Inductive Clustering: A Technique for Clustering Search Results

Hieu Khac Le
hieule2@uiuc.edu
Department of Computer Science
University of Illinois at Urbana-Champaign

Abstract

In this paper, we introduce a novel technique for clustering search results: Inductive Clustering. Interestingly, our approach avoids three essential components which exists in all of traditional clustering techniques: a similarity function, and a threshold of similarity or a predefined number of clusters. The experiment shows that Inductive Clustering work extremely well and really helps users to capture the return search results much more easily.

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 the information overload problem by organizing search results such that minimizing the 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 result: Inductive Clustering (IC). Following is the figure show how search results are clustered with IC in our illustration system.

![Figure 1: An example of clustered results](image-url)
The remained parts of this paper are organized as following: (1) Related works: we discuss about recent works which have the same or similar goal to us; (2) Three common components in traditional clustering: in this section, we describe the common characteristics of current clustering techniques and their limitation; (3) Inductive Clustering: we introduce our clustering technique; (4) Experiment: we describe experiment results show how our technique works in term of different measurements; (5) Discussion: in this part, we discuss and explain the experiment result, we also envision future work and room to improve our technique; And as normal, the last section is (6) Conclusion.

Related works

There are many works has been done in the problem of clustering searches results. Most of them (e.g. [2][4][5]) use traditional clustering algorithms as Hierarchical algorithms (e.g. agglomerative or divisive algorithms) and Partitional algorithms (e.g. K-means or Fuzzy C-means algorithms). After results were grouped into different clusters, descriptions for each cluster are generated from inside results. Usually, descriptions are a set of words from the results as in [1] and all techniques surveyed in [4]. Some of techniques was developed as an online service as: Vivisimo.com or Grouper (retired in 2000).

Three common components in traditional clustering

There are there common components in technique using traditional clustering algorithms: a similarity function, a threshold of similarity or a predefined number of clusters.

Both of hierarchical clustering algorithms and partitional algorithms need a similarity function. Similarity functions are formed by taking into account different factors as edit distance, link structure, and other attributes. Linear regression is a common technique used to combine those factors as a unique similarity function. Finding the weights vector for linear regression is often done via manually tuning or applying machine learning technique [1].

Hierarchical clustering algorithms need a threshold to determine whether two search results are grouped in a same cluster or not. The higher the threshold is, the further the search results are clustered in detail. Unlike Hierarchical clustering algorithms, Partitional clustering algorithms do not need a threshold to determine how granularly search results are clustered; instead, they need a prior given number of clusters.

Those components heavily affect clustering quality. Unfortunately, there is no guidance to tune or estimate them, especially with threshold and number of clusters.

Our technique: IC, in opposite, do not require any of those components to be able to form the search results.

Inductive Clustering

Inductive Clustering bases on following observation: (1) Users’ queries could be considered as summary of returned search results; (2) The more specific the query is, the fewer results are returned.

From above observation, we could summarize our method as following: From the returned results, generate a summary. Results agree with that summary will be the first cluster. Generate a summary for the remained results; results agree with that summary will be the second cluster. Do the same process until all results are clustered. Large clusters could be clustered further in the same way. Following picture illustrate the clustering process in general:
Suppose that a user wants to query all objects having blue color. Returned results including different blue objects with different shapes. First, a summary will be generated from the whole returned results; in this case we have square shape as a summary. Next, the summary square will be used as a query on results set and we get as subset of blue square object, this set forms the first cluster. All remained results which do not agree with the summary will be continually processed in the same way. The whole process finishes when all remained results agree with the generated summary or when the size of further cluster is too small (in this case, all remained results will be group as a single cluster with no specific title). In the following subsections, we will discuss in details each step of the technique.

**Preprocessing search results**

In this step, all returned results are converted to vectors of terms. All HTML tags are also removed.

**Generating summary**

Different document summarizing techniques could be applied in this step. The longer the summary is, the more strictly filter will be applied on the returned results. In opposite, if the summary is too short, it will be less meaningful while presented to the users. Therefore, we have to keep the summary long enough to be meaningful and short enough to be not over-fitting with minor number of results. To solve this two-fold difficulty, we use two different forms of summary: *internal summary* and *external summary*.

An *internal summary* is a set of important terms used to filter remained search results to form a new cluster. The number of terms in the summary depends on two factors: the scale of result population and the ability to measure semantic distance among terms in the domain if given one. As a choice of implementation, since we just invoke clustering on fairly small size of top results and the queries are not domain specific, we decide internal summary will be the most important term based on TF-IDF score. This term must be not included in the user’s query.

An *external summary* is a meaningful phrase extracted from results of a cluster and will be presented to users. Normally, an external summary will be a noun phrase, verb phrase, adjective phrase, or a proper name. An additional task is needed to detect boundaries of phrase in results. From the internal summary, the most common phrase containing all terms in internal summary in the result of a cluster will be selected as the external summary.
Filtering results to form clusters

After having internal summary, all un-clustered search results will be checked with that internal summary. Results agree with the internal summary will be putted as a new cluster with the title are the corresponding external summary. The results do not agree with the current internal summary will be kept for further clustering process. A result is considered as agreeing with an internal summary if it contains all terms in internal summary.

Ordering results inside clusters

There are two alternative options for ordering search results in clusters. The first one is that the results should be ordered according to the ranked score got by search engine. The other option is that the results will be re-ordered so that similar results appear close together. In the illustration system, we implement both choice and leave them as customizable options for users.

Ordering clusters

Clusters could be sorted in many different ways, in this work we propose three possible options: (1) clusters are sorted basing on the order in which they are created via IC; (2) clusters are sorted basing on the rank of the highest ranked result inside each clusters; (3) clusters are sorted basing on their population: the more results in the cluster, the higher it is ranked.

Confidence of clusters

The higher score the terms in the internal summary get, the more confident the related cluster is. As the way clusters are formed, clusters which are created sooner have higher confidence than the followed ones. In case clusters are not ordered according to option (1) in previous subsection, clusters are more confident than half of all clusters will be presented with different color (as the figure on the right).

Pseudo code for Inductive Clustering

Following figure is pseudo code for inductive clustering algorithms.
```java
1 Result[] results;
2 results = SearchEngine.search(query);
3 clusters = new Cluster[];
4 Result[] remainedResults = results;
5 while(remainedResults.count != 0)
6 {
7     string internalSummary = getInternalSummary(remainedResults, query);
8     Cluster newCluster = new Cluster();
9     foreach Result r in remainedResults
10     {
11         if agree(r, internalSummary)
12         {
13             newCluster.addResult(r);
14             remainedResults.removeResult(r);
15         }
16     }
17     newCluster.summary = getExternalSummary(internalSummary,
18     newCluster);
19     clusters.addCluster(newCluster);
20 }
```

**Experiment**

We conduct the experiments as following. We use thirty queries of three different types, most of them were proposed in [1], those are listed as below table.

<table>
<thead>
<tr>
<th>Type</th>
<th>Queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambiguous queries</td>
<td>jaguar, apple, *orange, jobs, jordan, tiger, trec, ups, quotes, matrix</td>
</tr>
<tr>
<td>Entity names</td>
<td>*chengxiang zhai, clinton, iraq, dell, disney, world war 2, ford</td>
</tr>
<tr>
<td>General terms</td>
<td>health, maps, flower, music, chat, games, radio, jokes, graphic design, resume, time zones, travel</td>
</tr>
<tr>
<td>*Complex queries</td>
<td>*clustering search results</td>
</tr>
</tbody>
</table>

Note: * new queries and new type which are not in [1]

**Clustering quality**

Usually, users only spend time on the top k results. In this experiment, only the first one hundred results are retrieved and clustered. Clustered results are judged by human subject with three different criteria: (1) whether a results agree with the majority of results in the same cluster; (2) whether a results agree with the cluster’s title; (3) whether a cluster’s title is meaningful and agreeable with majority of inside results. These three criteria are named respectively as: (1) Precision without cluster title, (2) Precision with cluster title, and (3) Precision of cluster’s title. Following are average results we get from above thirty queries:

<table>
<thead>
<tr>
<th></th>
<th>(1) Prec. w/title</th>
<th>(2) Prec. w/o title</th>
<th>(3) Prec. of title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambiguous queries</td>
<td>0.88</td>
<td>0.96</td>
<td>0.91</td>
</tr>
<tr>
<td>Entity names</td>
<td>0.89</td>
<td>0.96</td>
<td>0.84</td>
</tr>
<tr>
<td>General terms</td>
<td>0.92</td>
<td>0.97</td>
<td>0.92</td>
</tr>
<tr>
<td>Complex queries</td>
<td>1.00</td>
<td>0.95</td>
<td>0.70</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>0.90</strong></td>
<td><strong>0.96</strong></td>
<td><strong>0.89</strong></td>
</tr>
</tbody>
</table>

**Performance**
We use Google Web API for Microsoft .NET Framework to get the search results for each query. The system runs on a machine with CPU Intel P4 2.8 GHz, 1 GB RAM, and 223.4 GB Hard Disk. The average execution time for clustering all 100 results is **0.27 seconds**

**Discussion**

As mentioned, Inductive Clustering does not require manually tuned parameters: similarity threshold and predefined number of clusters. The internal and external summaries are created before clusters are formed; hence, results tend to agree with the summaries. Furthermore, it is also easy to continue cluster a large cluster into smaller clusters.

From the experiment observation, we saw that the titles of clusters are wrong usually because of incorrect extracting external summaries from internal summaries and results inside clusters. To improve the external summary accuracy, more sophisticated technique should be applied to the task of detecting meaningful phrases basing on terms in internal summary.

**Conclusion**

Inductive Clustering is a novel technique to post-process returned search results. The approach does not require manually tuned parameters as previous approaches. The experiments show that IC works extremely well: cluster’s titles are comprehensive, results in each cluster agree with the titles, and execution time is negligible. Results organized with IC are much more easy to captured by users. We envision that IC should be implemented as an online service or integrated with existing search engine for broad usage.

**Reference**


[7] Vivisimo.com

**Appendix**

Examples of different queries and clustered results processed by IC
<table>
<thead>
<tr>
<th>#1</th>
<th>DBLP: ChengXiang Zhai</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tao Tao, ChengXiang Zhai. A two-stage mixture model for pseudo feedback, ...</td>
<td></td>
</tr>
<tr>
<td>ChengXiang Zhai. A formal study of information retrieval heuristics, ...</td>
<td></td>
</tr>
<tr>
<td><a href="http://www.informatik.uni-bremen.de/keyindices">http://www.informatik.uni-bremen.de/keyindices</a></td>
<td>38k - Cached - Similar pages</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#91</th>
<th>DBLP: Tao Tao</th>
</tr>
</thead>
<tbody>
<tr>
<td>6. EE, Tao Tao, ChengXiang Zhai. A two-stage mixture model for pseudo feedback, ...</td>
<td></td>
</tr>
<tr>
<td>ChengXiang Zhai. A formal study of information retrieval heuristics, ...</td>
<td></td>
</tr>
<tr>
<td><a href="http://www.informatik.uni-bremen.de/keyindices">http://www.informatik.uni-bremen.de/keyindices</a></td>
<td>18k - Cached - Similar pages</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#64</th>
<th>Document language models, query models, and risk minimization for ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tao Tao, ChengXiang Zhai. A two-stage mixture model for pseudo feedback, ...</td>
<td></td>
</tr>
<tr>
<td>ChengXiang Zhai, John Lafferty. Two-stage language models for information retrieval. ...</td>
<td></td>
</tr>
<tr>
<td><a href="http://portal.acm.org/citation.cfm?id=383952.383977">http://portal.acm.org/citation.cfm?id=383952.383977</a></td>
<td>Cached - Similar pages</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#71</th>
<th>Language model for IR using collection information</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Lafferty, Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. Proceedings of the 24th ...</td>
<td></td>
</tr>
<tr>
<td><a href="http://portal.acm.org/citation.cfm?id=654376.56447">http://portal.acm.org/citation.cfm?id=654376.56447</a></td>
<td>Cached - Similar pages</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#91</th>
<th>Language model for IR using collection information</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Lafferty, Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. Proceedings of the 24th ...</td>
<td></td>
</tr>
<tr>
<td><a href="http://portal.acm.org/citation.cfm?id=654473">http://portal.acm.org/citation.cfm?id=654473</a></td>
<td>Cached - Similar pages</td>
</tr>
</tbody>
</table>

| #2   | Document language models, query models, and risk minimization |
#1 -- William J. Clinton Foundation
Bill Clinton (42nd President, 1993-2001). A showcase of the Clinton Administration's legacy.
http://www.clintonpresidentialcenter.org - 28k - Cached - Similar pages

#8 -- Letter to President Clinton on Iraq
The Honorable William J. Clinton President of the United States Washington, DC.
Dear Mr. President: We are writing you because we are convinced that ... http://www.newamericancentury.org/iraq/clintonletter - 15k - Cached - Similar pages

#74 -- From Revolution to Reconstruction: Presidents: Bill Clinton
USA-project, presidents-area, information regarding the forty-second president of the United States, Bill Clinton.
http://oururl.rug.nlr-usa/FSb42/ - 5k - Cached - Similar pages

#26 -- The American President: Bill Clinton
Fact file, biography, and review of His Presidency, based on the PBS series.
http://www.americanpresident.org/history/billclinton - 50k - Cached - Similar pages

#10 -- Biography of Hillary Clinton
Biography of First Lady Hillary Clinton, wife of President William J. Clinton.
http://www.whitehouse.gov/history/biographies/firstlady - 25k - Cached - Similar pages

#2 -- William J. Clinton Foundation
President Clinton's Humanitarian Work (04/18) Clinton School Inaugural Class Announced (02/11) President Clinton Announces HIV/AIDS Initiative to Provide ... http://www.clintonfoundation.org - 29k - Cached - Similar pages
<table>
<thead>
<tr>
<th>#5</th>
<th>Vivisimo // Clustering of Search Results Increases Click-Through Rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>#35</td>
<td>Vivisimo // Citizens Can Use Latest Technology to Easily Search ...</td>
</tr>
<tr>
<td>#32</td>
<td>Vivisimo // Vivisimo Introduces Velocity for the Enterprise</td>
</tr>
<tr>
<td>#14</td>
<td>Vivisimo // Vivisimo Upgrades Velocity Search Platform</td>
</tr>
<tr>
<td>#33</td>
<td>Vivisimo // Vivisimo Introduces Gov'Tab for Clusty</td>
</tr>
<tr>
<td>#41</td>
<td>Clusty // Why Cluster?</td>
</tr>
</tbody>
</table>

Clusters:
- clustering (27)
- topic clustering (4)
- clustering software (22)
- image search (4)
- group clustering (8)
- document clustering (4)
- webmaster blog (1)
- clustering results (9)
- web clusters (2)
- main search (2)
- evaluation technique (6)
- kilometric clustering (1)
- introduction search (7)
- select results (2)
- DLIMPark (1)
- benefit clustering (1)
- doing work (1)
- break results (1)
- clustering algorithim (1)
- results newly (1)