Search Result Clustering

Adrian Apap, Alan Falzon, Juan Buhagiar and Yurgen Schembri

Faculty of ICT, University of Malta, Msida
adrian.apap.14@um.edu.mt, alan.falzon.14@um.edu.mt, juan.buhagiar.14@um.edu.mt,
yurgen.schembri.14@um.edu.mt

Abstract

User queries in search engines may be ambiguous and have different senses. As a result, the search engine results may pertain to these different senses and may not all be relevant to the user.

An internet browser extension which clusters web search engine results was developed. Four different clustering algorithms, namely K-Means, No-K-Means, Clustering with a Self-Organising Map and Spectral Clustering were investigated. After evaluating the fitness of all, using the SemEval 2013 Task 11 gold standard collection, the former three were then applied to the add-on.

After the clustering occurs, each cluster is labelled, using a simple technique, with an extended search query which fits most results in the cluster. This is done in order to help the user articulate his query better, if necessary.

When testing the extension using Google search results, the three algorithms implemented in the extension returned satisfactory clusters, with No-K-Means giving better results. This was also reflected when all the algorithms were evaluated using the SemEval 2013 Task 11 gold standard.

1. Introduction

Search results clustering is the process of categorizing search results in meaningful groups. The clusters, containing a label which describes them, help the user to identify and search through the category which is in accordance with their interest, thus information is retrieved quicker by the user. This is particularly useful when a search query has multiple meanings. For example, the search query "jaguar" may refer to cats, cars, or maybe even something else. The search engine may return scattered results corresponding to these different meanings. Users are typically interested in only one meaning, and may not be able to express their needs well enough. By clustering the results and labelling them, the users should be able to identify the results relevant to their requirements more easily and may also further their search by altering their query to what is suggested by the cluster label.

The aim of this task is to develop a Google Chrome extension which performs search result clustering on results retrieved from the Google search engine, using different algorithms and to compare their results in order to determine the algorithm fittest for the application.

The browser extension was developed for Google Chrome as it is the most used internet browser [1] and operates on Google search results. When the user accesses the Google results page, the extension button becomes enabled and when clicked, the user chooses the preferred algorithm to cluster the results, through a drop-down menu. The clustered results are then presented to the user on the same page.

The unsupervised search result clustering algorithms operate on weighted term vectors. These are extracted from search result snippets, using the bag-of-words model, with the term frequency as the weight.

The three clustering algorithms, K-Means, No-K-Means and Clustering with a Self-Organising Map, were tested on Google search results, taking into consideration the clustering quality and the time of execution. These algorithms, along with another clustering algorithm, Spectral Clustering, were evaluated using the Gold Standard Dataset provided in SemEval 2013 Task 11 [2], where the accuracy is based on the obtained values of F1, Rand Index, Adjusted Rand Index and Jaccard Index.

A background of the algorithms and their implementations are discussed in sections 2 and 3, and The Google Chrome extension is discussed in section 4. In section 5 and 6, we discuss testing and evaluation, respectively, while we summarize our work distribution in section 8. In section 9, we conclude our task documentation.

2. Background

In this section, an overview or unsupervised clustering, cluster analysis and the algorithms discussed in this document is given.

2.1 Unsupervised Clustering

Unsupervised clustering algorithms are used to draw inferences from datasets which consist of data but are not labelled. In this project we will use cluster analysis, which is used to find patterns or groups from the data. The clusters are modelled using similarity metrics such as Euclidean and Cosine Distance. Since all the examples are unlabelled there is no evaluation function to identify potential solutions, hence one cannot utilise training. The algorithms described hereafter were used to try to find the best solutions for unlabelled data.

2.2 Cluster Validity Indices

In this project, two different indices are used as cluster validity indices; Dunn's Index and PBM Index, discussed below. These are both used to measure how correct the clusters seem to be without having any information about the contents of the clusters.

2.2.1 Dunn's Index

The aim of this metric is to identify sets of clusters that are compact, with small variance between data in clusters and separated well, meaning that clusters are as far away as possible. The Dunn's Index is used in the testing phase to identify the best number of clusters. The higher the Dunn's Index the better the clustering. [3]

The Dunn's Index is calculated using Eq. 5 below.

$$d_{kk'} = min_{i \in I_k \, j \in I_{k'}} \left\| M_i^{\{k\}} - M_j^{\{k'\}} \right\| \quad \text{Eq. 1}$$

$$d_{min} = min_{k \neq k'} d_{kk'} \quad \text{Eq. 2}$$

$$D_k = max_{i,j \in I_k \, i \neq j} \left\| M_i^{\{k\}} - M_j^{\{k\}} \right\| \quad \text{Eq. 3}$$

$$d_{max} = max_{1 \leq k \leq K} D_k \quad \text{Eq. 4}$$

$$C = \frac{d_{min}}{d_{max}} \quad \text{Eq. 5}$$

Where:

- $d_{kk'}$ is the distance between clusters C_k and $C_{k'}$, measured by their closest points, where K is the number of clusters.
- d_{min} is the smallest of the distances $d_{kk'}$.
- D_k is the largest distance separating two distinct points in the cluster, also known as the diameter of the cluster.
- d_{max} is the largest of the distances D_k.

2.2.2 PBM Index

The aim of this metric is to identify the best number of clusters to be returned by a clustering algorithm. The index is calculated by finding the distances between every data point to their centroid and the distances between the centroids themselves. This is also used in the evaluation section and compared the Dunn's Index. [3]

The PBM Index is calculated using Eq. 9 below.

$$D_B = \max_{k<k'} d(G^{\{k\}}, G^{\{k'\}}) \quad \text{Eq. 6}$$

$$E_W = \sum_{k=1}^{K} \sum_{i \in I_k} d(M_i, G^{\{k\}}) \quad \text{Eq. 7}$$

$$E_T = \sum_{i=1}^{N} d(M_i, G) \quad \text{Eq. 8}$$

$$C = \left(\frac{1}{K} * \frac{E_T}{E_W} * D_B \right)^2 \quad \text{Eq. 9}$$

Where:

- D_B is the largest distance d between two cluster barycenters (centroids), $G^{\{k\}}$. Where K is the number of clusters.
- E_W is the sum of distances of the points of each cluster to their barycentre.
- E_T is the sum of the distances of all the points to the barycentre G of the entire dataset.

2.3 K-Means

K-Means is one of the most common clustering algorithms. The main goal of K-Means is to cluster N vectors into K partitions, where $K \leq N$. K-Means tries to minimise the sum of distances of each vector to their respective cluster centroids, so that clusters are far apart but the vectors within each cluster are compact. [4]

2.3.1 Method

K-Means first starts with K non-equal points, which could be determined using various methods. These first K points are treated as the cluster centroids.

Each vector is then assigned to the nearest centroid, the distance between a vector and a cluster centroid can be calculated using different methods, for example by using Euclidean or Cosine distance metrics.

The centroids are then re-calculated and the membership of each vector is re-computed. This is done until the membership in the clusters is not changed in an iteration or a stopping condition is reached, as discussed in section 2.3.2. [4]

2.3.2 Limitations

K-Means keeps re-calculating the centroids and re-computing the membership of each vector until the clusters stop changing. This may take a lot of time so other stopping conditions are implemented, for example a maximum number of iterations is specified by the user.

The main problem of the K-Means function is how the initial centroids are chosen, this can either be the first K vectors or a random K vectors from the dataset. If the chosen initial centroids are vectors which are close to each other or one of the chosen vectors is a document which has no other vectors close to it, the K-Means algorithm will not be very effective when clustering. It is usually difficult to determine the initial centroids because no knowledge about the dataset is known before clustering.

Another common problem when using K-Means to cluster documents, is determining the value of K. To solve this problem other algorithms can be used to determine the best K value for a particular dataset. Examples of such algorithms are explained in section 2.2.1.

2.4 No-K-Means

The No-K-Means algorithm is a clustering algorithm inspired by the K-means algorithm. Similar to the K-Means algorithm, the aim of No-K-Means is to cluster vectors, where each cluster contains vectors with minimised distances between them, and all clusters centroids are as different as possible.

Unlike K-Means, No-K-Means doesn't need to have the resulting amount of clusters, K, specified beforehand, thus taking care of the problem regarding which initial centroids to choose for the clusters. [5]

2.4.1 Method

No-K-Means starts with an empty list of clusters, and then creates a new cluster for the first vector to be clustered, with the vector being the cluster centroid.

Every other vector is compared to every existing cluster. If the similarity between the vector and the most similar cluster is below than a specified threshold, then a new cluster is created with that vector as its centroid. Otherwise the vector is added to that cluster, and the cluster centroid is re-computed by calculating the average of the corresponding vector values of the cluster. [5]

2.4.2 Limitations

Despite the fact that No-K-means doesn't require the number of clusters to be specified, it still requires some threshold to be specified, for when deciding whether to create a new cluster or not.

The alteration of the cluster centroids when being re-computed may vary, as it depends on which order the vectors are inputted to the algorithm. This makes the results of No-K-Means dependent on the vector input order.

2.5 Self-Organising Maps

Self-Organising Maps are Neural Networks which, as the name implies, are able to manipulate their own structure when stimulated with an input. The network usually consists of a 2-dimensional array of units, each of which is assigned a vector of weights. The values of these units are altered depending on the different inputs during the training stage. The clustering process then makes use of these weights in order to collect the inputs. [6]

2.5.1 Method

The size of the input vectors is first taken as l. An n by m neural network is initialised, with each neuron having a vector of l weights with random values. A number of iterations k are defined to limit how long the training procedure takes.

For every iteration, all training inputs are fed to the network in sequence. For each input, the best matching unit, (BMU), (i.e. the neuron with weights having the least distance from the input) is calculated. The topological neighbourhood, N, of the BMU is calculated using a neighbourhood function. The weights of the BMU and the units in N are modified, using a learning function, to approach those of the input. Euclidean distance was used as the distance function. [6]

The clustering process takes place after the training is finished. A simple technique is adopted for the purpose of this assignment. To cluster the inputs, best matching unit for each is calculated, and the inputs with the same BMU are collected as a cluster.

2.6 Spectral Clustering

Spectral Clustering has been widely used in recent years and has become one of the most popular algorithms to use. This can be attributed to the simplicity to implement such a system along with results which usually perform better than other algorithms. Spectral clustering makes use of eigenvalues of a similarity matrix to perform dimensionality reduction. After which the resultant data will be clustered. [7]

2.6.1 Method

The first step to implement such a clustering algorithm is to create a similarity matrix. This is done by finding the similarities between all pairs of data points into a graph. In our implementation we are using an unnormalised graph Laplacian Matrix to represent this graph into a matrix. All the eigenvectors of the Laplacian Matrix are calculated and kept in another matrix. Finally the matrix with

eigenvectors is clustered using the K-Means algorithm. To find the best number of clusters for the K-Means algorithm we use metrics for evaluating the functions. The two metrics used are described above, Dunn's index and PBM index.

3. Clustering Implementation

In this section, the implementations of the algorithms along with certain limitations with respect to the Chrome extension developed are discussed.

3.1 K-Means

For the purpose of this task, a JavaScript implementation of the K-Means algorithm was found and modified to fit the needs of this task [8].

The K-Means used in the extension was evaluated using two methods to determine the best value of K. The two methods are the Dunn's index explained in section 2.2.1 and another method, where if the dataset contained more than a single Wikipedia search result, the Wikipedia search results were used as the initial centroids and K was changed to the amount of Wikipedia search results. It is assumed that Wikipedia typically contains a different page for each different sense of a term, hence one can assume the different Wikipedia search results to refer to different senses. However, the Dunn's index was not used in the extension since it was not feasible given that it greatly increases the clustering time. The method using Wikipedia search results produced better results when the algorithm was evaluated using the gold standard dataset.

The implemented K-Means was also tested using Euclidean distance and Cosine distance, where the cosine distance also produced better results when clustering sparse vectors.

The current implementation of K-Means currently uses cosine distance as a distance measure and the Wikipedia search results to choose the initial centroids. This can be clearly seen in Figure 7, where the algorithm found two Wikipedia search results, one on jaguar the car brand and the other Wikipedia search result on jaguar the animal. The algorithm set K to 2 and used the two Wikipedia search results as the initial centroids. If there is only one Wikipedia search result in the dataset, the initial centroids are chosen at random and the default value of K, 5, is used.

3.2 No-K-Means

For No-K-Means, the similarity between input vectors and cluster centroids was computed using the cosine similarity equation.

For the best results, a threshold of 0.05 was used when the algorithm gets to decide whether to create a new cluster or not. Since no cluster analysis methods are implemented in the extension, this threshold had to be fixed. This choice was based on a number of tests using results from the Google search engine. In addition, it was observed that the algorithm performed best with this threshold, as explained in section 6.4.

3.2.1 Limitations

In order to find the ideal threshold, cluster analysis could be performed, such as by using Dunn's Index. When using Dunn's Index with the K-Means algorithm, the time cost was expensive, hence it was not feasible to include it in the extension's No-K-Means implementation.

3.3 Self-Organising Maps

For the purpose of this task, a JavaScript implementation of a Self-Organising Map was found [9].

3.3.1 Limitations

Due to the expected response time, certain limitations to the quality of the results had to be imposed on the implementation.

Ideally, the parameters for the algorithm are not fixed, but determined using Dunn's or PBM Indices. This, however, is not feasible as it greatly increases the computation time for the algorithm.

After testing with several input arguments (mainly varying the number of iterations and the network size), the ideal values for these parameters were taken based on the computation time and the quality of the results. The choice of these values is discussed in section 6.5.

Another issue with the implementation is that the distance metric used is Euclidean. While Cosine distance was considered, and tested, this required more sophisticated clustering methods, such as using a U-Matrix [6]. These, however, were not found available in JavaScript, and the implementation was limited to a simple clustering technique.

3.4 Spectral Clustering

Due to the fact that libraries for the Spectral Clustering algorithm were not available in JavaScript it was decided to use MATLAB [10]. As a result, it was not feasible to include this algorithm with the Google Chrome extension.

3.4.1 Limitations

In the MATLAB Implementation a cosine similarity matrix was tested but did not work with our data

since it resulted in a singular matrix. This was attributed to the fact that our data was sparse.

4. The Google Chrome Extension

The developed Chrome extension aims to group the top hundred search results returned from the Google search engine when and if requested by the user. This should be helpful to distinguish between results corresponding to different senses of an ambiguous query phrase. For example, search results for "jaguar" may be referring to a car manufacturer, a type of cat, or other possibly meanings of the same word.

In this section, the development of the product is discussed and examples of its usage are given.

4.1 Development of the Artefact

The chrome extension is split into 3 main parts, the background script, the main script and the popup HTML page which uses the popup script. The background script monitors the current URL and if the URL is equal to a google search URL the extension then enables itself and allows the user to choose a clustering method. Once the user chooses a clustering method the extension clusters the Google results, and displays the clusters, as shown in Figure 3, instead of the Google results.

4.1.1 The Background Script

The background script is kept in memory and monitors the current URL. Once the URL is a Google search URL, the background script adds 2 parameters to the URL to force Google to display 100 results in a single page and to disable auto complete, which is enabled by default. The background script then enables the chrome extension, which the user can see as the chrome extension icon becomes coloured.

4.1.2 The Popup Page and the Popup Script

The purpose of the popup page is to serve as the user interface, where once the user clicks on the extension icon the popup page is displayed as a dropdown window as can be seen in Figure 1. From the popup page the user is allowed to choose the algorithm, which he wishes to use for clustering as can be seen in Figure 2. The popup page also contains a button called "Cluster Results" which the user can click once he has chosen the clustering algorithm. Once the button is clicked the popup script constructs a configuration object, containing all the options the algorithms use. The popup script then accesses the background page and sends the configuration object to the main script, which starts the clustering process.

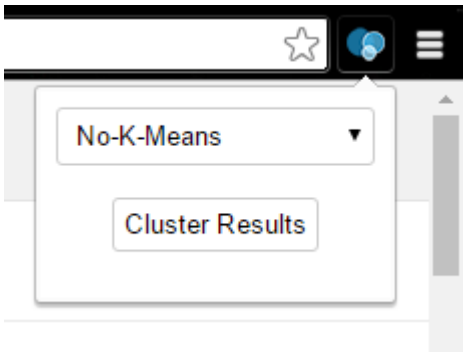

Figure 1 Popup Menu

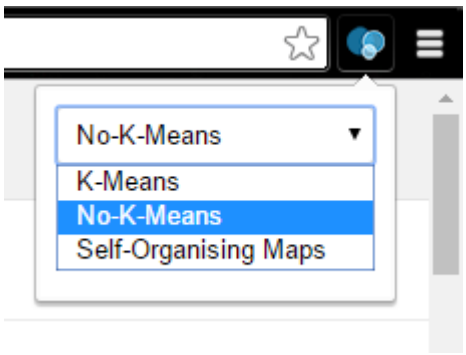

Figure 2 Popup Menu - Clustering Algorithms

4.1.3 The Main Script

The main script first fetches the google search results and performs pre-processing on the results. The algorithm then calls the respective clustering algorithm to perform clustering. After the algorithm is done clustering the main script also displays the results in place of the actual google results.

4.2 Information Pre-Processing

Pre-Processing occurs in the main script. Before clustering the main script pre-processes the results, to remove stop words, and stem the remaining words. Stop words are words which are too frequently occurring across different documents. An example of a stop word is the determiner "the" which will not be useful to distinguish between documents. Stemming is the process of reducing words to their stems, e.g. "connect" is the stem for "connecting", "connected", "connection" etc... The main script first obtains the google search results, and excludes any results containing images. The pre-processing function then uses the title and content of each search result, converts them to lower case and removes any symbols using a regular expression. It then calls the stemming function which uses an implementation of Porter's stemmer

[11]. The stemming function then stems the query terms, the stop words, which is a pre-defined list of stop words [12], and stems the list of words obtained from the google search results. The function then removes any stop words and query terms which occur in the list of words obtained from the google search results. The remaining words are then used to build a histogram template. The histogram template is a blank histogram with no word frequencies which the algorithm can use to keep the same order of words when constructing the histogram for each search result. The histogram template is then used to create a word histogram for each search result, which is made up of words and their frequencies.

4.3 Clustering of Results

After the feature vectors are passed to the algorithm in question, the function returns a list of clusters, each containing a list of search results, including the original HTML. These snippets are used to construct the clustered results in the Google results page itself. Every cluster is collapsed, leaving only one significant result appearing, as shown in Figure 3. The user is then able to expand and collapse the results in the cluster, as shown in Figure 4. Every cluster is labelled with a query which attempts to summarise the contents of the cluster. This query is clickable, and forwards the user to a new Google search. The query is constructed by appending the most occurring word in the cluster (i.e. the word/s appearing most in the cluster documents) to the original query.

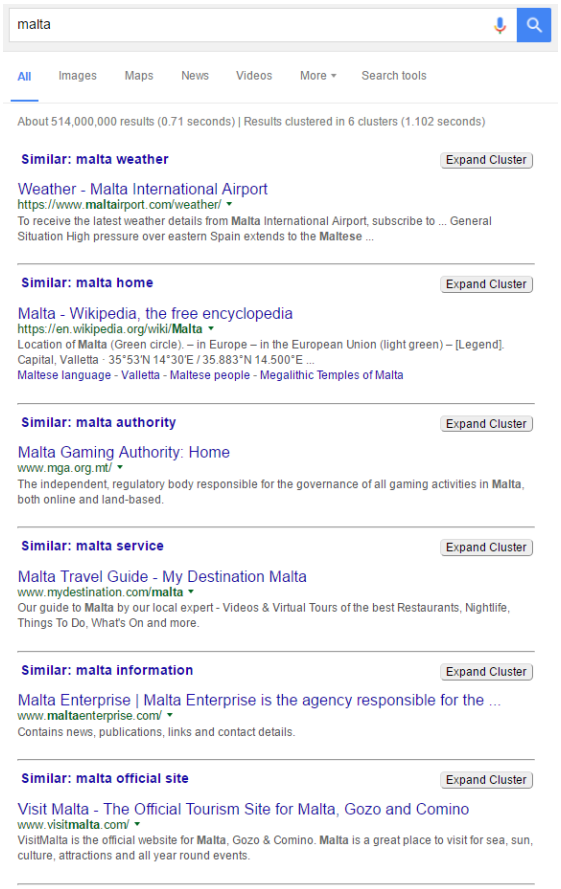

Figure 3 Results shown in collapsed clusters.

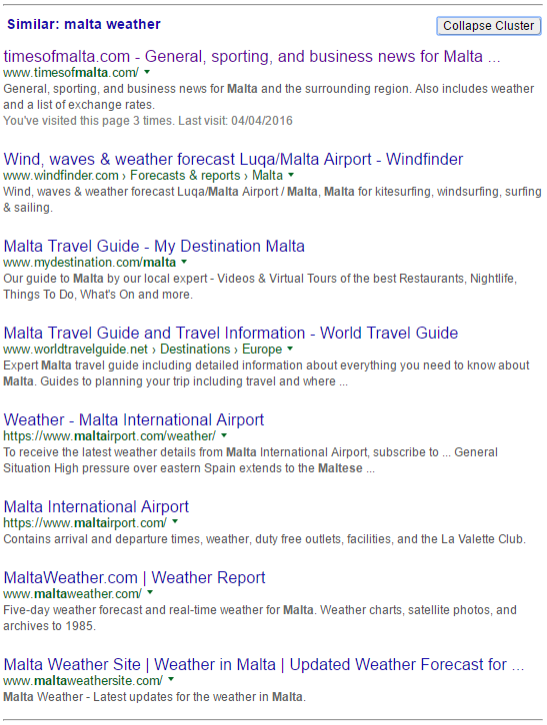

Figure 4 Results in an expanded cluster.

5. Testing

In this section, the three algorithms implemented in the extension are tested using two different queries; "fish and chips" and "jaguar". The first query phrase is not ambiguous like the second, however it is shown below that the extension is able, at different degrees, to cluster similar results for this query as well; for example by grouping results referring to restaurants and those referring to recipes. For the second query, results are expected to contain links to websites concerning different meanings of "jaguar". The following tests show how the algorithms perform at handling this ambiguity.

In addition, the depth at which the system is able to keep clustering search results is examined for each algorithm. This is done by searching for similar results to particular clusters and clustering the new results again.

The time taken for each algorithm to cluster the results is also examined.

5.1 K-Means

5.1.1 Test 1 – Search Query "fish and chips"

As can be seen in Figure 5, K-means did not manage to find more than one Wikipedia search result and used the default value of K. This resulted in five clusters.

As can be seen in Figure 6 the expanded cluster titled "fish and chips recipe", most of the documents in that particular cluster are about recipes.

5.1.2 Test 2 – "jaguar"

As can be seen in Figure 7, K-Means found 2 Wikipedia search results; one on jaguar the animal, the other one on jaguar the car, it then used these as the initial clusters and used $K = 2$. This results in two clusters one on cars the other one on animals. As can be seen Figure 8 the documents in the expanded cluster named "jaguar cat" are mostly about animals.

5.1.3 Depth Clustering

The first depth clustering test was done by refining the query to "jaguar cat", and the new results where re-clustered. The result can be seen in Figure 9 and in Figure 10 one can see the documents in one of the clusters titled "jaguar cat wild", where the documents in this particular cluster are mostly about the wild animal.

The second depth clustering test was done by refining the query to "jaguar cat wild", and the new results where re-clustered. The results of this re-clustering can be seen in Figure 11 and in Figure 12 one can see one of the expanded clusters titled "jaguar cat wild habitats", where the documents in this particular cluster are mostly about the habitat of the wild animal.

The third and final depth clustering test was done by refining the query to "jaguar cat wild habitats", and the new results where re-clustered. The results of this re-clustering can be seen in Figure 13 and in Figure 14 one can see the documents in one the expanded clusters titled "jaguar cat wild habitat protecting", where the documents in this particular cluster about the title "jaguar cat wild habitat protecting" are considerably less.

5.2 No-K-Means

No-K-Means was tested using the search query "fish and chips", in order to examine the speed and quality of the clustering, and also using the search query "jaguar", mostly to examine the depth of clustering.

As shown in Figure 15, the clustering took just 0.581 seconds. The figure also shows the usefulness of clustering, since the user is able to choose which category to browse. For example, in this example, one can expand the "fish and chips recipe" or the "fish and chips restaurant", depending on the user's interest.

In Figure 16, a cluster with similar query "fish and chips restaurant" is expanded, and it can be clearly shown that every search result in this cluster is about a specific restaurant, or restaurants. This shows great clustering quality in order to fit the user's interest.

In Figure 17, searching the term "jaguar" and clustering the results gives good unambiguous results. The algorithm took even less time, i.e. 0.445 seconds. As before, the results clusters seem to be quite sensible, where the results are categorised as cars, cats, and even restaurants.

In Figure 18, the depth of clustering is increased, by re-clustering a search query, "jaguar cats", which was extracted from a cluster in the previous clustering.

Figure 19 - Figure 22 show clustering depth, where 6 clusters are reduced to just 1 cluster, due to the increased similarity between the remaining search results. The search query became very specific ("jaguar cats southern arizona 2016 wild mexico sonora"), referring to a specific event where a very rare jaguar was seen in Southern Arizona, Sonora, Mexico.

5.3 Self-Organising Maps

Since the SOM algorithm is not deterministic, as the initial network weights are randomised, the clustering of the results may vary from one

execution to another, even when given the same training vectors.

As evident in Figure 23, SOM took just over 1 second to cluster the results with the fixed parameters (5 iterations and a network width of 5).

The clusters feature different topics which may be of interest to different users or at a different time. For example, the second cluster is labelled "fish and chips restaurant" while the last cluster is labelled "fish and chips recipe". These may be relevant to different users; those who are looking for some place to eat, and others searching for a method to cook.

In Figure 24, it is evident that the results in the 'restaurant' cluster are actually all restaurants which serve fish and chips or have "fish and chips" in their name.

For the second query, "jaguar", SOM took around 1.5 seconds to cluster the results with the fixed parameters. This is shown in Figure 25.

The proposed queries for similar searches for the clusters, however, seem less sensible than before. A number of clusters seem to be referring to the Jaguar automobiles while only one cluster for "cats" was created. This is examined in Figure 26.

In Figure 26, it is shown that the top 10 results in the "jaguar cats" cluster contain a majority of the documents related to cats, one result related to an American football team and one related to a Jaguar particular car model.

If the user decides to continue searching using the "Similar" functionality with the "jaguar cats" query, and clusters the results once again, output similar to that in Figure 27 should be expected.

As evident, the number of clusters is greater for this particular search query, however some appear to be close to each other, such as "jaguar cats America" and "jaguar cats Americas", and the three clusters with a similar query "jaguar cats tiger".

From the above and other tests, it was noted that SOM was more effective when the search query is specific, such as "fish and chips", and thus useful for queries which are not ambiguous.

5.4 Discussion

When there was minimal ambiguity in the query phrase, the three algorithms managed to give satisfactory results when clustering the google search results. When the search query was ambiguous the SOM algorithm was not as suitable at clustering the search results as the other algorithms. This is due to the quality of the clusters and the time taken to perform the clustering, which was longer than 1 second.

When depth clustering was performed, both the K-Means and the No-K-Means algorithms managed to produce good clusters which had a good refined query phrase (similar query link). The No-K-Means algorithm managed to perform better in depth clustering, since it managed to reduce the documents to one cluster after the query was refined.

6. Evaluation

The four clustering algorithms (the above three and Spectral Clustering using MATLAB), were evaluated using a specific dataset. This allows the determination of the fittest algorithm for the type of data we are considering, i.e. results from a search engine.

In this section, the gold standard used in the evaluation is described, followed by an overview of the evaluation metrics. The evaluation of each of the four clustering algorithms is then discussed.

6.1 The SemEval 2013 Task 11

SemEval is a collection of evaluations of computational semantics and is concerned with the meaning of language.

SemEval-2013 Task 11[1] is concerned with grouping search result snippets into clusters where the results in each cluster have similar semantic meaning, and each cluster has its own meaning of the search query.

The organiser of this task, Navigli et al. [2] created a gold standard collection which contains 100 search queries about ambiguous topics. These queries were then searched using Google search engine, retrieving 64 results for each query. The results were then manually organised into groups, classified by the topic, and every result was represented by its URL, title and snippet.

An automated evaluator was provided in order to determine the accuracy of any search clustering results, which is evaluated using the values F1, Rand Index, Adjusted Rand Index and Jaccard Index.

6.2 Scoring and Measures

In this section, the sample space S represents all the documents to be clustered, while sets X and Y refer

[1] https://www.cs.york.ac.uk/semeval-2013/task11/

to the list of clusters generated by the algorithm and the list of clusters of the SemEval 2013 Task 11 dataset.

Subsets $\{X_1, X_2, \dots, X_i\}$ and $\{Y_1, Y_2, \dots, Y_j\}$ refer to the clusters of documents of X and Y respectively.

6.2.1 F1

The F1 score is an accuracy measure of two data sets, were the result is a value between 0 and 1, 0 being the worst possible value while 1 being the best.

Precision is defined as the number of positive cases which were predicted positives divided by the number of all the predicted positives.

Recall is defined as the number of positive cases which were predicted positives divided by the number of all the positive cases.

In this case the positive cases of a cluster are the documents of the cluster according to the SemEval 2013 Task 11 dataset, while the predicted positives are the documents of the cluster according to the clustering algorithm being used.

F1 is then calculated as the weighted product of precision and recall, divided by the sum of precision and recall, given by Eq. 10:

$$F_1 = \frac{2 \times precision \times recall}{precision + recall} \quad \text{Eq. 10}$$

6.2.2 Rand Index

Two sets X and Y, which are the subsets that make up sample space S, can be divided into subsets {X1, …, Xi} and $\{Y_1, \dots, Y_j\}$ respectively.

The number of pairs of elements in S which are in the same set in X and in the same set in Y, is denoted as a.

The number of pairs of elements in S which are in different sets in X and in different sets in Y, is denoted as b.

The addition of these values, i.e. a + b, gives the number of agreements between X and Y.

The number of pairs of elements in S which are in the same set in X and in different sets in Y, is denoted as c.

The number of pairs of elements in S which are in different sets set in X and in the same set in Y, is denoted as d.

The addition of these values, i.e. c + d, gives the number of disagreements between X and Y.

Rand index is defined as the number of agreements divided by the sum of the number of agreements and the number of disagreements, given by Eq. 11:

$$R = \frac{a+b}{a+b+c+d} \quad \text{Eq. 11}$$

The rand index is also a value between 0 and 1.

6.2.3 Adjusted Rand Index

Two sets X and Y, which are subsets of the sample space S, can be divided into subsets $\{X_1, \dots, X_i\}$ and $\{Y_1, \dots, Y_j\}$ respectively, which can be summarised in the following table:

X/Y	**Y_1**	**Y_2**	**…**	**Y_j**	**Sums**
X_1	n_{11}	n_{12}	…	n_{1j}	a_1
X_2	n_{21}	n_{22}	…	n_{2j}	a_2
. .	. .	. .	. .	. .	. .
X_i	n_{i1}	n_{i2}	…	n_{ij}	a_i
Sums	b_1	b_2	…	b_j	n

where n_{ij} is the number of common objects in the corresponding sets, given by:

$$n_{ij} = |X_i \cap Y_j| \quad \text{Eq. 12}$$

The adjusted rand index is defined as the following:

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{n}{2}}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_{bj} \binom{b_j}{2}\right] - \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{n}{2}}} \quad \text{Eq. 13}$$

6.2.4 Jaccard Index

Jaccard index compares the similarity of two data sets, X and Y by dividing the number of intersecting elements in the sets by the size of the union of the sets, given by:

$$J(X,Y) = \frac{|X \cap Y|}{|X \cup Y|} \quad \text{Eq. 14}$$

This gives a value between 0 and 1, where 0 means that no elements in one set can be found in the other while 1 means all elements of one set can be found in the other.

6.3 K-Means

The K-Means implementation was tested using search results from the gold standard dataset. The algorithm was tested using 4 different cases, in each case the algorithm was tested multiple times with different values of K, the best result in each test case is displayed in the Table 1.

The K value is the number of clusters which the K-Means algorithm should produce if it does not manage to meet certain conditions. This is mainly used in the Wikipedia search results method where if the algorithm does not manage to find more than

1 Wikipedia search results it uses the default K value to find random initial centroids.

In the case when the Wikipedia search results are not considered the number of clusters is set to the default K value.

Table 1 F1, RI, ARI and Jaccard scores for K-Means using WikiPedia and Dunn's Index.

	K	F1	RI	ARI	JI	Avg.	Avg. # Clusters	Avg. Cluster size
Using Wikipedia	3	0.68	**0.65**	**0.21**	**0.35**	**0.47**	5.92	15.24
Without Wikipedia	9	**0.70**	0.62	0.14	0.20	0.42	9	7.11
Dunn's Index - Using Wikipedia	/	0.63	0.59	0.08	0.20	0.38	7.71	10.75
Dunn's Index - Without Wikipedia	/	0.63	0.59	0.08	0.20	0.37	7.49	10.95

The first case was using Wikipedia search results as the initial centroids if the amount of Wikipedia search results was greater than 1, this also meant that the value of K changed each time to the amount of Wikipedia search results. If the algorithm only found one Wikipedia search result it used a default value for K for this particular case which was 3. This case produced the best results.

The second case the algorithm produced the best results with the default value for K for this particular case which was 9. In this case the algorithm used random documents as the initial centroids.

The third case, the algorithm was tested using an implementation of Dunn's index and using Wikipedia search results as the initial centroids. Where if the algorithm does not find more than 1 Wikipedia search result, the algorithm uses the K value determined by the Dunn's index. If the algorithm manages to find more than one Wikipedia search results the amount of Wikipedia search results overrides the K value.

The fourth case, the algorithm was tested using an implementation of Dunn's index where the initial centroids where random documents.

6.4 No-K-Means

In order to evaluate the No-K-Means algorithm, different threshold values were passed to it, in order to determine the threshold for the best results.

Cosine similarity between two text vectors can only vary from 0 to 1, thus for evaluation, thresholds from 0 to 1 were passed, at intervals of 0.01, which sum up to 100 different thresholds.

The following table shows the top five results:

Table 2 F1, RI, ARI and Jaccard scores for No-K-Means.

t	F1	RI	ARI	JI	Avg.	Avg. # Clusters	Avg. Cluster Size
0.05	0.72	**0.69**	**0.28**	0.36	**0.51**	7.8	9.01
0.04	0.7	0.68	0.27	**0.37**	0.51	7.12	9.86
0.06	0.73	0.69	0.27	0.34	0.51	8.28	8.59
0.07	**0.74**	0.69	0.26	0.33	0.51	8.73	8.12
0.08	0.74	0.69	0.26	0.32	0.50	9.02	7.79

This shows that the best results are obtained with a threshold of 0.05, and that the threshold values of the top five results are all similar.

When testing the algorithm with a threshold of 0.05, the maximum Rand Index (RI), 0.69, and the maximum Adjusted Rand Index (ARI), 0.28, were obtained. The maximum F1, 0.76, was obtained with a threshold of 0.15, and the maximum Jaccard Index (JI), 0.4, was obtained with a threshold of 0. However, when calculating the average of the four scores, the maximum average score, 0.51, was obtained with the threshold value 0.05.

6.5 Self-Organising Maps

The results yielded by the algorithm for the SemEval dataset were evaluated by the following procedure.

The number of iterations, i, was varied between 5 and 20, and a square network of width, w, varying from 3 to 25 was used. The results were documented and the following table shows the top three.

Table 3 F1, RI, ARI and Jaccard scores for SOM.

i	*w*	F1	RI	ARI	JI	Avg.	Avg. # Clusters	Avg. Cluster Size
8	23	**0.68**	**0.61**	**0.15**	**0.25**	**0.42**	4.80	13.49
8	21	0.68	0.61	0.15	0.25	0.42	4.89	13.30
7	19	0.68	0.61	0.14	0.25	0.42	4.82	13.45

The above results indicate that the optimal parameters for using the Self-Organising Map with the data in question are a network width of 23 and a limit to 8 iterations.

This is, however, unfeasible for the purpose of our extension, as the training takes seconds, or even minutes, to finish.

In light of this, a sub-optimal choice of parameters was taken, without radically reducing the effectiveness of the algorithm. Both the number of iterations and the network width are locked at 5. This decision was based on a number of tests on data retrieved from the Google search engine.

The table below shows the values for F1, Rand Index, Adjusted Rand Index and Jaccard Index metrics for SOM with the 5 iterations and a network width of 5, and how these compare to the top results.

Table 4 F1, RI, ARI and Jaccard scores for SOM.

i	*w*	F1	RI	ARI	JI	Avg.	Avg. # Clusters	Avg. Cluster Size
8	23	0.68	**0.61**	**0.15**	**0.25**	**0.42**	4.80	13.49
8	21	0.68	0.61	0.15	0.25	0.42	4.89	13.30
7	19	0.68	0.61	0.14	0.25	0.42	4.82	13.45
5	5	**0.69**	0.60	0.12	0.21	0.40	6.53	10.44

6.6 Spectral Clustering

The algorithm was tested using the SemEval gold standard query data. Tests for spectral clustering were conducted with 2 different indices, each index was implemented with 2 different distance functions, Euclidean and Cosine distance. Spectral clustering has one parameter, the Sigma. Multiple tests were conducted with Sigma ranging from 0-10 in increments of 0.1. These tests were also run for the 4 indices explained above. The clustered data is evaluated by 4 metrics F1, Rand Index, Adjusted Rand Index and Jaccard Index. The Average of these indices is calculated to identify the best parameter for the algorithm. The table below shows the top test results and it concluded that the best index was PBM with cosine distance having an average of 0.393 and the worst was Dunn's index with cosine distance having an average of 0.367. The best sigma for PBM with cosine distance was 0.9.

Table 5 F1, RI, ARI and Jaccard scores for Spectral Clustering using PBM and Dunn's Indices, and Cosine and Euclidean Distances.

	Sigma	F1	RI	ARI	JI	Avg.	Avg. # Clusters	Avg. Cluster Size
PBM Cosine	0.9	**0.70**	**0.55**	**0.06**	0.26	**0.39**	19.23	4.36
PBM Euclidean	1.3	0.63	0.54	0.05	0.29	0.38	10.97	8.44
Dunn's Cosine	0.7	0.62	0.49	0.04	**0.32**	0.37	9.69	10.78
Dunn's Euclidean	1.3	0.63	0.54	0.05	0.29	0.38	10.97	8.44

6.7 Discussion

After evaluating the algorithms using the SemEval 2013 Task 11 dataset, it was evident that the No K-means Algorithm performed better than the others. This was concluded by comparing the average of the F1, Rand Index, Adjusted Rand Index and Jaccard Index, where No-K-Means achieved an average of 0.51, while K-Means, SOM and Spectral Clustering achieved averages of 0.47, 0.42 and 0.39, respectively.

7. Limitations and Future Work

In section 3, it is mentioned that the algorithms implemented in the extension do not perform any cluster analysis. The reason for this is that performing cluster analysis by using, for example, Dunn's Index or PBM Index to identify the best set of clusters, would highly increase the computation time, especially when the number of varying parameters is more than one. This is not practical as the user expects the system to operate in real-time.

As described in section 4.3 the Google Chrome extension has a clickable link that restructures the query for the Google search to reflect that cluster. It would be a future improvement to explore the restructuring of the query such that the change in performance can be compared. One such method to restructure the query would be to add the query label for the selected cluster and also negate the other cluster query terms. For example, "fish and chips recipes –restaurants" to include results referring to recipes, but excluding those related to restaurants. In addition, currently we are extracting the cluster label by picking the most frequent word in that cluster. It would be more beneficial to identify and implement a more effective labelling algorithm.

Another point, that could make the extension more user-oriented, is to identify ways to rank clusters so that the clusters presented to the users are ordered in a more logical way. In addition to this, it would be beneficial to use data available in the Google Chrome framework, such as browser history and bookmarks, and rank clusters depending on the user model.

8. Division of Work

In this section, the roles of each member of the group are defined. The work was fairly divided, with some focusing more on the development of the extension and others on the research and improvements of the clustering algorithms.

8.1.1 Apap Adrian

During the course of this project I contributed to the implementation of modules of the google chrome extension and implemented the pre-processing methods used before clustering is performed. I also researched K-Means and modified a JavaScript implementation of K-Means to work with the chrome extension.

8.1.2 Buhagiar Juan

During the first phases of the project I researched and tried to implement the Gaussian Mixture Models algorithm. Due to certain limitations of this algorithm it was decided to find another algorithm. Spectral clustering was chosen to replace GMM. Testing was done with spectral clustering and I also implemented two indices Dunn's and PBM index.

8.1.3 Falzon Alan

At the early stages of the project, I searched for JavaScript implementations of K-Means, Self-Organising-Map and Gaussian Mixture Models. My work mostly consisted of the implementation of the No-K-Means search results clustering algorithm. I also researched the F1 score, Rand Index, Adjusted Rand Index and Jaccard Index.

8.1.4 Schembri Yurgen

My role during the development of this project was to implement the post-processing methods (i.e. output of the clusters). With regards to the research aspect, I investigated Self-Organising Maps and modified the JavaScript implementation adopted for this assignment. In addition, I prepared scripts for automating the extraction of results from the SemEval evaluator for easier evaluation.

9. Conclusion

The paper outlines the process that was taken to design and implement a system, which enables users to identify better a set of relevant web pages, using search result clustering. As described, there are several algorithms that cluster data, we have performed tests and evaluation to identify the best algorithm for textual clustering analysis. Using the developed browser extension, tests were performed on real data from google searches which gave promising results. The evaluation process, with the SemEval 2013 Task 11 Dataset, provided a more scientific way to interpret results from the tested algorithms, with the help of the F1, Rand Index, Adjusted Rand Index and Jaccard Index. It was concluded, based on results obtained above, that the No-K-Means algorithm has given the best overall results.

10. References

[1] "Browser Statistics", *W3schools.com*, 2016. [Online]. Available: http://www.w3schools.com/browsers/browsers_stats.asp. [Accessed: 20- May- 2016].

[2] R. Navigli and D. Vanella, "*SemEval-2013 Task 11: Word Sense Induction & Disambiguation within an End-User Application"*, 2013. Appendices

[3] B. Desgraupes, "Clustering Indices", *cran.r-project.org*, 2016. [Online]. Available: https://cran.r-project.org/web/packages/clusterCrit/vignettes/clusterCrit.pdf. [Accessed: 16- May- 2016].

[4] J. MacQueen, “Some methods for classification and analysis of multivariate observations”, *Proceedings of the 5. Berkeley Symposium on Mathematical Statistics and Probability*. Berkeley, Calif.: University of California Press, 1967, pp. 281-297.

[5] C. Staff, J. Azzopardi, C. Layfield and D. Mercieca, "Search Results Clustering without External Resources," *2015 26th International Workshop on Database and Expert Systems Applications (DEXA)*, Valencia, 2015, pp. 276-280.

[6] J. Vesanto and E. Alhoniemi, "Clustering of the self-organizing map", *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 586-600, 2000.

[7] U. von Luxborg, "A Tutorial on Spectral Clustering", *Statistics and Computing*, vol. 17, no. 4, 2007.

[8] "harthur/clusterfck", *GitHub*, 2016. [Online]. Available: https://github.com/harthur/clusterfck. [Accessed: 16- May- 2016].

[9] "LucidTechnics/som", *GitHub*, 2016. [Online]. Available: https://github.com/LucidTechnics/som. [Accessed: 16- May- 2016].

[10] S. Clustering, "Spectral Clustering - File Exchange - MATLAB Central", Mathworks.com, 2015. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/44879-spectral-clustering. [Accessed: 16- May- 2016].

[11] "Porter's Stemmer JavaScript Implementation", *Tartarus.org*, 2016. [Online]. Available: http://tartarus.org/martin/PorterStemmer/js.txt. [Accessed: 16- May- 2016].

[12] "Stopword List 1", *Lextek.com*, 2016. [Online]. Available: http://www.lextek.com/manuals/onix/stopwords1.html. [Accessed: 16- May- 2016].

10.1 K-Means Testing Screenshots

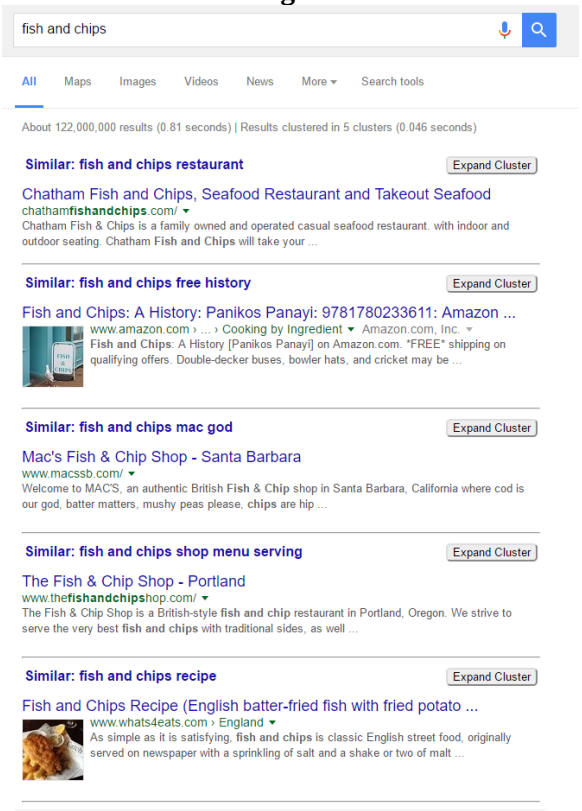

Figure 5 "fish and chips" clustering results

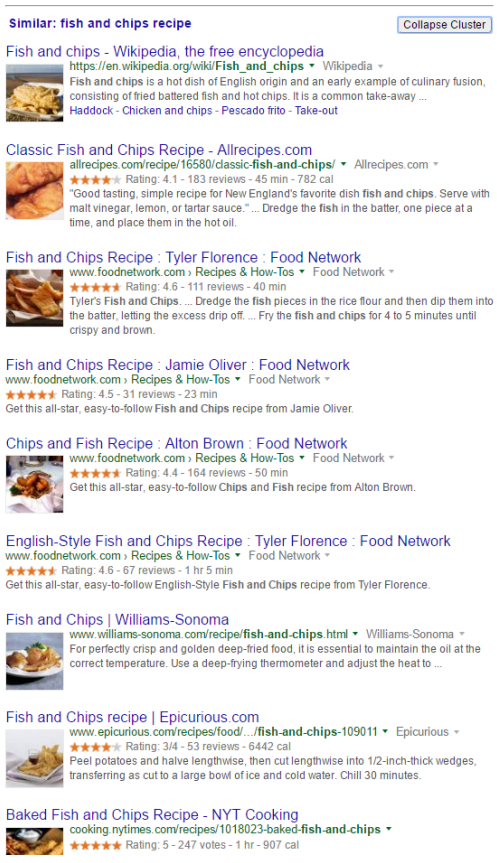

Figure 6 "fish and chips recipe" cluster

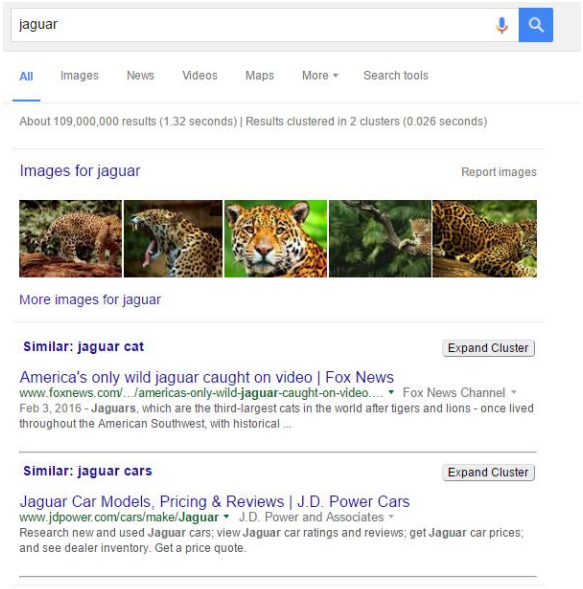

Figure 7 "jaguar" clustering results

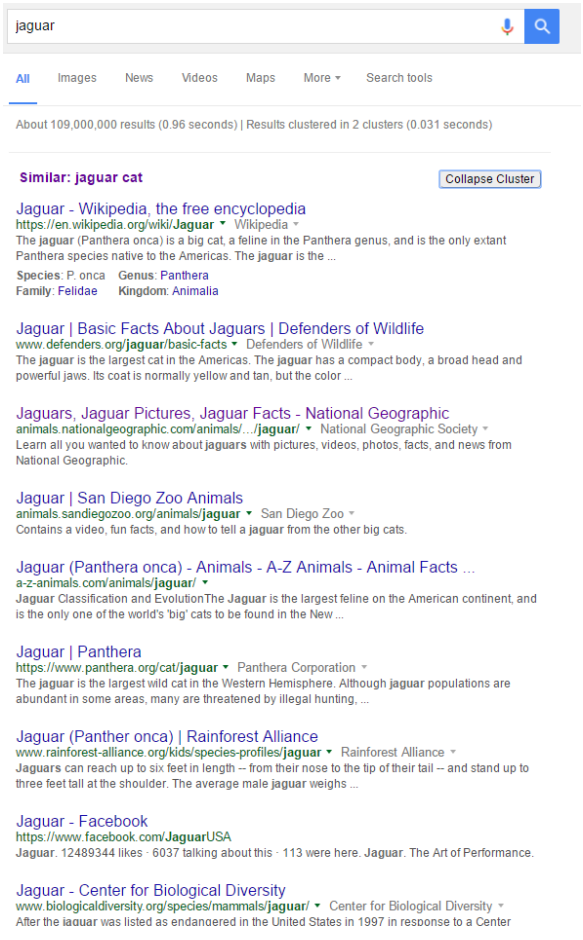

Figure 8 "jaguar cat" cluster

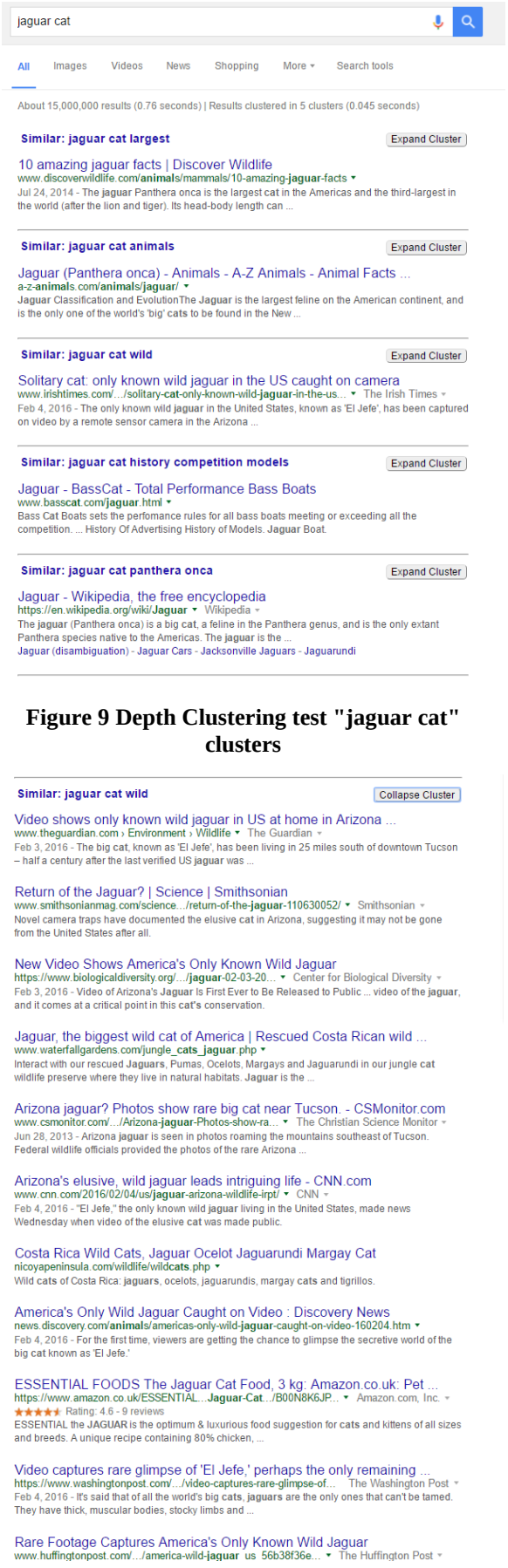

Figure 9 Depth Clustering test "jaguar cat" clusters

Figure 10 "jaguar cat wild" cluster

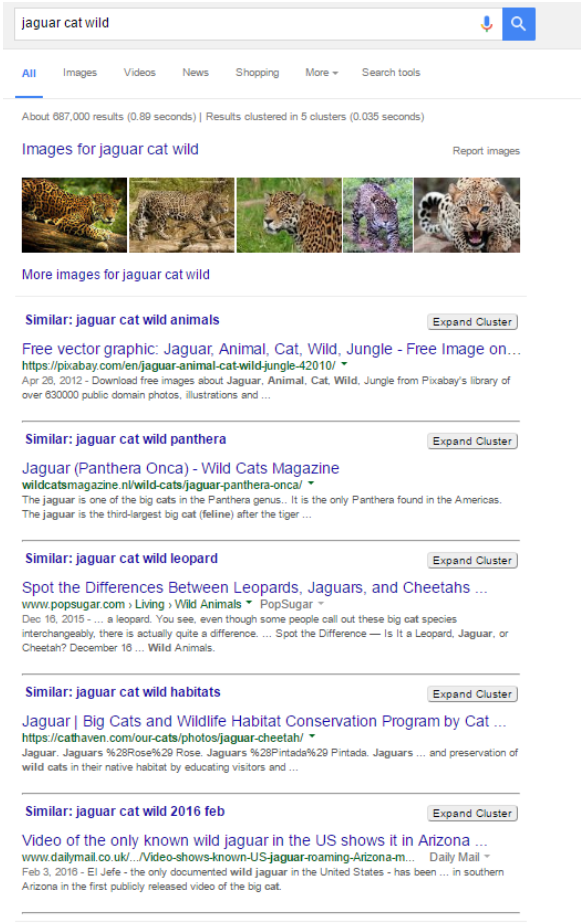

Figure 11 Depth Clustering test "jaguar cat wild" clusters

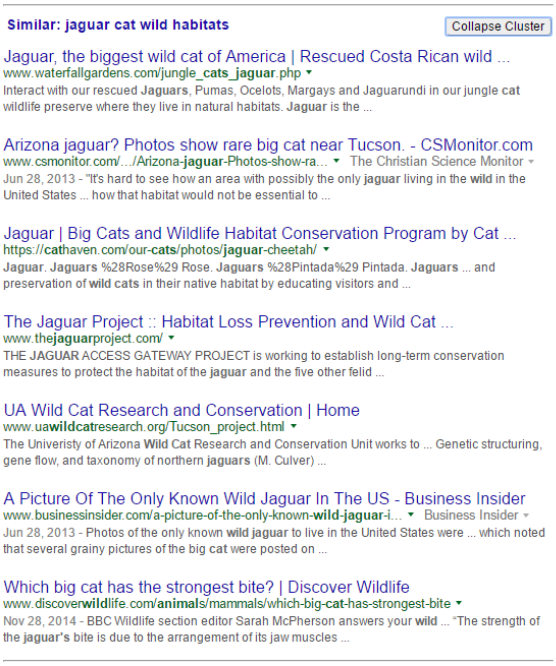

Figure 12 "jaguar cat wild habitats" cluster

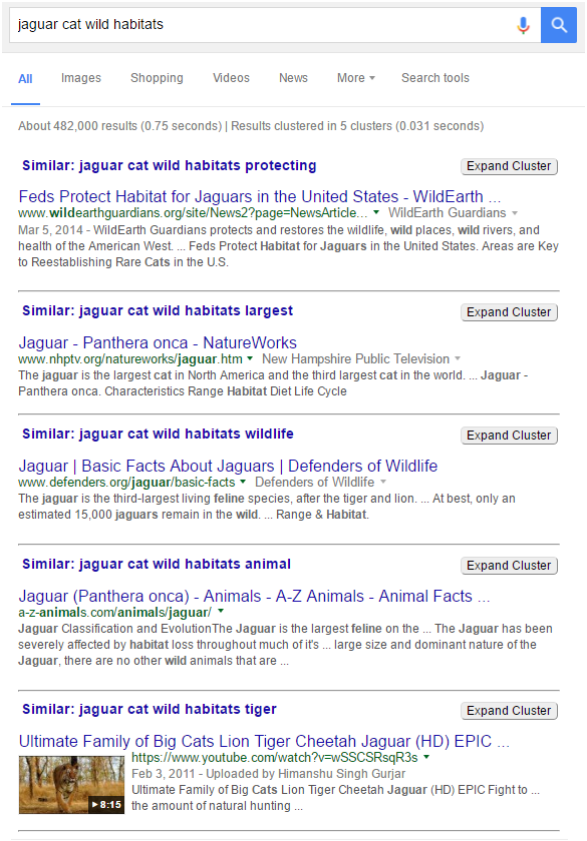

Figure 13 Depth Clustering test "jaguar cat wild habitats" clusters

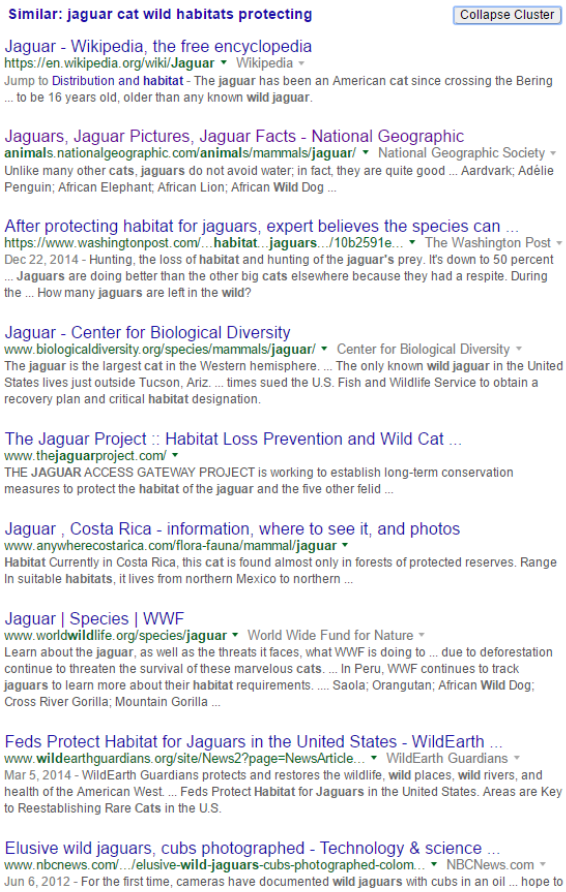

Figure 14 "jaguar cat wild habitats protecting" cluster

10.2 No-K-Means Testing Screenshots

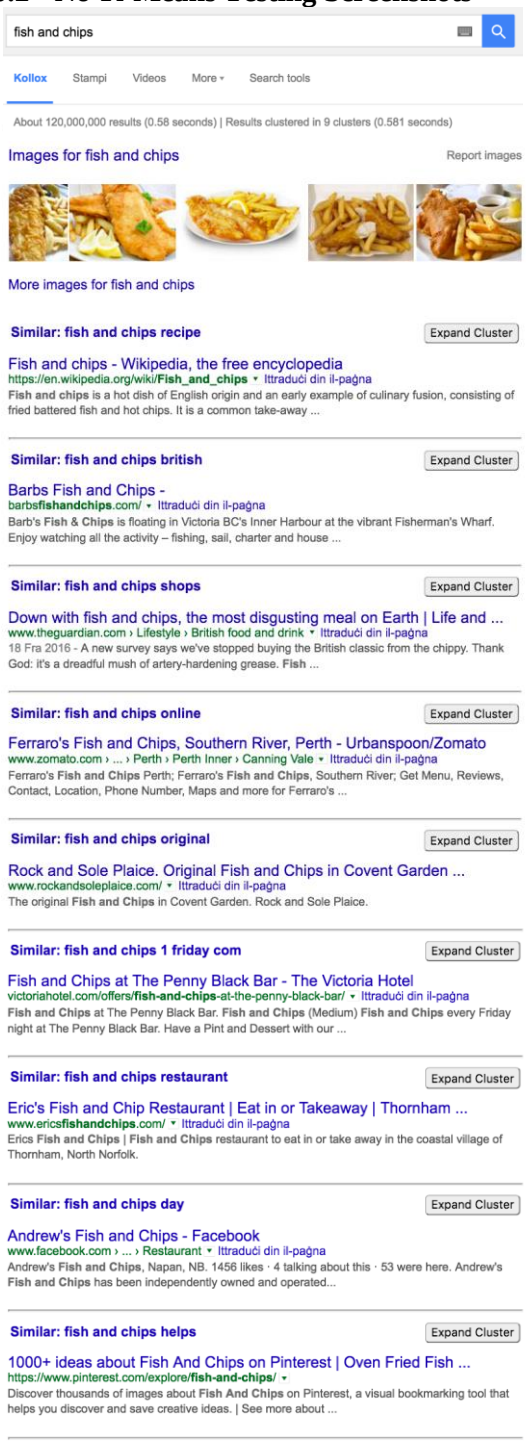

Figure 15 Clustering of results for "fish and chips" using No-K-Means.

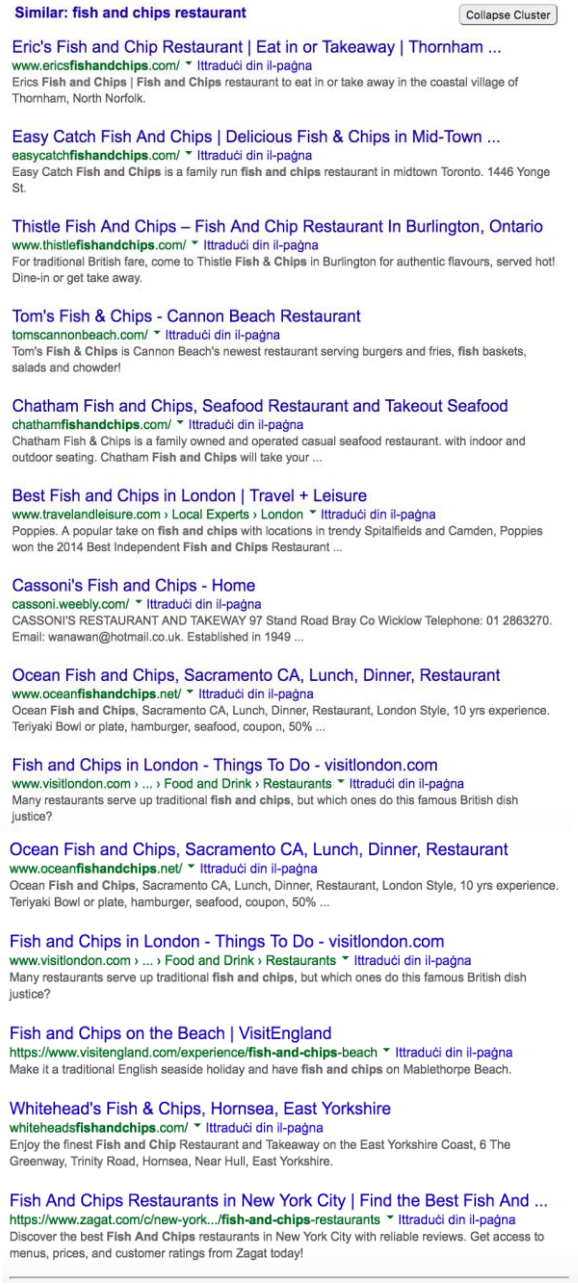

Figure 16 Expansion of cluster with similar search phrase "fish and chips restaurant".

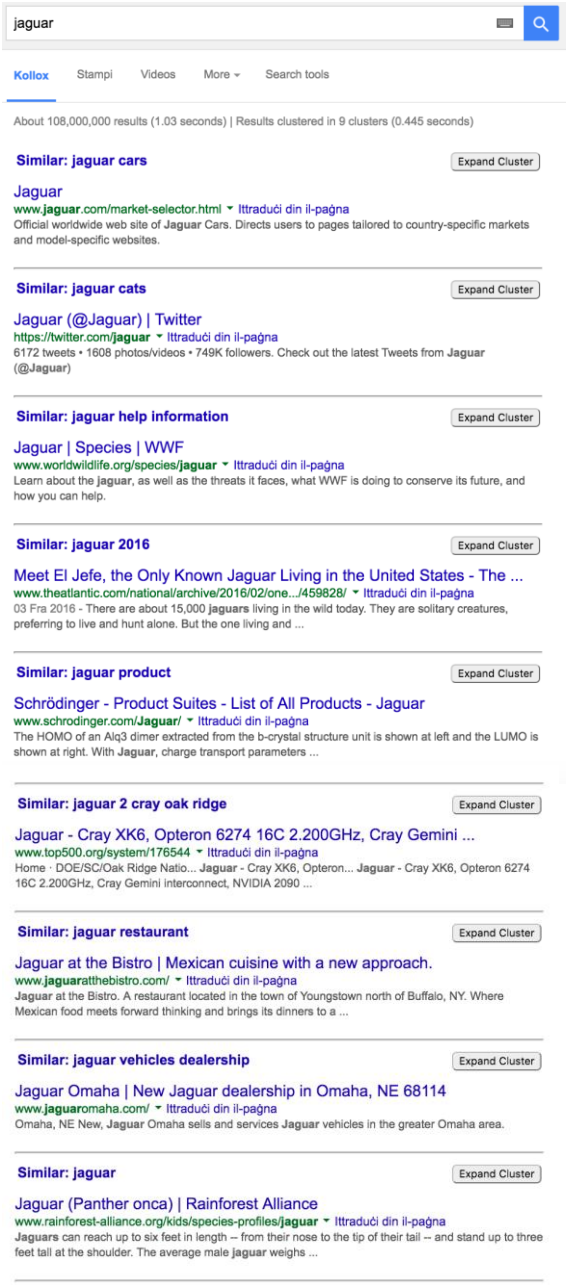

Figure 17 Clustering of search term "jaguar" using No-K-Means.

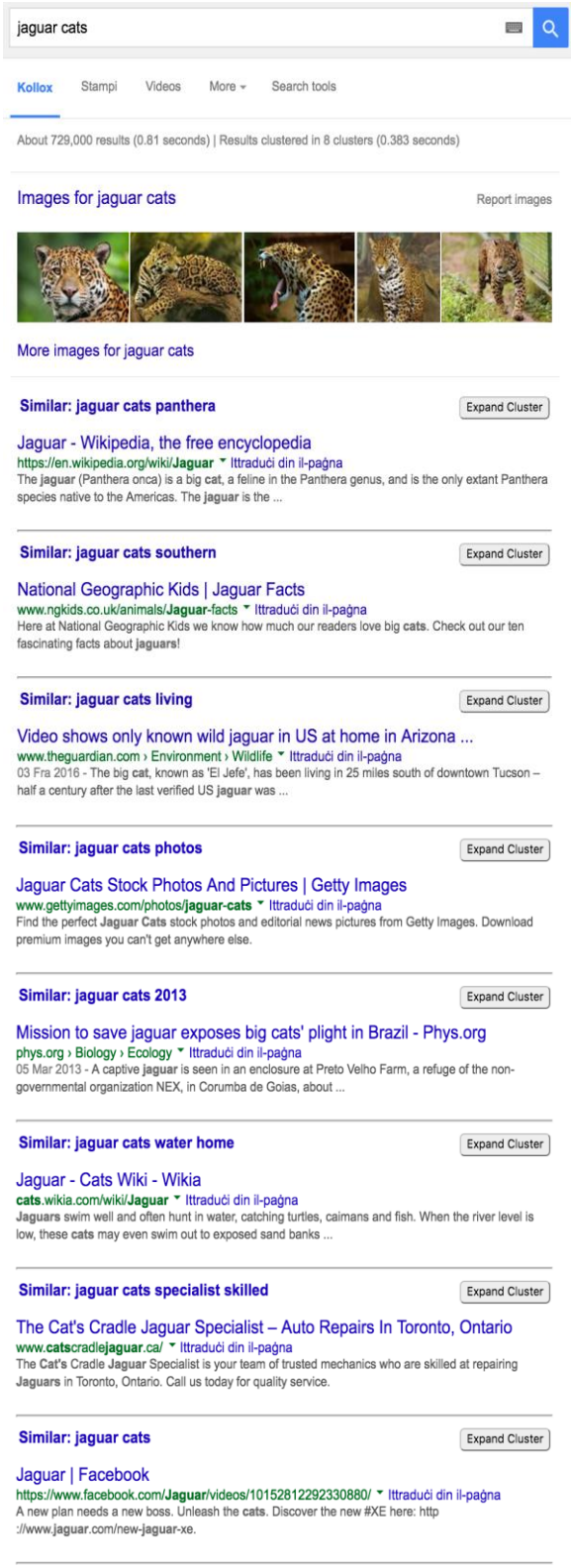

Figure 18 Clustering of search phrase "jaguar cats" using No-K-Means.

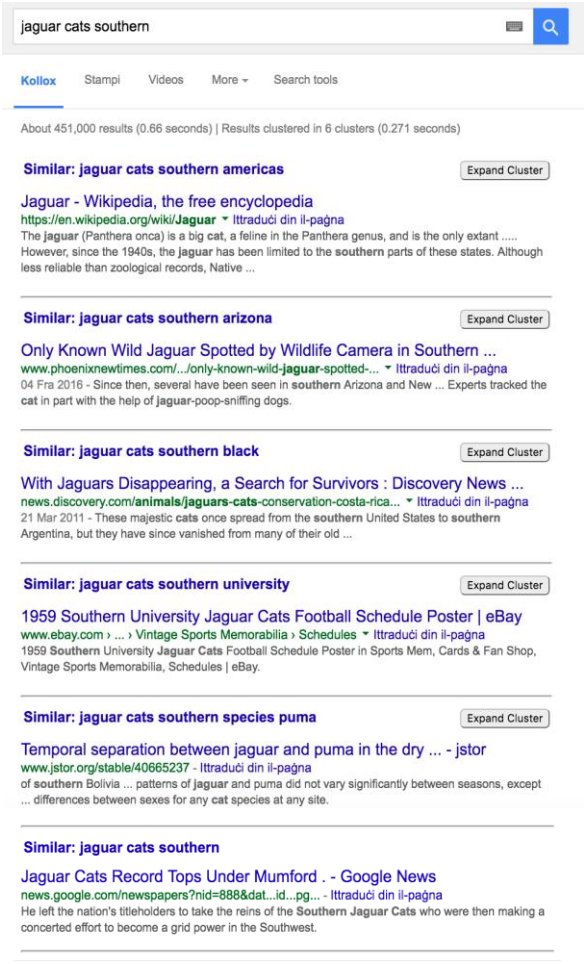

Figure 19 Clustering of results for "jaguar cats southern" using No-K-Means.

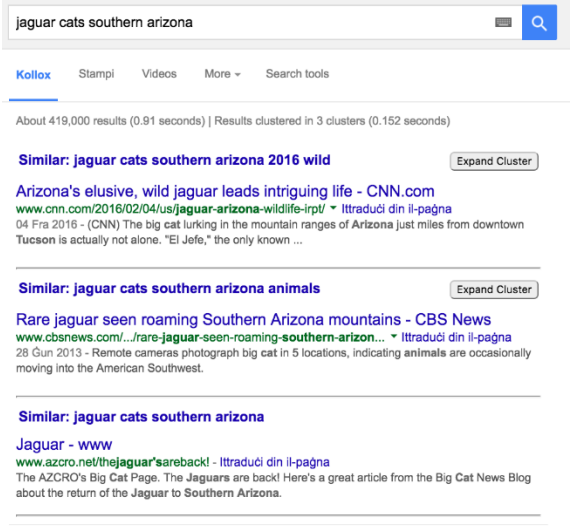

Figure 20 Clustering of results for "jaguar cats southern arizona" using No-K-Means.

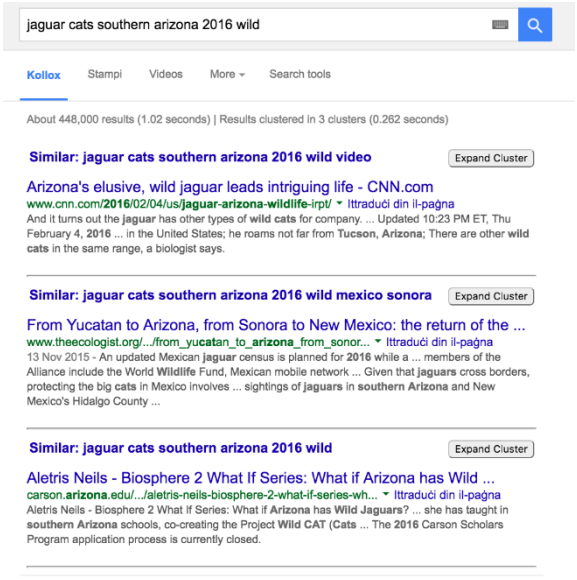

Figure 21 Clustering of results for "jaguar cats southern arizona 2016 wild" using No-K-Means.

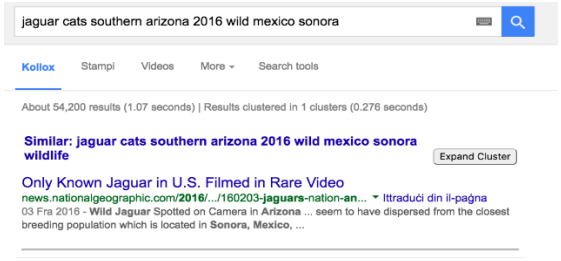

Figure 22 Clustering of results for "jaguar cats southern arizona 2016 wild mexico sonora" using No-K-Means.

10.3 Self-Organising Map Testing Screenshots

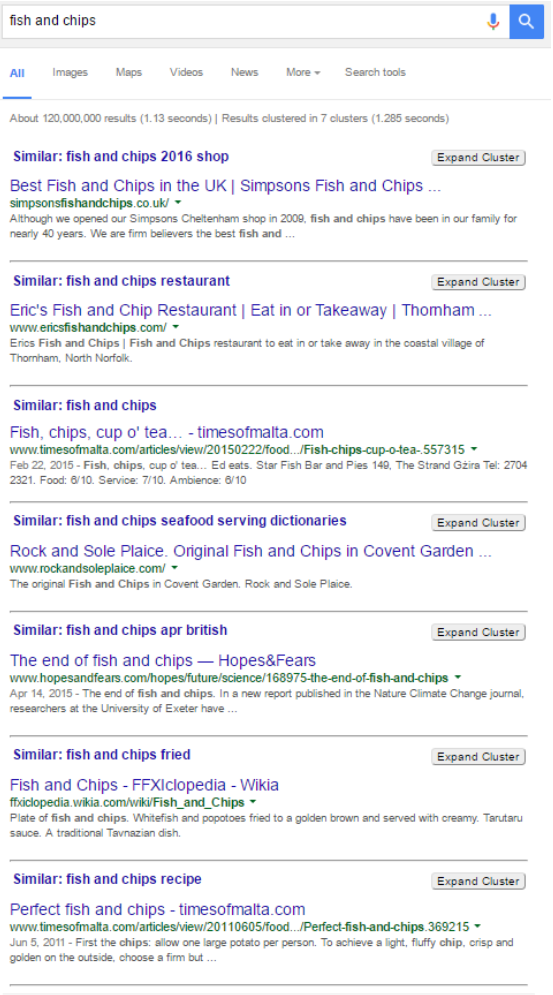

Figure 23 Collapsed clusters for "fish and chips", using SOM.

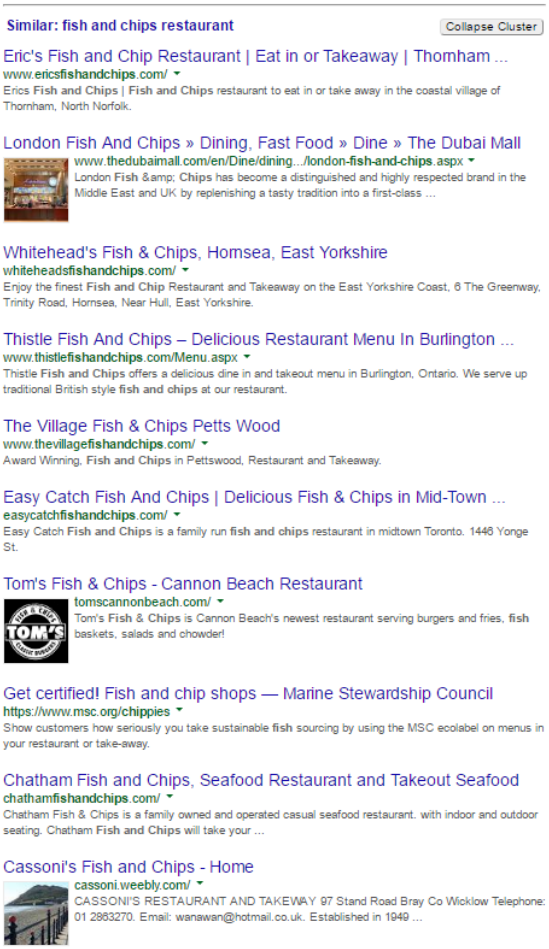

Figure 24 Top 10 results in the expanded cluster "fish and chips restaurant".

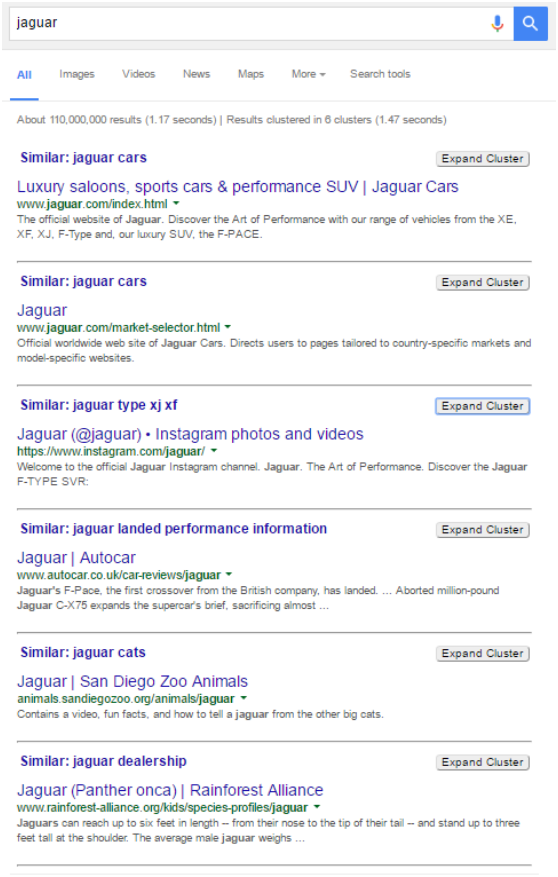

Figure 25 Collapsed clusters for "jaguar" using SOM.

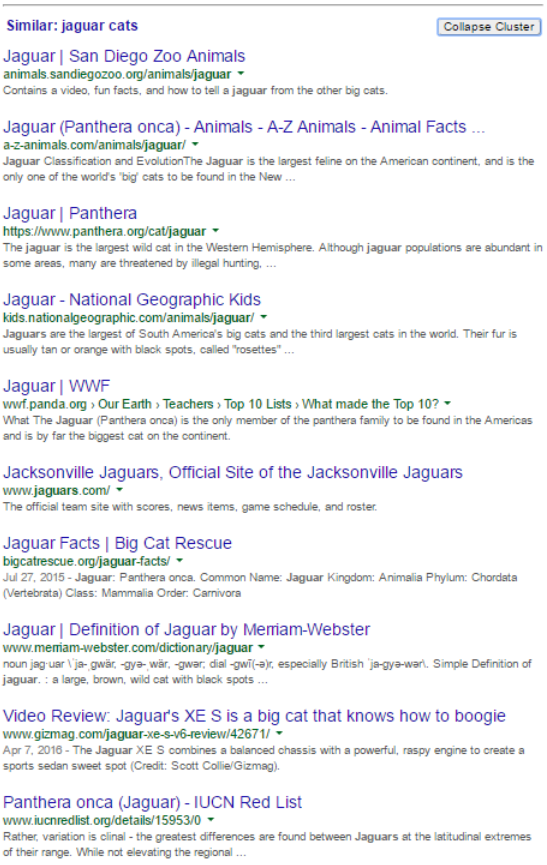

Figure 26 Top 10 results in the expanded cluster "jaguar cats".

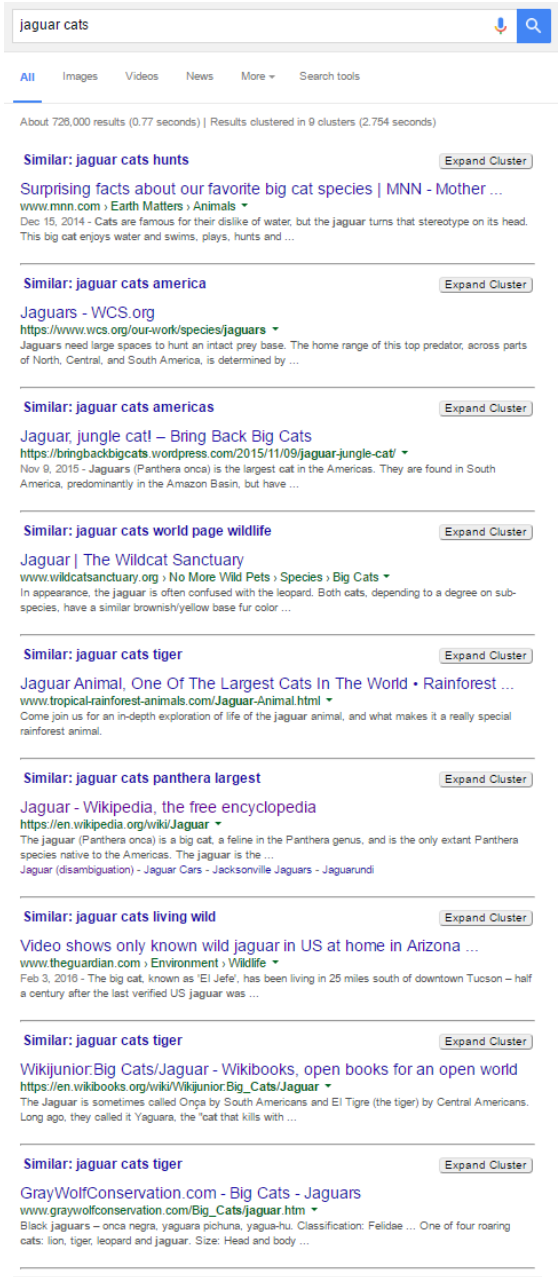

Figure 27 Clustering of results using SOM for the search "jaguar cats".