. This complete information was available in none of the actual queries issued.
- There are instances when old search sessions become active again (Queries 10-12 activating cluster #1 and query 28 activating cluster #6).

We also observe that the clustering is not ‘perfect’. Queries 16 and 24-26 should ideally have been a part of the same cluster as queries 17-19, 21-23, 27 and 29.

Much deeper insights can be drawn about user intent using other metadata for such queries such as click dwell time, time between consecutive queries, time between the first and the last clicks, etc.
What more can be done if time permits

Better distance metric

We have implemented a very simplistic approach to the problem for determining the similarity of a query to a cluster - word edit distance. We can use better metrics to take care of multiple words describing the same idea (car sales and auto sales). This can be achieved by way of several techniques - word2vec embeddings from Wikipedia, combining meta-data information of the website clicked-through.

Disambiguation & usage of slang

Queries often contain ambiguity and slang. These form the primary challenge to this task. Disambiguation also poses a significant challenge. A google search for ‘apple’ does not give results about the fruit Apple till the third page! Even if you have been searching about other fruits! While some amount of disambiguation information can be used from sources such as Wikipedia, a more realistic approach would be to show diverse results and aggregate click data.

Similarity based on semantic meaning

In the current approach, the meaning of the words is not considered and so, search terms with similar intentions might not get mapped to the same cluster. For instance, consider queries ‘home loan’ and ‘credit score’. We generally know that both these terms are related and often your credit score forms an integral part of the home loan application. The challenge is making the computer understand that these two search terms are related and thus need to be clusterized into the same session.