**Bachelor thesis**

# Clustering the Web
# Comparing Clustering Methods in Swedish

by

**Joel Hinz**

LIU-IDA/KOGVET-G--13/035--SE

2013-07-02



Supervisor: Arne Jönsson

Examiner: Carine Signoret

**Abstract**

Clustering – automatically sorting – web search results has been the focus of much attention but is by no means a solved problem, and there is little previous work in Swedish. This thesis studies the performance of three clustering algorithms – k-means, agglomerative hierarchical clustering, and bisecting k-means – on a total of 32 corpora, as well as whether clustering web search previews, called snippets, instead of full texts can achieve reasonably decent results. Four internal evaluation metrics are used to assess the data. Results indicate that k-means performs worse than the other two algorithms, and that snippets may be good enough to use in an actual product, although there is ample opportunity for further research on both issues; however, results are inconclusive regarding bisecting k-means vis-à-vis agglomerative hierarchical clustering. Stop word and stemmer usage results are not significant, and appear to not affect the clustering by any considerable magnitude.

**Keywords:** clustering, web, search results, snippets, k-means, agglomerative hierarchical clustering, bisecting k-means, swedish

# Contents

# 1 Introduction

The internet can be a scary place. Not all are privileged enough that they can sort search results quickly and easily in their minds. Thus, if computers can be taught to perform this sorting automatically, the web would be made more accessible to everybody, regardless of their reading ability – and users without reading disabilities have been shown to benefit from such systems, too [14, 28].

It is the aim of this thesis to aid one such project, *Webbklustring*[1] ("Web Clustering"), built at Linköping University under the supervision of Arne Jönsson, by comparing various *clustering* algorithms – methods of putting web pages in groups automatically, based on their likeness to each other – and ultimately by suggesting one which to use in the actual project.

In addition, clustering of full web pages is compared to clustering of snippets[2], in order to advice on using one over the other in the final product. And finally, two more questions regarding clustering practices are researched: whether a list of stop words[3] should be used, and whether stemming[4] is appropriate.

## 1.1 Clustering



Clustering is, essentially, the putting of various things together in groups, based on their proximity or likeness. It has been widely researched by and applied in plenty of areas, such as bioinformatics, machine learning, and pattern recognition. Web search clustering may be a relatively young field, but document clustering has been around for decades, working with the assumption that "closely associated documents tend to be relevant to the same request" – the *cluster hypothesis* [29].

The goal of any clustering is to identify data which are in some way similar. Any two clusters should be as dissimilar as possible, yet data within a cluster should be as similar as possible. Clustering can, essentially, be performed in order to identify a degree of similarity, to gain insight about data and detect patterns or anomalies, and to organise and summarise data [12]. In

[1] Currently available at http://www.ida.liu.se/projects/webbklustring/

[2] The preview Google, Bing, or a similar search results provider displays. Typically includes title, a short extract, and some metadata.

[3] Words too common for inclusion.

[4] The reduction of inflected word forms to their roots, e.g. *temptation* becomes *temptat*.

the case of *Webbklustring*, the goal of the clustering is to organise documents by means of similarity.

## 1.2 Webbklustring

The *Webbklustring* service is being built as an aid for information retrieval with the main purpose of helping users screen web searches in order to detect if results contain new information or if they simply state the same things. It is being built upon two predecessors: *Webblättläst*, which ranks search results by readability, and *EasyReader*, which summarises texts, thus making them more easily accessible and intelligible.

In addition to detecting duplicate content, the application will also be able to summarise web pages and determine text legibility. The intention is to provide inexperienced users and people who suffer from reading and/or writing disabilities with tools to retrieve information and acquire knowledge on their own, instead of having to rely on others to assist them.

The full scope of the project contains not only technical aspects such as clustering and multi-document summarisation, but also interaction design. Eventually, the goal is to provide to the end user a graphical user interface enhanced with further features to facilitate searching and reading of web pages, although this thesis will be restricted to clustering.

A typical example use would be dyslexic users searching for information but finding the task hard since it takes them longer to read. They benefit from short summaries, since it helps them decide whether a full article is relevant or not, and they may want to understand text essence more easily. Grouping documents into clusters means that they do not have to go through several similar texts in order to find one they're looking for, and a visual representation lets them identify desired documents.

Another example is that of exchange students, or second-language Swedish speakers in general, who wish to learn the language but finds some texts too difficult and lose interest. For them, *Webbklustring* may become a tool which assists in finding texts on appropriate difficulty levels. In addition, since they read articles that interest them, grouping documents together by clustering provides a way of finding information that they want to read more about.

# 2 Clustering

This section describes how clustering works, and gives a rundown on clustering practices and associated topics which are applicable to this thesis. Different algorithms are introduced, and potential problems are discussed briefly, as is previous research on clustering in Swedish.

## 2.1 Clustering algorithms

The most common algorithms by far in web search clustering are *k-means* and *hierarchical* clustering, or derivations thereof. They have in common that they operate using the *vector space model*, according to which documents (or any other objects, technically) are converted into corresponding term vectors counting the term frequencies of all words within the documents. The vector space model, first described in 1975 [24], is "mathematically well defined and well understood" [23], and, as such, a good fit for this thesis.

In k-means, a pre-defined number ($k$) of *centroids*[5] are initially chosen at random. Each document is put into the cluster which has a centroid closest to the document mean, with new centroids calculated after every such inclusion, and the process iterates until all documents have been assigned to a cluster. The algorithm can be repeated for increased validity if so desired.

Hierarchical clustering methods can be divided into agglomerative (bottom-up) and divisive (top-down) approaches. In the former, all documents are placed in one large cluster, which is split recursively until a stopping criterion is met. In the latter, every document starts in a cluster of its own, and the two closest clusters are merged, again recursively until a stopping criterion is met.



As document collections grow, so, obviously, do their associated vectors, increasing time and memory complexity. For this reason, methods to approximate full vectors while maintaining their most important features – such as *Latent Semantic Indexing* [8] and *Random Indexing* [23] – have been proposed, and are quite interesting, but for this thesis, it will be assumed that full vectors are available, and that CPU power is no object, as the number of documents is limited by the maximum number of search results returned.

[5] A centroid is the cluster mean, which is the vector containing the average of each dimension in the documents associated with the cluster. For instance, the cluster mean of {[5, 2], [3, 4]} is [4, 3].

## 2.2 Labelling

While clustering in itself strives only to group documents, a *Webbklustring* end user will have little use for the accumulated clusters without a label accompanying them. For this reason, it has been suggested that labelling is far more important than cluster optimisation, as a good label with a bad cluster is still better than a good cluster with a bad label [3]. Finding the best label turns out to be a serious problem indeed, and one which has been the focus of much attention within information retrieval – lately more so, even, than cluster optimisation [3].

Although the matter of solving the labelling problem — a topic exhaustively discussed elsewhere – will not be explicated here, it does have consequences for the final project. Humans tend to more often than not apply labels not occurring in the clustered texts [6], using semantic knowledge computers cannot access, and it has been argued that for web clustering to be actually useful, better clustering techniques are required to generate readable names [32].

## 2.3 Waiting vs. snippeting

In addition to the labelling problem discussed above, there is also the issue of what data to use. When making a Google search (or using a similar search engine), the querist is presented with a brief summary of each link, to give a hint of the contents therein. This summary is called a *snippet*, and affects clustering hugely.



The obvious advantage of clustering snippets instead of full web pages is that only one page load is required, instead of $n + 1$, where $n$ is the number of search results returned. This is greatly beneficial to the end user, who doesn't have to suffer possibly egregious loading times [32], and most web search clustering research has been using snippets for this reason. Snippets may also take document metadata such as title and description into account, something full document parsing usually does not.

However, since snippets contain only a fraction of the complete text contents, any algorithm employing them suffers a great penalty to its clustering quality. One study found that snippet clustering yields over 40 % worse results than full page clustering [18], and although Google's snippeting capabilities have certainly improved since this number was given in 2006, there's no reason to assume that full page clustering does not still perform significantly better.

In fact, snippets were easier to extract when web pages were simpler — so much so, that snippet clustering was considered almost as good as full page clustering in a seminal paper by Zamir and Etzioni in 1998 [31].

Furthermore, snippets are "highly biased toward the query" [19], which while in itself not necessarily a bad thing means that they produce different results than, say, document summarisation techniques. This causes further problems yet for labelling algorithms, since they have even less to go on [25]. And finally, polysemous terms cause worse issues when occurring in snippets, since they are given a proportionally higher importance than in larger documents with more words [25].

Whether snippets are good enough vis-à-vis full web pages for clustering Swedish search results is one of the two main questions for this thesis.

## 2.4 Previous work in Swedish

Some other document clustering efforts have been made in Swedish, notably by Magnus Rosell [22], who has looked into clustering in general, as well as in the specific news articles genre. Per-Anders Staav has researched how Random Indexing can be applied to Swedish texts [26], and Carl-Oscar Erneholm, most recently, covered clustering briefly as a pre-requisite for performing multi-document summarisation [10].

This thesis concentrates on web search results specifically, however, and that appears to be a largely unexplored topic within Swedish clustering, not due to a lack of interest but rather because of a lack of resources – *Webbklustring* is, in fact, one of the first projects to receive funding for this kind of research.

# 3 Method

Apart from performing the actual clustering, two distinct additional phases of the project were required: *corpora*[6] with test data were assembled, and the results of the clustering were evaluated using standard metrics. This section describes each phase separately.

## 3.1 Creating corpora

In order to create corpora, four Google searches were made, with the search engine set to Swedish and with no prior browser or account history. Each search yielded a hundred results (as that is the maximum number of returned results per page for a Google query), and the source code of each result was saved, along with its URL, search result order ("pagerank"), search term, and snippet.

### 3.1.1 Choosing queries

It has been proposed to use three levels of specialisation for queries: ambiguous, entity name, and general term [32], and the first three queries used match this suggestion. These queries were, in that order: "jaguar", being the de facto standard query in web search results clustering, and having multiple meanings – felid, car make, gaming console, operating system, etc. – in Swedish as well as in English; "ibuprofen", an anti-inflammatory drug; and "katter" (*cats*). In addition, a query in the form of a question – "hur startade ryska revolutionen?" (*how did the Russian Revolution start?*) – was used as well, as *Webbklustring* end users may want to enter questions occasionally, and this needed to be researched, too.



### 3.1.2 Data extraction

Since HTML source code usually contains a lot more data than just the text content which the user is interested in reading, the pages needed to prepared for clustering by means of HTML sanitising. This is a non-trivial task, and two different methods using different approaches were considered:

[6] Collections of documents. Each collection is called a corpus.

1. The Boilerpipe algorithm [15], developed in Java at the Leibniz Universität Hannover, applies quantitative linguistics theory in order to generate document contents using shallow text features. The method has been made freely available online by its inventors[7]. It is worth noting that the authors also supply their test datasets freely for research purposes – however, as these are not in Swedish and as they concern specific domains, they were not used in this thesis.

2. AlchemyAPI[8] is a text mining platform built to perform semantic analysis in the natural language processing field. Although free to use for the purposes of this comparison, its algorithms are primarily commercial in nature, and therefore only the results are available for further inspection. Unlike Boilerpipe, AlchemyAPI was not built with the single purpose of extracting boilerplate text, but for this project, no other features of the API were used.

The top ten results of each Google search were also sanitised manually, thereby creating a *gold standard*[9] of reference texts to which the results of the above two algorithms could be compared. Obviously, this method is subjective, but its results can be seen as one out of possibly several correct answers, and thus serve as benchmark comparison points.

For the comparison, the standard *F-measure* algorithm[10] was employed. Boilerpipe was found to be better, but not significantly so – although with mean scores of 0.815 and 0.739, respectively, both tools performed generally well. Considering the slightly better results as well as its open-source nature, the full web page text contents used to build the final corpora for this thesis were extracted using Boilerpipe. It is also worth noting that not all search results were readily parsable. For instance, sites using frames caused problems for both algorithms, and some results – such as Microsoft PowerPoint slides – are not always readable by standard web browsers.



### 3.1.3 Computing vectors

In order to perform web page clustering, three steps are required [3]: tokenisation, the separation of every syntactic entity in the text; stemming,

[7] https://code.google.com/p/boilerpipe/, see also [15]

[8] http://www.alchemyapi.com/

[9] A set of answers or results which is considered to be correct, for the purposes of evaluation.

[10] The F-measure is the harmonic mean of *precision* (percentage of correct retrievals) and *recall* (percentage of relevant retrievals).

the reduction of inflected word forms to their roots (optional for larger corpora, yet crucial for smaller samples and in inflection-heavy languages); and data extraction, equivalent here to the removal of HTML. The tokenisation was performed by removing anything superfluous from the samples, and the Swedish algorithm included in the default Snowball language package [20] was used to perform the stemming.

A list of stop words was considered, and corpora were created with as well as without stop words, in order to evaluate whether excluding common words affected performance. The full list can be found in Appendix A.

Standard term frequency–inverse document frequency (*tf-idf*) vectors were then created for each resulting corpus. This is a common metric in information retrieval, and consists of two parts: the term frequency *tf* is the number of times a term $i$ occurs in a document $j$, and the inverse document frequency *idf* is the the number of documents in a cluster divided by the number of documents in which $i$ occurs at least once, often applied a log function upon. [13] The *tf-idf* weighting for a term $i$ in a document vector $j$ occuring in the document collection $N$ is thus

$$w_{i,j} = \mathrm{tf}_{i,j} * \log(\frac{N}{n_i}) \tag{1}$$

As a result, every document is represented by a row in the corpus vector, and every term by a column. It is then easy to look up the tf-idf score for any term in any document. The number of dimensions in the corpus – and in each document – is simply the number of terms.



## 3.2 Clustering

As mentioned earlier, k-means and the two types of hierarchical clustering are the most common approaches to web and/or document clustering in the literature. In fact, the agglomerative hierarchical clustering algorithm is probably the most frequently used technique for document clustering specifically [5]. For the divisive hierarchical clustering, bisecting k-means was used. These three will be described below.

### 3.2.1 The k-means algorithm

Without a doubt, k-means is certainly the most known clustering algorithm, and all applications considered, it may well be the most used, too. Its major

advantages are that it's quite simple to implement and use, and also often rather fast.

Upon initialisation, k-means chooses $k$ documents at random to use as starting centroids for each cluster. Then, each document in the collection is placed in the cluster to which it is the nearest, after which every cluster centroid is re-calculated. This process of assigning documents and re-calculating centroids is repeated until documents no longer change clusters (although it is also possible to stop at other times, e.g. when a set number of iterations has been reached or clusters are good enough based on some evaluation algorithm). [17]

Pseudo-code for the algorithm is shown in Algorithm 1.

**Algorithm 1:** The k-means algorithm

**input** : number of clusters $k$, document collection $D$
**output**: a set of $k$ clusters

```
choose centroids at random from D
while no document changes clusters do
    foreach document d in D do
        calculate distance from d to each centroid
        assign d to the cluster of the nearest centroid
    end
    foreach c in centroids do
        re-calculate c as average of all documents in cluster
    end
end
```



Computing cluster averages is done by calculating the mean of each dimension for all documents in that cluster:

$$\text{avg}_j = \frac{1}{|C_j|} \sum_{\text{dim}_i \in C_j} \text{dim}_i \tag{2}$$

where $|C_j|$ is the number of documents in the cluster $C_j$.

The distance between any two points $p$, $q$ in the vector space is defined as the *Euclidean distance* between them, so that

$$\text{dist}(\text{p}, \text{q}) = \|\text{p} - \text{q}\| = \sqrt{\sum_{i=1}^{n} (\text{p}_i - \text{q}_i)^2} \tag{3}$$

### 3.2.2 Deciding on a $k$

There is an obvious problem for the k-means algorithm: how would the computer (or, to contrast, a human) know what number of clusters there will be before performing the search?

It is worth noting that too many clusters will confuse the reader; no more than ten clusters in total has been suggested [25]. As the aim of this thesis is tightly connected to an end-user system, it will be assumed that the value of $k$ is set to 7 – the number of clusters which will be displayed in the final application.

This serves the double purpose of setting a stopping criterion for the hierarchical algorithms as well. Since there will be seven clusters in total, the algorithms can stop once seven clusters have been reached.

### 3.2.3 Agglomerative hierarchical clustering

The basic concept for the agglomerative hierarchical clustering algorithm (AHC) is simple: start with one cluster for each document, then merge the two closest clusters recursively until either a stopping criterion is met or only a single, very large cluster remains [17].

Unlike the other two algorithms used, AHC therefore cannot yield different results when run twice on the same document collection.

There are multiple ways of computing the distance between clusters to decide which the closest ones are, called linkages. For this thesis, the *centroid-based linkage* method was employed, which uses the distance between cluster centroids. *Cosine similarity* was used as distance measure, and is computed as

$$\cos_{\mathrm{p,q}} = \frac{\mathrm{p} \cdot \mathrm{q}}{\|\mathrm{p}\|\|\mathrm{q}\|} = \frac{\sum_{i=1}^{n} \mathrm{p}_i \mathrm{q}_i}{\sqrt{\sum_{i=1}^{n} (\mathrm{p}_i)^2} \sqrt{\sum_{i=1}^{n} (\mathrm{q}_i)^2}} \tag{4}$$

for two clusters $p$ and $q$. AHC was implemented as seen in Algorithm 2.

**Algorithm 2:** Agglomerative hierarchical clustering

**input** : number of clusters $k$, document collection $D$
**output**: a set of $k$ clusters

```
foreach document d in D do
    initialise cluster
    assign d to cluster
end
while num clusters > k do
    foreach i, j in clusters do
        calculate proximity between i and j
    end
    merge clusters with lowest proximity
end
```

#### 3.2.4 Divisive hierarchical clustering, or bisecting k-means

At first glance, divisive hierarchical clustering sounds as simple as its agglomerative counterpart: start with all documents in a single cluster, and keep splitting clusters until either a stopping criterion is met or every cluster contains only a single document. But whereas AHC can simply derive its next merging from a chosen linkage measure, the number of possible intra-cluster combinations to compare is enormously larger than the number of possible inter-cluster ones.

To combat this, the *bisecting k-means* algorithm [27] was deviced. For each iteration, the worst cluster is split in two by means of k-means with a $k$ of 2. It has been found that simply choosing the largest cluster every time does not typically yield worse results than computing intra-cluster similarity [27], and hence this method was chosen for this study. In order to calculate the largest cluster, each document's Euclidean distance to its cluster centroid is measured[11], and all such distances are summed for each cluster.

[11] See section 3.2.1.

Bisecting k-means is performed as seen in Algorithm 3.

**Algorithm 3:** Bisecting k-means

**input** : number of clusters $k$, document collection $D$
**output**: a set of $k$ clusters

```
initialise cluster
put all documents in cluster
while num clusters < k do
    foreach c in clusters do
        calculate size of c
    end
    split largest cluster using k-means (k = 2)
end
```

## 3.3 Evaluation

Four standard metrics were used to evaluate the results: intra-cluster similarity, sum of squared errors, Dunn index, and Davies–Bouldin index. These are all examples of *internal evaluation*, which is based entirely on the resulting data, whereas its counterpart *external evaluation* compares the data to some gold standard or training set. The aim of any internal evaluation scheme is to measure the extent to which similarity within a cluster is maximised, while similarity between clusters is minimised.

Although external evaluation appears to be the more common option in clustering, the choice to use internal measures was made in order to match a real-world scenario and because constructing gold standards – which are mostly available as predefined datasets in other research – for the vast amount of data was deemed infeasible. It should be noted, however, that clustering evaluation is a very hard-to-solve problem, and currently one for which nobody has an inarguably good solution.



### 3.3.1 Intra-cluster similarity

Intra-cluster similarity (ICS) measures content compactness by calculating for each cluster the average cosine similarity between its centroid $o$ and all of its documents [32]. Similarity is measured by distance, therefore a low ICS score implies higher similarity between documents in a cluster. ICS is computed as:

$$\mathrm{ICS} = \frac{1}{|C_j|} \sum_{\mathrm{dim}_i \in C_j} cos(\mathrm{dim}_i, \mathrm{o}) \tag{5}$$

Recall again from section 3.2.1 that the cluster centroid $o$ is calculated as the average of every dimension in a cluster:

$$\mathrm{o}_j = \frac{1}{|C_j|} \sum_{\mathrm{dim}_i \in C_j} \mathrm{dim}_i \tag{6}$$

Also recall from section 3.2.3 that cosine similarity is defined as

$$\mathrm{cos}_{\mathrm{p,q}} = \frac{\mathrm{p} \cdot \mathrm{q}}{\|\mathrm{p}\|\|\mathrm{q}\|} = \frac{\sum_{i=1}^{n} \mathrm{p}_i \mathrm{q}_i}{\sqrt{\sum_{i=1}^{n} (\mathrm{p}_i)^2} \sqrt{\sum_{i=1}^{n} (\mathrm{q}_i)^2}} \tag{7}$$

for two clusters $p$ and $q$.

### 3.3.2 Sum of squared errors

The sum of squared errors (SSE) is a commonly used metric in document clustering (e.g. [30]), and since the goal of k-means essentially is to minimise the SSE, it makes sense to use it. With four queries and three binary variables (full texts/snippets, stopword usage, stemming usage), there were $4*2*2*2 = 32$ corpora in total. To compensate for various corpora having different sizes, the SSE was also normalised as a function of the number of dimensions in each cluster. Otherwise, the SSE would be skewed as a result of some corpora having far more dimensions than others.



To calculate the SSE for a cluster, the following formula is used:

$$\mathrm{SSE} = \sum_{i=1}^{k} \sum_{\mathrm{x}_j \in C_i} \|\mathrm{x}_j - o_i\|^2 \tag{8}$$

meaning that each dimension's squared error rate from the cluster centroid is calculated for each document in each cluster, after which they are all summed together.

In addition to regular SSE, a derivation thereof called SSE/DIM is introduced as well. This is simply defined as the SSE, divided by the number of dimensions in a cluster. It thus measures average dimensional error rather than total dimensional errors.

### 3.3.3 Dunn index

The Dunn index, first used in 1974 [9], is the ratio between smallest inter-cluster distance and largest cluster size [2]. The larger the Dunn index, the better. Measuring by Dunn index does, unfortunately, carry with it the drawback of a high computational cost, as well as a susceptibility to noise, and it is hence not recommended for live applications [16].

There are multiple ways of calculating both dividend and divisor. For inter-cluster distance, the distance between centroids, called *centroid linkage distance*, was used.

$$\mathrm{d}_C(C_i, C_j) = \mathrm{d}(o_i, o_j) \tag{9}$$

For cluster size, or *intra-cluster diameter*, centroid diameter distance was used, defined as the double average distance between centroid and all documents.

$$\Delta(C) = \frac{2}{|C|} \sum_{i \in C} \mathrm{d}(i, o) \tag{10}$$



### 3.3.4 Davies–Bouldin index

Whereas the Dunn index computes extremes, its Davies–Bouldin counterpart measures average error [7], or put differently, average similarity between each cluster and the cluster to which it is the most alike [16].

The formula for the Davies–Bouldin index is hence very simple:

$$\mathrm{DB} = \frac{1}{N} \sum_{i=1}^{N} \mathrm{D}_i \tag{11}$$

where $N$ is the number of clusters and $\mathrm{D}_i$ is the highest symmetry to any other cluster for a cluster $i$. For the cluster comparison, centroid diameter and centroid linkage distances were again used, so that the $\mathrm{D}_i$ value is defined as

$$\mathrm{D}_i = \max_{j : i \neq j} \frac{\Delta_i + \Delta_j}{\mathrm{d}(o_i, o_j)} \tag{12}$$

where $\Delta$ is the double average distance from centroid to all documents; see Equation 10.

## 3.4 Understanding the evaluation

All of the results do not, as will be thoroughly explicated in the Results and Discussion sections, have obvious interpretations. Hence, it's not always easy to draw conclusions from them.

To combat this problem, it is well worth looking at the different evaluation metrics, in order to understand what each measures and what the measurements mean. Projects such as *Webbklustring* can then be made choices for in accordance with what the specific requirements for each application are.

### 3.4.1 Measuring errors

The SSE metric concentrates, as its name implies, on error measuring. Since errors are defined as a function of vector-space distance, more specifically to a cluster centroid, SSE per definition measures intra-cluster compactness.

It follows logically, then, that the odds of getting a lower SSE score increase as clusters grow larger in dimensions. Each additional dimension brings another potential error to the table. SSE/DIM is an attempt to remedy this by simply dividing the SSE by the number of dimensions. Though both are relevant in their own rights, the results were measured by query average and are, as such, comparable by SSE – if not it would have been a pointless measurement method. Were the queries considered variables, like e.g. stemmer usage or algorithm choice, SSE scores would have run the risk of becoming heavily biased.



ICS takes a roughly similar distance-based approach, but, as seen in section 3.3.1, calculates the mean similarity – which, in this case, is a form of error rate, based on a similarity measure different from SSE – rather than simply adding all squared errors together.

These metrics all focus on cluster compactness, but they say essentially nothing about inter-cluster relationships or distances.

### 3.4.2 Measuring ratios

Whereas the error-measuring metrics give a good idea of cohesion, Dunn and Davies–Bouldin indices both use ratios between inter- and intra-cluster values.

Dunn uses smallest inter-cluster distance, defined as centroid-to-centroid distance, for the inter-cluster part, and largest cluster size as the intra-cluster measure. Hence, the longer the distances, and the smaller the clusters, the better the Dunn score. As such, it tries to combine cluster separation and cohesion into one value.

Davies–Bouldin is slightly more contrived, measuring the average highest similarity. It, too, uses largest cluster size and distance between centroids, but as it is a measure of similarity, it strives to be minimised, as low similarity means that clusters are well separated and cohesive.

### 3.4.3 Measurement conclusions

Though all of the above metrics can be (and are) employed on the clusters in this thesis, it is worth noting that projects may face varying requirements.

An application which aims only to make sure that documents within a clusters are as alike as possible may want to look primarily at SSE and ICS.

Contrariwise, if knowing that clusters are well separated, Dunn and Davies–Bouldin indexing may be preferable. Furthermore, since the former measures extremes, it may notice differences more clearly, but at the cost of increasing variance. The latter, by contrast, measures averages and thus inherently applies a form of smoothing on the values.

The results discussion will offer suggestions based on the above arguments when applicable.

# 4 Results

In total, there were 96 different clusters, formed by the Cartesian product of all criteria used to build them.

- Full text or snippet
- Query (out of four)
- Stemmer usage, yes/no
- Stop words usage, yes/no
- Clustering algorithm (out of three)

Considering that there were four different evaluation metrics as well, it is certainly possible to simply list data upon data upon data. In order to be able to draw satisfying conclusions, there is thus vast need for an easy way to determine how the results ought to be sorted.

In light of the goals of the thesis (see Section 1), the search query should be considered an asset to the main goals, rather than being a variable in its own right. It was therefore used to reduce the number of clusters.

For instance, there were four clusters with the criteria combination {full text, k-means, stop words used, stemmer not used}, one for each query. Instead of using all four when calculating significance, their results were combined into only one SSE score, which was simply the average of all four individual SSE scores. The same goes for the other metrics as well.

By taking the average evaluation results, the 96 clusters were reduced to only 24 – a quarter of the original size. Standard statistical tests were then employed to perform comparisons on the resulting 24 clusters.

Results tables for full as well as reduced clusters are available in Appendices B and C.



## 4.1 Algorithms

Picking a clustering algorithm is arguably the single most important question for this thesis, and as such, the results comparing algorithms should be considered first and foremost.

One-way ANOVA tests were made on the data, with Tukey's HSD as post-hoc comparison. The results of these tests are available in Tables 1 and

2.

Table 1: Results by algorithm, SSE and ICS evaluation

| | SSE | | SSE/DIM | | ICS | |
|---|---|---|---|---|---|---|
| | Mean | Std err | Mean | Std err | Mean | Std err |
| AHC | 9,026 | 2,629 | 3.528 | 0.612 | 426 | 368 |
| Bisecting | 6,441 | 1,312 | 2.864 | 0.553 | 3785 | 2905 |
| k-means | 603,878 | 204,792 | 118.555 | 20.254 | 141863 | 61694 |
| F | 8.472 | | 32.386 | | 5.122 | |
| Significant | **Yes (0.002)** | | **Yes (0.000)** | | **Yes (0.015)** | |

Table 2: Results by algorithm, Dunn and Davies–Bouldin evaluation

| | Dunn | | Davies–Bouldin | |
|---|---|---|---|---|
| | Mean | Std err | Mean | Std err |
| AHC | 0.093 | 0.006 | 0.332 | 0.011 |
| Bisecting | 0.074 | 0.004 | 0.236 | 0.005 |
| k-means | 0.106 | 0.019 | 0.476 | 0.010 |
| F | 1.87 | | 27.415 | |
| Significant | No (0.179) | | **Yes (0.000)** | |



The results of the Tukey's HSD post-hoc comparison were as follows:

- SSE ($p = 0.002$): Regular k-means performed significantly worst. Bisecting k-means had the best results, but they were not significant when compared to AHC.
- SSE/DIM ($p = 0.000$): As with SSE, regular k-means performed significantly worst of the group. Bisecting k-means had the best results, but not significantly so over AHC.
- ICS ($p = 0.015$): Again, regular k-means performed significantly worst. AHC did better than bisecting k-means, but not significantly.
- Dunn ($p = 0.179$): Here, regular k-means did best, and AHC came in second, but no results were significant.
- Davies–Bouldin ($p = 0.000$): Bisecting k-means performed significantly

better than regular k-means, which in turn performed significantly better than AHC.

## 4.2 Full text or snippets

The second most important question to answer is whether snippets are good enough to use vis-à-vis full web pages. Table 3 contains the test data for this comparison, $p > .05$ for a two-tailed independent t-test.

Table 3: Results by text source

| | Full text | | Snippets | | | |
|---|---|---|---|---|---|---|
| | Mean | Std err | Mean | Std err | t | Significant |
| SSE | 387,922 | 161,517 | 24,975 | 8,960 | 2.244 | **Yes (0.046)** |
| SSE/DIM | 58.1 | 24.1 | 25.2 | 9.0 | 1.279 | No (0.222) |
| Dunn | 0.107 | 0.012 | 0.075 | 0.004 | 2.556 | **Yes (0.024)** |
| Davies–B | 0.276 | 0.016 | 0.286 | 0.011 | -0.515 | No (0.612) |
| ICS | 97,329 | 44,472 | 54 | 16 | 2.187 | No (0.051) |

SSE, SSE/DIM, and ICS all favour snippets. Although only SSE shows a significant difference, it is worth noting that ICS is very close to being significant as well. Dunn indexing shows a significant advantage for full texts, and Davies–Bouldin indexing, too, prefers full texts, without a significant difference.



## 4.3 Stemmer usage

Stemming usage is less important than the two above questions, but still worth testing on. Table 4 contains the test data for this comparison, $p > .05$ for a two-tailed independent t-test.

Table 4: Results by stemmer usage

| | Stemming | | No stemming | | | |
|---|---|---|---|---|---|---|
| | Mean | Std err | Mean | Std err | t | Significant |
| SSE | 202,883 | 123,851 | 210,013 | 129,670 | -0.400 | No (0.969) |
| SSE/DIM | 44.1 | 20.0 | 39.2 | 17.7 | 0.186 | No (0.854) |
| Dunn | 0.096 | 0.012 | 0.087 | 0.008 | 0.631 | No (0.535) |
| Davies–B | 0.284 | 0.015 | 0.279 | 0.012 | 0.214 | No (0.833) |
| ICS | 48,607 | 33,358 | 48,776 | 35,988 | -0.003 | No (0.997) |

Using a stemmer yielded slightly better results when evaluated by SSE, Dunn index, or ICS. Not using a stemmer gave slightly better results for SSE/DIM and Davies–Bouldin index. No evaluation metric, however, found any significant differences between using and not using a stemmer.

## 4.4 Stop words usage

As with stemming, knowing whether to apply stop words is a minor question, but one which is interesting nonetheless. Table 5 contains the test data for this comparison, $p > .05$ for a two-tailed independent t-test.



Table 5: Results by stop words usage

| | Stop words | | No stop words | | | |
|---|---|---|---|---|---|---|
| | Mean | Std err | Mean | Std err | t | Significant |
| SSE | 190,848 | 115,196 | 222,048 | 137,263 | -0.174 | No (0.863) |
| SSE/DIM | 39.0 | 16.7 | 44.3 | 20.8 | -0.200 | No (0.844) |
| Dunn | 0.086 | 0.007 | 0.093 | 0.012 | -0.228 | No (0.822) |
| Davies–B | 0.273 | 0.013 | 0.289 | 0.014 | -0.823 | No (0.419) |
| ICS | 30,785 | 18,890 | 66,598 | 44,640 | -0.739 | No (0.468) |

Not using stop words gave better results when measured by Dunn index, but all other metrics preferred usage of stop words. No evaluation metric, however, found any significant differences between using and not using stop words.

# 5 Discussion

With 32 corpora[12], three clustering algorithms, and four evaluation metrics, it is hardly surprising to find that not all data are in clear agreement.

To reiterate, the two main and two secondary goals of the thesis are

1. to choose one clustering algorithm over the others,
2. to decide whether to use snippeting or full texts in the *Webbklustring* project,
3. to find out if stop words should be applied, and
4. to recommend that stemming be used or not.

The discussion section will focus on the four questions asked in this thesis. While the results do not offer clear-cut answers, they do indicate what could be included in – and excluded from – further research on the same topic, and they are sufficient to provide the project with the information it requires.

## 5.1 Choosing an algorithm

Considering how the choice of algorithm is one of the two main goals, and arguably the most important one, at that, results related to this goal are analysed first. The matter of removing k-means from prospective inclusion in the final project is discussed, as is the remaining options of AHC and bisecting k-means.



### 5.1.1 Disregarding k-means

The most obviously noticable statistic in the results is that k-means perform far worse than the other two when measured by SSE and SSE/DIM. This trend continues in the ICS results, where, once again, k-means performs significantly worse.

SSE, SSE/DIM, and ICS, as discussed above, all measure cohesion by adding up distances in different ways – SSE by just adding the totals, and ICS by calculating averages. In other words: k-means, when applied to the corpora in this thesis, yield large clusters that are not coherent – documents in each clusters differ a lot. It is not the cause of a few rogue results, either –

[12] See 3.3.2

when compared individually, as opposed to as averages, every single k-means clustering resulted in worse SSE and SSE/DIM scores than any bisecting or AHC clustering. For ICS, 48 out of 64 non-k-means scores were better than the best achieved by k-means.

Whereas the above metrics indicate how large errors in the clusters are, the Dunn and Davies–Bouldin indices are ratios between cluster cohesion and inter-cluster distances. Here, k-means does a lot better, as shown in the results: it's neither better nor worse according to Dunn, and it's significantly worse than bisecting k-means yet also significantly better than AHC when measured by Davies–Bouldin.

With all of the above conclusions in mind, it is safe to say that the k-means algorithm cannot be legitimately considered for the *Webbklustring* application. It does, after all, produce clusters that are far worse off than its compared competitors, according to standard SSE and ICS metrics.

### 5.1.2 AHC or bisecting k-means

While it's easy to disregard k-means, as argued above, making a choice between agglomerative hierarchical clustering and bisecting k-means is harder. Below is a summary of the results between them.

Table 6: Comparison between AHC and bisecting k-means

| | Best performer | Significantly? |
|---|---|---|
| SSE | Bisecting | No |
| SSE/DIM | Bisecting | No |
| ICS | AHC | No |
| Dunn | AHC | No |
| Davies–Bouldin | Bisecting | Yes |



In short, the only significant difference is in Davies–Bouldin performance. Though AHC did better when measured by ICS and Dunn indexing, and bisecting k-means outperformed AHC for the two SSE metrics, neither of these results were significant.

Either algorithm appears sufficient for the application meant to use them. Although their scores clearly differ, it is not possible to say with certainty

that one outperforms the other. Hence, the results are not conclusive, but do provide some assistance in the choice of algorithm.

Keeping in mind that no error-measuring metric shows significant results, and that neither algorithm does better than the other at all three, it can be concluded that AHC and bisecting k-means both produce cohesive clusters.

Further, that AHC is significantly worse when measured by Davies–Bouldin but slightly better according to Dunn implies higher dissimilarity between clusters at the cost of cohesiveness. In contrast, bisecting k-means, by the same reasoning, may have better intra-cluster similarity but make more small errors when assigning documents to clusters. This is not surprising, given that AHC iterates over the least extreme values while bisecting k-means tries to make a smooth split between elements in a larger cluster.

As it is the goal of this thesis to suggest one algorithm over the others, and bisecting k-means does significantly better by at least one measure, this algorithm will be proposed for the *Webbklustring* application. It is also likely more beneficial to the aims of the project (see 5.4). This choice would certainly be facilitated by further research, focusing e.g. on external evaluation, on time and cpu consumption, etc.

## 5.2 Are snippets good enough?

Evaluating snippeting can be more difficult than comparing algorithms, because the results are no longer computed from the same data. Snippet-based corpora typically contain far fewer dimensions, as previews tend to be a lot shorter than full web pages, and the number of words is thus diminished. In practice, however, this should not affect the t-test comparisons.

To reiterate, here is a summary of the text source results:



Table 7: Comparison between full texts and snippets

| | Best performer | Significantly? |
|---|---|---|
| SSE | Snippets | Yes |
| SSE/DIM | Snippets | No |
| Dunn | Full texts | Yes |
| Davies–Bouldin | Full texts | No |
| ICS | Snippets | No |

Based on simple enumeration of performances, it appears that the "more than 40 %" performance degradation Mecca et al. got from snippets in 2007 (by using the external F-measure metric) [18] is nowhere to be found here, which bodes well for the snippets. After all, it goes without saying that the time requirement gains from needing to amass up to a hundred fewer pages can certainly be a more than acceptable trade-off for many users of any web service. Giving percentage scores is, however, not readily achievable based on these evaluations, as that would require an external measuring metric without which there is no comparison point to match against.

It is worth looking closer at what, exactly, the alternatives are good at. Snippets do well when it comes to minimising the error rates, while full text clusters seem to be more well formed. Neither result is a surprise, really, considering that the difference in dimensions make for vastly different error rates, and also – mostly – for relatively more compact clusters and longer inter-cluster distances, as uncommon words occur in more texts. When viewed that way, it becomes apparent that it is not simply a matter of stating whether snippets are generally good enough – they excel at other things than do full texts. If anything, projects wishing to emphasise cohesion may want to try snippeting first, whereas projects focusing on cluster separation might have better luck with full texts.

In summary, snippets produce decent clusters, but it's impossible to say whether any cluster – snippet-based or otherwise – is actually correct without external evaluation as well. Without in-application testing and evaluation, using snippets instead of full texts cannot be assumed to be the superior option, based on clustering error alone – but neither can full texts. They both appear to be viable alternatives. If possible, the best recommendation may be to try out both and see what the end users think.



### 5.3 Stop words and stemmer

These were the least differing results, by far. Out of ten comparisons, no significant result was detected – in fact, the result which came closest to the $p > .05$ boundary still had a 0.419 significance. This is illustrated in Table 8.

Table 8: Comparison of stemming and stop word results

| | Stemmer | | Stop words | |
|---|---|---|---|---|
| | Best when | Significantly? | Best when | Significantly? |
| SSE | Used | No | Used | No |
| SSE/DIM | Not used | No | Used | No |
| Dunn | Used | No | Not used | No |
| Davies–Bouldin | Not used | No | Used | No |
| ICS | Used | No | Used | No |

So, in essence, no recommendation regarding neither stop word lists nor stemmers can be offered, given the results. Considering the advantages and drawbacks of each metric does not help, either, since there are no significant results – it simply seems that these two variables don't affect the clustering by much. This is not necessarily bad news in itself; needing to distinguish fewer choices might actually be helpful. It will give the *Webbklustring* project options to choose from, and the knowledge that both options appear to work equally well.

## 5.4 Final recommendations

The goals of the *Webbklustring* project are to identify repeat information, to summarise multiple documents, and to compute search result legibility – the latter of which has very little do to with clustering.

Finding duplicates and summarising clusters are both tasks that favour cohesiveness over separation. The tighter the cluster, the more likely it is that information is repeated within it, and the more representative summarisations become. Distance between clusters becomes less meaningful as intra-cluster dissimilarity increases, as that implies that the information contained within clusters grows less similar.



As such, by the arguments in the above discussions on each individual variable, this thesis recommends that *Webbklustring* use bisecting k-means as its clustering algorithm, but it must be stressed that the difference from AHC does not appear to be large. Similarly, snippeting is recommended based on it performing better as measured by cohesion-rewarding metrics. The fact that it requires far fewer resources and pageloads is a huge additional bonus.

Whether to use a stemmer and/or stop words seems not to matter much

from a cluster quality standpoint. That said, using both is recommended as doing so may drastically reduce the number of dimensions involved, and thus improve execution time and memory consumption.

## 5.5 Further research

Clustering, while quite feasible in itself, clearly has plenty of room for improvement. Approaches to do so generally include some kind of heuristic such as web graphing, according to the principle that pages on the same topic tend to link to each other. Results are inconclusive [1], but also taxed with longer loading times.

Other notions include the one that meta data could be used to improve clustering [4], or that semantic knowledge such as the sense relation derived from *WordNet* can be employed in the same way [11]. Manually applied tags could also facilitate clustering [21]. It is obvious that there can be multiple approaches to the problem, and that, as they all have distinct advantages and disadvantages, they may be usable in different contexts.

Finally, there is also the vast problem that is labelling, which could certainly be researched upon in conjunction with the results of this thesis.

# References

[1] Ron Bekkerman, Shlomo Zilberstein, and James Allan. Web page clustering using heuristic search in the web graph. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2719–2724, 2007.

[2] Marcel Brun, Chao Sima, Jianping Hua, James Lowey, Brent Carroll, Edward Suh, and Edward R Dougherty. Model-based evaluation of clustering validation measures. *Pattern Recognition*, 40(3):807–824, 2007.

[3] Claudio Carpineto, Stanislaw Osiński, Giovanni Romano, and Dawid Weiss. A survey of web clustering engines. *ACM Computing Surveys (CSUR)*, 41(3):17, 2009.

[4] Claudio Carpineto and Giovanni Romano. Optimal meta search results clustering. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 170–177. ACM, 2010.

[5] Hung Chim and Xiaotie Deng. A new suffix tree similarity measure for document clustering. In *Proceedings of the 16th international conference on World Wide Web*, pages 121–130. ACM, 2007.

[6] Wisam Dakka and Panagiotis G Ipeirotis. Automatic extraction of useful facet hierarchies from text databases. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 466–475. IEEE, 2008.

[7] David L Davies and Donald W Bouldin. A cluster separation measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (2):224–227, 1979.

[8] Scott Deerwester, Susan T. Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[9] Joseph C Dunn. Well-separated clusters and optimal fuzzy partitions. *Journal of cybernetics*, 4(1):95–104, 1974.

[10] Carl-Oscar Erneholm. Multi-document summaries of swedish documents as search result. Master's thesis, KTH Royal Institute of Technology, 2012.

[11] Reza Hemayati, Weiyi Meng, and Clement Yu. Semantic-based grouping of search engine results using wordnet. *Advances in Data and Web Management*, pages 678–686, 2007.

[12] Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.

[13] Dan Jurafsky and James H Martin. *Speech & Language Processing*. Pearson Education India, second edition, 2000.

[14] Mika Käki. Findex: search result categories help users when document ranking fails. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 131–140. ACM, 2005.

[15] Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. Boilerplate detection using shallow text features. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 441–450. ACM, 2010.

[16] Ferenc Kovács, Csaba Legány, and Attila Babos. Cluster validity measurement techniques. In *6th International Symposium of Hungarian Researchers on Computational Intelligence*. Citeseer, 2005.

[17] Bing Liu. *Web data mining: exploring hyperlinks, contents, and usage data*. Springer Verlag, 2007.

[18] Giansalvatore Mecca, Salvatore Raunich, and Alessandro Pappalardo. A new algorithm for clustering search results. *Data & Knowledge Engineering*, 62(3):504–522, 2007.

[19] Stanislaw Osinski and Dawid Weiss. A concept-driven algorithm for clustering search results. *Intelligent Systems, IEEE*, 20(3):48–54, 2005.

[20] Martin Porter. Snowball: A language for stemming algorithms. `http://www.snowball.tartarus.org/texts/introduction.html`, 2001. Online; accessed 20-January-2013.

[21] Daniel Ramage, Paul Heymann, Christopher D Manning, and Hector Garcia-Molina. Clustering the tagged web. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 54–63. ACM, 2009.

[22] Magnus Rosell. *Text Clustering Exploration: Swedish Text Representation and Clustering Results Unraveled*. PhD thesis, KTH Royal Institute of Technology, 2009.

[23] Magnus Sahlgren. An introduction to random indexing. In *Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE*, volume 5, 2005.

[24] Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[25] Ugo Scaiella, Paolo Ferragina, Andrea Marino, and Massimiliano Ciaramita. Topical clustering of search results. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 223–232. ACM, 2012.

[26] Per-Anders Staav. Text clustering with random indexing. Master's thesis, KTH Royal Institute of Technology, 2009.

[27] Michael Steinbach, George Karypis, Vipin Kumar, et al. A comparison of document clustering techniques. In *KDD workshop on text mining*, volume 400, pages 525–526. Boston, 2000.

[28] Hiroyuki Toda and Ryoji Kataoka. A clustering method for news articles retrieval system. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 988–989. ACM, 2005.

[29] Ellen M Voorhees. The cluster hypothesis revisited. In *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 188–196. ACM, 1985.

[30] Miao Wan, Arne Jönsson, Cong Wang, Lixiang Li, and Yixian Yang. Web user clustering and web prefetching using random indexing with weight functions. *Knowledge and information systems*, 33(1):89–115, 2011.

[31] Oren Zamir and Oren Etzioni. Web document clustering: A feasibility demonstration. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 46–54. ACM, 1998.

[32] Hua-Jun Zeng, Qi-Cai He, Zheng Chen, Wei-Ying Ma, and Jinwen Ma. Learning to cluster web search results. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 210–217. ACM, 2004.

# Appendices

## A List of stopwords

The following is a list of stop words in Swedish, which is based on word frequencies and has been used for previous research by Christian Smith at Linköping University's Dept. of Computer and Information Science.

| | | | |
|---|---|---|---|
| alla | allt | att | av |
| blev | bli | blir | blivit |
| de | dem | den | denna |
| deras | dess | dessa | det |
| detta | dig | din | dina |
| ditt | du | där | då |
| efter | ej | eller | en |
| er | era | ert | ett |
| från | för | ha | hade |
| han | hans | har | henne |
| hennes | hon | honom | hur |
| här | i | icke | ingen |
| inom | inte | jag | ju |
| kan | kunde | man | med |
| mellan | men | mig | min |
| mina | mitt | mot | mycket |
| ni | nu | när | någon |
| något | några | och | om |
| oss | på | samma | sedan |
| sig | sin | sina | sitta |
| själv | skulle | som | så |
| sådan | sådana | sådant | till |
| under | upp | ut | utan |
| vad | var | vara | varför |
| varit | varje | vars | vart |
| vem | vi | vid | vilka |
| vilkas | vilken | vilket | vår |
| våra | vårt | än | är |
| åt | över | | |

# B Average results

These are the averages, on which statistical tests were then performed, as explained in the Results section.

Because of a lack of space, some abbreviations are used. The *Type* column refers to whether the cluster consists of full web pages or snippets. The first value of the S/S column corresponds to whether stemming was used, and the second to whether stop words were. Both are binary Y/N variables. D-B stands for Davies–Bouldin index.

All evaluation methods aim for scores as close to zero as possible, except Dunn, which is better the higher it is.

| Type | Algorithm | S/S | SSE | SSE/DIM | Dunn | D-B | ICS |
|---|---|---|---|---|---|---|---|
| Full | AHC | Y/Y | 25,965 | 5.576 | 0.099 | 0.330 | 2,999 |
| Full | AHC | Y/N | 8,970 | 1.790 | 0.090 | 0.373 | 114 |
| Full | AHC | N/Y | 11,053 | 2.033 | 0.119 | 0.339 | 130 |
| Full | AHC | N/N | 9,412 | 1.644 | 0.097 | 0.348 | 142 |
| Full | Bisecting | Y/Y | 14,391 | 2.954 | 0.094 | 0.207 | 23,834 |
| Full | Bisecting | Y/N | 7,912 | 1.[illegible]92 | 0.064 | 0.243 | 407 |
| Full | Bisecting | N/Y | 8,011 | 1.425 | 0.060 | 0.242 | 2,103 |
| Full | Bisecting | N/N | 5,523 | 0.920 | 0.070 | 0.230 | 3,799 |
| Full | k-means | Y/Y | 1,010,254 | 161.793 | 0.134 | 0.251 | 176,973 |
| Full | k-means | Y/N | 1,220,0[illegible] | 200.678 | 0.208 | 0.238 | 378,597 |
| Full | k-means | N/Y | 1,0[illegible]665 | 140.667 | 0.108 | 0.248 | 163,020 |
| Full | k-means | N/N | 1,256,897 | 176.283 | 0.142 | 0.267 | 415,828 |
| Snip | AHC | Y/Y | 5,417 | 6.092 | 0.098 | 0.349 | 10 |
| Snip | AHC | Y/N | 3,326 | 3.277 | 0.073 | 0.320 | 3 |
| Snip | AHC | N/Y | 4,585 | 4.620 | 0.100 | 0.266 | 7 |
| Snip | AHC | N/N | 3,480 | 3.187 | 0.069 | 0.331 | 5 |
| Snip | Bisecting | Y/Y | 4,893 | 5.509 | 0.079 | 0.231 | 95 |
| Snip | Bisecting | Y/N | 3,139 | 3.091 | 0.084 | 0.256 | 12 |
| Snip | Bisecting | N/Y | 4,443 | 4.476 | 0.068 | 0.236 | 11 |
| Snip | Bisecting | N/N | 3,216 | 2.945 | 0.077 | 0.243 | 20 |
| Snip | k-means | Y/Y | 59,881 | 67.291 | 0.057 | 0.293 | 139 |
| Snip | k-means | Y/N | 70,439 | 69.877 | 0.068 | 0.310 | 102 |
| Snip | k-means | N/Y | 64,621 | 65.403 | 0.060 | 0.290 | 96 |
| Snip | k-means | N/N | 72,256 | 66.447 | 0.071 | 0.313 | 149 |

# C Full results

These are the full results for all 96 clusters, as explained in the Results section.

The cluster field in the below tables has been abbreviated for formatting purposes. It should be read as follows:

1. Type: S for snippet / F for full text
2. Clustering method: K for k-means / A for AHC / B for bisecting
3. Query: I for *ibuprofen* / K for *katter* / J for *jaguar* / R for *hur startade ryska revolutionen?*
4. Stemmer: Y or N depending on whether a stemmer was used
5. Stop words: Y or N depending on the stop words list was used

## C.1 Sorted by SSE

Data have been rounded to the nearest integer.

| Cluster | Result | Cluster | Result | Cluster | Result |
|---|---|---|---|---|---|
| S B I Y N | 2145 | F A R N Y | 4788 | S K K Y Y | 55,734 |
| S A I Y N | 2256 | S A K Y N | 4898 | S K I Y Y | 57,990 |
| S B I N N | 2339 | S A R Y Y | 4969 | S K R N Y | 60,771 |
| S A I N N | 2524 | S A K N N | 4993 | S K I N Y | 61,936 |
| S B J N N | 2547 | F B J Y N | 5042 | S K K N Y | 62,772 |
| S B I N Y | 2641 | S B J Y Y | 5073 | S K R Y N | 65,644 |
| S B J Y N | 2843 | F B K N Y | 5085 | S K I N N | 66,537 |
| S A I N Y | 2852 | F A K N Y | 5210 | S K I Y N | 68,009 |
| S A J Y N | 2869 | F A J Y N | 5256 | S K K Y N | 69,291 |
| S A J N N | 2935 | F B J N N | 5280 | S K R N N | 69,824 |
| F B R Y N | 3077 | F A J N N | 5695 | S K J Y Y | 70,497 |
| S B R Y N | 3084 | S B K Y Y | 5766 | S K K N N | 71,528 |
| S B R N N | 3280 | F B K Y Y | 5996 | S K J N Y | 73,004 |
| S A R Y N | 3282 | S A J Y Y | 6087 | F A I Y Y | 78,699 |
| F B K Y N | 3386 | F B J N Y | 6203 | S K J Y N | 78,811 |
| S A R N N | 3468 | S A K Y Y | 6239 | S K J N N | 81,135 |
| S B J N Y | 3597 | F A K Y Y | 6,347 | F K J Y Y | 570,786 |
| F B R N N | 3611 | F A J N Y | 6,456 | F K J Y N | 586,098 |
| F A K Y N | 3679 | S B K N Y | 6,915 | F K J N Y | 635,765 |
| S A J N Y | 3803 | S A K N Y | 7,050 | F K J N N | 731,857 |
| F B K N N | 3814 | F B R Y Y | 7,170 | F K I N Y | 749,195 |
| S B I Y Y | 3974 | F A R Y Y | 7,585 | F K I Y Y | 787,190 |
| F A R Y N | 4008 | F B I Y N | 9,386 | F K K Y Y | 791,764 |
| F A K N N | 4047 | F B I Y Y | 11,115 | F K K N Y | 829,413 |
| S A I Y Y | 4375 | F A J Y Y | 11,228 | F K I Y N | 918,239 |
| S B K Y N | 4483 | F B I N Y | 16,175 | F K I N N | 1,022,403 |
| F A R N N | 4512 | F B I Y N | 20,142 | F K K N N | 1,319,628 |
| F B R N Y | 4581 | F A I Y N | 22,935 | F K K Y N | 1,356,757 |
| S B R N Y | 4598 | F A I N N | 23,343 | F K R Y Y | 1,891,277 |
| S B K N N | 4698 | F A I N Y | 27,757 | F K R N N | 1,953,701 |
| S A R N Y | 4735 | F B I Y Y | 33,282 | F K R Y N | 2,018,940 |
| S B R Y Y | 4760 | S K R Y Y | 55,303 | F K R N Y | 2,092,288 |

## C.2 Sorted by SSE/DIM

Data have been rounded to the nearest three decimals.

| Cluster | Result | Cluster | Result | Cluster | Result |
|---|---|---|---|---|---|
| F B R N N | 0.211 | S A J J N | 2.661 | S K R N J | 54.163 |
| F B R J N | 0.245 | S A I N N | 2.697 | S K R J J | 56.489 |
| F A R N N | 0.266 | S B R J N | 2.761 | S K R N N | 56.492 |
| F B R N J | 0.271 | S A R N N | 2.806 | S K R J N | 58.768 |
| F A R N J | 0.284 | S A R J N | 2.938 | S K K N J | 66.285 |
| F A R J N | 0.319 | S B I N J | 3.046 | S K K N N | 66.786 |
| F B R J J | 0.582 | F B I N J | 3.284 | S K K J J | 67.392 |
| F A R J J | 0.616 | S A I N J | 3.289 | S K K J N | 69.639 |
| F B K N N | 0.793 | S B J N J | 3.436 | S K J N J | 69.727 |
| F A K N N | 0.841 | S A J N J | 3.633 | S K I N N | 71.087 |
| F B K J N | 0.843 | S B R N J | 4.098 | S K J N N | 71.421 |
| F B J N N | 0.847 | S A R N J | 4.220 | S K I N J | 71.438 |
| F B J J N | 0.898 | F B I J N | 4.382 | S K J J J | 72.156 |
| F A J N N | 0.913 | S B K N N | 4.387 | S K J J N | 73.108 |
| F A K J N | 0.916 | S B K J N | 4.505 | S K I J J | 73.127 |
| F A J J N | 0.936 | F A I N N | 4.554 | S K I J N | 77.992 |
| F B J N J | 1.042 | S A K N N | 4.662 | F K J J N | 104.344 |
| F A J N J | 1.084 | S B R J J | 4.862 | F K J N J | 106.762 |
| F B K N J | 1.102 | S A K J N | 4.913 | F K J J J | 107.009 |
| F A K N J | 1.129 | F A I J N | 4.989 | F K R N N | 114.025 |
| F B K J J | 1.570 | S B I J J | 5.011 | F K J N N | 117.360 |
| F A K J J | 1.662 | S A R J J | 5.076 | F K R N J | 123.987 |
| F B I N N | 1.831 | S B J N J | 5.192 | F K I N J | 152.121 |
| F B J J J | 2.084 | S A J J J | 5.517 | F K R J J | 153.513 |
| F A J J J | 2.105 | F A I N J | 5.636 | F K R J N | 160.782 |
| S B J N N | 2.242 | S A J J J | 6.230 | F K I J J | 179.274 |
| S B I J N | 2.4[illegible] | S B K J J | 6.972 | F K K N J | 179.799 |
| S B I N N | 2.499 | S B K N J | 7.323 | F K I N N | 199.454 |
| S A J N N | 2.583 | S A K N J | 7.338 | F K I J N | 199.747 |
| S A I J N | 2.587 | S A K J J | 7.544 | F K K J J | 207.377 |
| S B J J N | 2.638 | F B I J J | 7.580 | F K K N N | 274.294 |
| S B R N N | 2.654 | F A I J J | 17.923 | F K K J N | 337.838 |

## C.3 Sorted by ICS

Data have been rounded to the nearest integer.

| Cluster | Result | Cluster | Result | Cluster | Result |
|---|---|---|---|---|---|
| F K K N N | 1,252,675 | S K K J N | 142 | S B I J J | 16 |
| F K K J N | 1,246,841 | S K K N N | 135 | S B R J J | 15 |
| F K K N J | 371,434 | S K I N J | 127 | F B K N J | 14 |
| F K K J J | 354,956 | S B K J J | 122 | S A K N N | 14 |
| F K R N N | 252,555 | S K J J N | 113 | S A J J J | 14 |
| F K R J J | 197,617 | S K K N J | 109 | S B R N N | 13 |
| F K R N J | 130,272 | S K J J J | 107 | S B K N J | 13 |
| F K J J J | 110,732 | S K J N J | 96 | F A R J N | 12 |
| F K J N N | 106,915 | S K I J N | 90 | F A K J J | 12 |
| F K J J N | 106,367 | S K J N N | 88 | S B J N J | 12 |
| F K J N J | 105,431 | S K R N N | 73 | F A R N J | 11 |
| F K R J N | 103,101 | F B J N N | 69 | F A J N N | 11 |
| F B I J J | 95,102 | S K R J N | 64 | S B R N J | 11 |
| F K I J N | 58,078 | S K R J J | 62 | S A K N J | 11 |
| F K I N N | 51,169 | F A J J J | 54 | S B R J N | 9 |
| F K I N J | 44,944 | S K R N J | 51 | S B I N J | 7 |
| F K I J J | 44,587 | F B J J J | 47 | F A K J N | 6 |
| F B I N N | 14,945 | F B K J J | 38 | S B I N N | 6 |
| F A I J J | 11,742 | S B K J N | 34 | F A K N N | 5 |
| F B I N J | 8,343 | S B K N N | 33 | S B I J N | 4 |
| F B I J N | 1,422 | F B R N J | 32 | S A I J N | 4 |
| F A I N N | 534 | S B R J N | 27 | S A K J N | 4 |
| F A I N J | 464 | F B R J N | 23 | S A I J J | 4 |
| F A I J N | 418 | F B J N J | 22 | S A R N N | 3 |
| S K I N N | 299 | F A J N J | 22 | S A R J J | 3 |
| S B J J J | 235 | F B K J N | 21 | S A R J N | 2 |
| S K K J J | 205 | F A J J N | 20 | S A J N J | 2 |
| F A R J J | 187 | F A K N J | 20 | S A J N N | 1 |
| S K I J J | 180 | S A K J J | 20 | S B J J N | 1 |
| F B R N N | 168 | F A R N N | 18 | S A J J N | 1 |
| F B R J N | 163 | F B K N N | 17 | S A I N N | 1 |
| F B R J J | 150 | S A R N J | 17 | S A I N J | 0 |

## C.4 Sorted by Dunn index

Data have been rounded to the nearest three decimals.

| Cluster | Result | Cluster | Result | Cluster | Result |
|---|---|---|---|---|---|
| F K K J N | 0.364 | S B I J J | 0.090 | S K R J J | 0.069 |
| F K J J J | 0.302 | F A I J J | 0.088 | S B J N J | 0.067 |
| F K K N N | 0.300 | F K R J J | 0.086 | S B I N N | 0.067 |
| F K J J N | 0.270 | S B R N J | 0.086 | S B I J N | 0.067 |
| S A K N J | 0.248 | F A K N N | 0.086 | S K K N N | 0.066 |
| S A K J J | 0.176 | S K R J N | 0.085 | S A R N J | 0.066 |
| F B R J J | 0.170 | S B R J J | 0.083 | F B J N J | 0.065 |
| F A J N J | 0.161 | F A R J J | 0.081 | F K I N N | 0.064 |
| S B R J N | 0.150 | F A R J N | 0.081 | S K R N N | 0.064 |
| F A K N J | 0.140 | F A K J N | 0.081 | S K R N J | 0.064 |
| F A K J J | 0.128 | S A I N N | 0.079 | F B K N J | 0.062 |
| F A J N N | 0.125 | F B J N N | 0.077 | S K I N J | 0.059 |
| F K K N J | 0.124 | S B K N N | 0.077 | F B I J J | 0.058 |
| S K I N N | 0.116 | F B R N N | 0.076 | F K I J J | 0.057 |
| S A R J J | 0.114 | F A R N J | 0.076 | F B K J N | 0.056 |
| F K I N J | 0.112 | S B K J J | 0.075 | S A J J J | 0.056 |
| F A I N N | 0.109 | F B J J J | 0.075 | S K I J J | 0.054 |
| F A I J N | 0.109 | S K K J J | 0.074 | S A K N N | 0.054 |
| F K J N N | 0.107 | F B K J J | 0.074 | F B K N N | 0.053 |
| F K I J N | 0.107 | S K K J N | 0.073 | S A J N J | 0.052 |
| F K J N J | 0.103 | S B R N N | 0.073 | S A J J N | 0.052 |
| F A I N J | 0.101 | F B I N J | 0.072 | S B J J N | 0.047 |
| F B R J N | 0.101 | S K J J N | 0.072 | S A J N N | 0.046 |
| F A J J J | 0.099 | S K I N J | 0.072 | S K K N J | 0.044 |
| S A R N N | 0.098 | S B I N J | 0.072 | S B K N J | 0.044 |
| F K R N N | 0.097 | S A K J N | 0.072 | S A I J J | 0.044 |
| S A R J N | 0.093 | S B K J N | 0.071 | S K I J N | 0.043 |
| S B J N N | 0.093 | S A I J N | 0.071 | F B R N J | 0.043 |
| F K R N J | 0.092 | F B I N J | 0.070 | S K J N N | 0.036 |
| F K R J N | 0.091 | F B I J N | 0.070 | S A I N J | 0.034 |
| F A J J N | 0.091 | S B J J J | 0.070 | S K J J J | 0.029 |
| F K K J J | 0.090 | F A R N N | 0.070 | F B J J N | 0.028 |

## C.5 Sorted by Davies–Bouldin index

Data have been rounded to the nearest three decimals.

| Cluster | Result | Cluster | Result | Cluster | Result |
|---|---|---|---|---|---|
| F B J J J | 0.164 | F B R N J | 0.251 | S K R N N | 0.307 |
| F B K J J | 0.166 | S A K J N | 0.254 | F A R N N | 0.308 |
| F B I J J | 0.166 | S K K N N | 0.254 | S K K J N | 0.309 |
| F K I J N | 0.177 | S B K N J | 0.255 | S K I N J | 0.310 |
| S A J N J | 0.177 | S B R N N | 0.258 | F K R J N | 0.318 |
| F B J N N | 0.182 | S B J J N | 0.258 | S K J J N | 0.318 |
| F B I J N | 0.183 | F B K N N | 0.260 | S K R J N | 0.325 |
| S B J J J | 0.185 | S A K N N | 0.267 | F A R J J | 0.325 |
| F K I N J | 0.188 | F B K J N | 0.274 | F K J N N | 0.329 |
| S B J N J | 0.188 | S A J N N | 0.276 | S A J J J | 0.329 |
| F B I N N | 0.189 | S B R N J | 0.277 | F B R J J | 0.332 |
| F B J N J | 0.198 | S K J N J | 0.278 | F A R J N | 0.337 |
| F K I N N | 0.201 | S K R N J | 0.278 | S K J N N | 0.338 |
| S B I J N | 0.203 | S B I J J | 0.286 | F A I J J | 0.339 |
| F K K N J | 0.208 | S A J J N | 0.286 | S B R J N | 0.340 |
| F K J J N | 0.212 | S K I J J | 0.288 | S A I J J | 0.341 |
| F A R N J | 0.212 | F A K J J | 0.290 | S A K N J | 0.343 |
| S B K J J | 0.216 | F B R N N | 0.290 | F A K N J | 0.344 |
| F B J J N | 0.217 | S K K J J | 0.290 | S A I J N | 0.348 |
| F B I N J | 0.222 | S K R J J | 0.291 | S K I N N | 0.353 |
| S B K J N | 0.223 | S K I J N | 0.291 | S A K J J | 0.361 |
| S B I N J | 0.225 | S K K N J | 0.292 | S A R J J | 0.364 |
| F K I J J | 0.225 | F K R N J | 0.293 | F A I N N | 0.364 |
| F K K N N | 0.234 | F K I N J | 0.295 | F A J J J | 0.367 |
| S B R J J | 0.236 | F A K J N | 0.295 | F A J N J | 0.378 |
| F K K J J | 0.237 | F B K N J | 0.296 | S A R N N | 0.388 |
| S B K N N | 0.2[illegible] | F B R J N | 0.297 | F A J J N | 0.389 |
| S B I N N | 0.237 | S A R N J | 0.298 | S A I N N | 0.392 |
| S B J N N | 0.240 | F K J N J | 0.300 | S A R J N | 0.393 |
| S A I N J | 0.244 | F K R N N | 0.302 | F A J N N | 0.418 |
| F K K J N | 0.246 | F A K N N | 0.302 | F A I N J | 0.423 |
| F K J J J | 0.249 | S K J J J | 0.303 | F A I J N | 0.474 |

LINKÖPINGS UNIVERSITET

**Avdelning, Institution**
Division, Department

IDA,
Dept. of Computer and Information Science
581 83 LINKÖPING

**Datum**
Date

2013-07-02

**Språk**
Language

☐ Svenska/Swedish
☒ Engelska/English

☐ _____

**Rapporttyp**
Report category

☐ Licentiatavhandling
☒ Examensarbete
☐ C-uppsats
☐ D-uppsats
☐ Övrig rapport
☐ _____

**ISBN**
—

**ISRN**
LIU-IDA/KOGVET-G–13/025–SE

**Serietitel och serienummer** **ISSN**
Title of series, numbering —

**URL för elektronisk version**
-

**Titel** Webbklustring
En jämförelse av klustringsmetoder på svenska

Title Clustering the Web
Comparing Clustering Methods in Swedish

**Författare** Joel Hinz
Author

**Sammanfattning**
Abstract

Clustering – automatically sorting – web search results has been the focus of much attention but is by no means a solved problem, and there is little previous work in Swedish. This thesis studies the performance of three clustering algorithms – k-means, agglomerative hierarchical clustering, and bisecting k-means – on a total of 32 corpora, as well as whether clustering web search previews, called snippets, instead of full texts can achieve reasonably decent results. Four internal evaluation metrics are used to assess the data. Results indicate that k-means performs worse than the other two algorithms, and that snippets may be good enough to use in an actual product, although there is ample opportunity for further research on both issues; however, results are inconclusive regarding bisecting k-means vis-à-vis agglomerative hierarchical clustering. Stop word and stemmer usage results are not significant, and appear to not affect the clustering by any considerable magnitude.



**Nyckelord**
Keywords clustering, web, search results, snippets, k-means, agglomerative hierarchical clustering, bisecting k-means, swedish



# Copyright

## Svenska



## English






Linköping, July 2, 2013