Clustering Search Results with Carrot\textsuperscript{2}

Stanisław Osiński\textsuperscript{1} Dawid Weiss\textsuperscript{2}

Poznan Supercomputing and Networking Center,
ul. Noskowskiego 10, 61-704, Poznan, Poland
stanislaw.osinski@man.poznan.pl

Institute of Computing Science, Poznan University of Technology,
ul. Piotrowo 3A, 60-965 Poznań, Poland
dawid.weiss@cs.put.poznan.pl

January 2007
<table>
<thead>
<tr>
<th><strong>Dawid Weiss</strong></th>
<th><strong>Stanisław Osiński</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>entertainment experience</td>
<td>MSc in Design of IT Systems</td>
</tr>
<tr>
<td>MSc in Software Engineering</td>
<td>industrial/ research experience</td>
</tr>
<tr>
<td>industrial experience</td>
<td>employed in a research institute</td>
</tr>
<tr>
<td>currently in academia</td>
<td>PhD in progress...</td>
</tr>
<tr>
<td>PhD in Information Retrieval</td>
<td></td>
</tr>
</tbody>
</table>
1 Introduction to Search Results Clustering

2 Carrot$^2$ Framework

3 Lingo clustering algorithm

4 Summary
Ranked lists are not perfect
Search results clustering is one of many methods that can be used to improve user experience while searching collections of text documents, web pages for example. To illustrate the problems with conventional ranked list presentation, let’s imagine a user wants to find web documents about “apache”. Obviously, this is a very general query, which can lead to...
Ranked lists are not perfect
Clustered Search Results...

- Introduction to Search Results Clustering

- **Ranked lists** are not perfect

...large numbers of references being returned, the majority of which will be about the Apache Web Server.
Ranked lists are not perfect
A more patient user, a user who is determined enough to look at results at rank 100, should be able to reach some scattered results about the Apache Helicopter or Apache Indians. As you can see, one problem with ranked lists is that **sometimes users must go through many irrelevant documents** in order to get to the ones they want.
Search Results Clustering can help
So how about an interface that groups the search results into separate semantic topics, such as the Apache Web Server, Apache Indians, Apache Helicopter and so on? With such groups, the user will immediately get an overview of what is in the results and should be able to navigate to the interesting documents with less effort.

This kind of interface to search results can be implemented by applying a document clustering algorithm to the results returned by the search engine. This is something that is commonly called Search Results Clustering.
Search Results Clustering is an interesting problem
Search Results Clustering is an interesting problem. Search Results Clustering has a few interesting characteristics and one of them is the fact that it is based only on the fragments of documents returned by the search engine (document snippets). This is the only input an algorithm has, full text of documents is not available.
Search Results Clustering is an interesting problem.
Document snippets returned by search engines are usually very short and noisy. So we can get broken sentences or useless symbols, numbers or dates in the input.
Search Results Clustering is an **interesting problem**

- **Semantic** clusters
- **Meaningful** cluster labels
- **Small** input
Clustering Search Results... 

Introduction to Search Results Clustering

...Search Results Clustering is an interesting problem

In order to be helpful for the users, search results clustering must put results that deal with the same topic into one group. This is the primary requirement for all document clustering algorithms.

But in search results clustering very important are also the labels of clusters. We must accurately and concisely describe the contents of the cluster, so that the user can quickly decide if the cluster is interesting or not. This aspect of document clustering is sometimes neglected.

Finally, because the total size of input in search results clustering is small (e.g. 200 snippets), we can afford some more complex processing, which can possibly let us achieve better results.
Search Results Clustering is an interesting problem

...and that's why we created carrot²
1 Introduction to Search Results Clustering

2 Carrot\(^2\) Framework

3 Lingo clustering algorithm

4 Summary
Carrot² is about **search results clustering**

**Carrot² is a...**

- framework for **experimenting** with processing and presentation of search results
- framework for building **real-world** production-quality applications
- BSD-licensed **open source** project
Carrot² targets researchers, developers and end-users
Carrot$^2$ is based on **processing pipelines**
Carrot$^2$ offers **ready-to-use components**: input

Fetching search results from:

- Google (API)
- Yahoo (API)
- MSN (API)
- Open Search
- Lucene
- ODP Project
Carrot² offers **ready-to-use components**: clustering

5 search results clustering algorithms:

- **Lingo**  
  (Stanisław Osiński)

- **STC**  
  (Oren Zamir, Oren Etzioni)

- **Rough-KMeans**  
  (Ngo Chi Lang)

- **HAOG**  
  (Karol Gołęmbiak, Irmina Masłowska)

- **FuzzyAnts**  
  (Steven Schockaert)
Carrot² offers **ready-to-use components**: other

Other utilities:
- language tokenizers, stemmers and stop word lists
- very fast matrix computations library
- desktop browser application for tuning and rapid experiments
Data Mining (135)
Knowledge Discovery (17)
Data Mining Tools (8)
Data Mining and Knowledge Discovery (4)
Data Mining Software (8)
Data Mining Techniques (6)
Business Intelligence (9)
Software (19)
Tools (18)
Data Mining Applications (5)
Data Mining: Concepts and Techniques (3)
Machine Learning (6)
Information (16)
Knowledge Discovery in Databases (4)
Data Mining Products (4)

**KDNuggets: Data Mining, Web Mining, and Knowledge Discovery Guide** [1] [en]
Newsletter on the data mining and knowledge discovery industries, offering information on data mining, knowledge discovery, text mining, and web mining software, courses, jobs, publications, and meetings. http://www.kdnuggets.com/

**Data Mining and Analytic Technologies (Kurt Thearling)** [3] [en]
Kurt Thearling’s site dedicated to sharing information about data mining, the automated extraction of hidden predictive information from databases, and other analytic technologies. http://www.thearling.com/
The desktop application allows detailed tuning of each algorithm. In the query panel we have options for:

- input/algorithm selection,
- number of search results to fetch,
- default algorithm configuration settings.

After a query is performed, a result tab appears on screen allowing:

- benchmarking,
- visualization,
- on-line modification of algorithm parameters, reflected in the clusters panel.
1. **Climate Change | U.S. EPA**
   - The issue of climate change and global warming in a way that is accessible and
   - An archive of the Global Warming Site is available. ...
   - [http://www.epa.gov/climatechange/index.html](http://www.epa.gov/climatechange/index.html)

2. **Global Warming**
   - Find answers about global warming, climate change, and the world's weather. From
   - the National Oceanic and Atmospheric Administration.

3. **Global warming - Wikipedia, the free encyclopedia**
   - Read about the global warming and climate crisis debate, with information about
   - warming's causes and its effects on life on Earth. Wikipedia's user-written overview also discusses climate models, statistics, and predictions.

4. **Global Warming International Center - Home**
   - Disseminates information on global warming science and policy, serving governmental and non-governmental organizations, and industries in more than 120 countries.
   - [http://www.globalwarming.net/](http://www.globalwarming.net/)
The on-line demo is a playground for users, but also a demonstration of the technology really used by quite a number of people.
Carrot\(^2\) has a number of **real-world applications**
Commercial spin-off: Carrot Search s.c.

- A different, improved clustering algorithm – Lingo3G
- Consulting and support for the open source project
- Text-mining consultancy
Lingo3G introduces many improvements

- hierarchical results,
- a number of customization options,
- much faster and robust,
- better cluster labels.
1. Introduction to Search Results Clustering

2. Carrot² Framework

3. Lingo clustering algorithm

4. Summary
Lingo is designed specifically for search results clustering

- Semantic clusters
- Meaningful cluster labels
- Small input
The primary assumption we made when working Lingo was that it should be an algorithm specifically designed to handle search results clustering. Therefore our main focus was the quality of cluster label. We also were aware that, due to the small size of input, we could afford more complex processing.
Cluster description has a priority

Classic clustering

- Snippets
- Cluster
- Clusters
- Describe
- Results
Having in mind the requirement for high quality of cluster labels, we experimented with **reversing the normal clustering order** and giving the cluster description a priority.

In the classic clustering scheme, in which the algorithm starts with finding document groups and then tries to label these groups, we can have situations where the algorithm knows that certain documents should be clustered together, but at the same time the algorithm is unable to explain to the user what these documents have in common.
Cluster description has a priority

**Classic clustering**

- Snippets → cluster
- Clusters → describe
- Describe → Results

**Description comes first clustering**

- Snippets → find labels
- Labels → add snippets
- Snippets → Results
We can try to avoid these problems by starting with finding a set of meaningful and diverse cluster labels and then assigning documents to these labels to form proper clusters. This kind of general clustering procedure we called "description comes first clustering" and implemented in a search results clustering algorithm called LINGO.
Phrases are good label candidates

1. **Apache Software Foundation**
   - Membership-based, not-for-profit corporation that exist to provide organizational, legal, and financial support for the Apache open source software projects.
   - Category: Unix Servers > Apache
   - [More from this site - Save](http://www.apache.org)

2. **Apache HTTP Server Project**
   - Effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows NT.
   - [httpd.apache.org](http://httpd.apache.org) - 32k - Cached - More from this site - Save

3. **Download - The Apache HTTP Server Project**
   - Essentials, Downloads, Get Involved. Subprojects. Use the links below to download the Apache HTTP Server from one of our mirrors. You must verify the integrity of the downloaded files using signatures downloaded from our main distribution directory.
   - [http://mirror.apache.org/download.cgi](http://mirror.apache.org/download.cgi) - 17k - Cached - More from this site - Save

4. **Apache.com - Providing Web Server and Network Security Resources**
   - Web health control panel. Control panels provide a convenient interface with which to create new users, assign admin privileges, bind DNS entries, control servers, sites, and more. ... One ADC, Zeus Enterprise Traffic Manager (ZETM) can accelerate Apache by as much as 100 times ... The Apache HTTP Server Project develops and maintains an open-source HTTP server for
   - [www.apache.com](http://www.apache.com) - 15k - Cached - More from this site - Save

5. **The Jakarta Site - The Jakarta Project - Java Related Products**
   - ... solutions and is a part of The Apache Software Foundation (ASF) which encourages a collaborative, consensus ... Copyright © 1999-2005, The Apache Software Foundation. Legal information
   - Category: Unix Servers > Apache
   - [RSS, Wyatts 25k - Add to My Yahoo](http://jakarta.apache.org)

   - ... Apache Micro located in Irvine California is one of the world's leading analog modern and broadband ... manufacturing broadband and analog modules. Apache-Micro also produces a wide range ... [www.apache-micro.com](http://www.apache-micro.com) - 11k - Cached - More from this site - Save

7. **Apache XML Project**
   - To provide commercial-quality standards-based XML developed in an open and cooperative fashion and to provide feedback to standards bodies.
   - Category: Unix Servers > Apache
   - [xml.apache.org](http://xml.apache.org) - 31k - Cached - More from this site - Save

8. **Apache HTTP Server Version 2.0 Documentation - Apache HTTP Server**
   - [http://httpd.apache.org/docs/2.0](http://httpd.apache.org/docs/2.0) - 71k - Cached - More from this site - Save

... and 300 more...
So how do we go about finding good cluster labels? One of the first approaches to search results clustering called Suffix Tree Clustering would group documents according to the common phrase they shared. **Frequent phrases** are very often collocations (such as Web Server or Apache County), which increases their descriptive power. But how do we select the best and most diverse set of cluster labels? We've got quite a lot of label candidates...
Approximate matrix factorizations can find labels

$$\begin{bmatrix}
\text{doc 1} & \text{doc 2} & \text{doc 3} & \text{doc 4} \\
\text{term 1} & * & * & * & * \\
\text{term 2} & * & * & * & * \\
\text{term 3} & * & * & * & * \\
\text{term 4} & * & * & * & * \\
\text{term 5} & * & * & * & * \\
\text{term 6} & * & * & * & * \\
\end{bmatrix} = A$$

Term X

Term Y

- a document

Choosing a cluster's label

Cluster content discovery

- assigned

- unassigned
Approximate matrix factorizations can find labels

We can do that using **Vector Space Model** and matrix factorizations. To build the Vector Space Model we need to create a so called term-document matrix: a matrix containing frequencies of all terms across all input documents. If we had just two terms – term X and Y – we could visualise the Vector Space Model as a plane with two axes corresponding to the terms and points on that plane corresponding to the actual documents.
Approximate matrix factorizations can find labels

\[ A = \begin{bmatrix}
* & * & * & * \\
* & * & * & * \\
* & * & * & * \\
* & * & * & *
\end{bmatrix} \rightsquigarrow \begin{bmatrix}
* & * \\
* & * \\
* & * \\
* & * 
\end{bmatrix} \times \begin{bmatrix}
* & * & * & * \\
* & * & * & * \\
* & * & * & * \\
* & * & * & *
\end{bmatrix} \]

- label candidate

Term X

Term Y

- a document

Documents in terms' space

Term X

Term Y

- base vectors

Visual representation of base vectors

("concepts") acquired by SVD decomposition of the term-document matrix

Cluster label candidates expressed in the vector space of the documents

Choosing a cluster's label

Term X

Term Y

Cluster content discovery

- assigned

- unassigned

base vectors

coefficients

- base vectors
The task of an **approximate matrix factorization** is to break a matrix into a product of usually two matrices in such a way that the product is as close to the original matrix as possible and has much lower rank. The left-hand matrix of the product can be thought of as a set of base vectors of the new low-dimensional space, while the other matrix contains the corresponding coefficients that enable us to reconstruct the original matrix.

In the context of our simplified graphical example, base vectors show the general directions or trends in the input collection.
Approximate matrix factorizations can find labels
Approximate matrix factorizations can find labels

Please notice that both frequent phrases and base vectors are expressed in the same space as the input documents (think of the phrases as tiny documents). With this assumption we can use e.g. cosine distance to find the best matching phrase for each base vector. In this way, each base vector will lead to selecting one cluster label.
Cosine distance can find documents

- label candidate
- a document
- base vectors

Visual representation of base vectors (“concepts”) acquired by SVD decomposition of the term-document matrix

Cluster label candidates expressed in the vector space of the documents

Choosing a cluster's label

Cluster content discovery
- assigned
- unassigned
To form proper clusters, we can again use cosine similarity and assign to each label those documents whose similarity to that label is larger than some threshold.
Giving priority to labels pays off.
Here we show how 100 search results obtained from Yahoo! for the query “tiger” were clustered by Lingo (with SVD decomposition), STC and Rough K-Means algorithms. As you can see Lingo did not manage to avoid generating useless labels (such as “sign” or “use”), but it to highlight some tiger-related topics that the remaining algorithms did not discover (helicopter, java, security tool).
1 Introduction to Search Results Clustering

2 Carrot² Framework

3 Lingo clustering algorithm

4 Summary
Exploit the potential of existing ontologies?
Investigate support for more languages.
Investigate more data sources.
References


**Carrot² links**

- On-line demo:
  http://www.carrot2.org

- Open source project:
  http://project.carrot2.org

- SourceForge (repository etc.):
  http://sourceforge.net/projects/carrot2

- Carrot Search:
  http://www.carrot-search.com