Bachelor Thesis

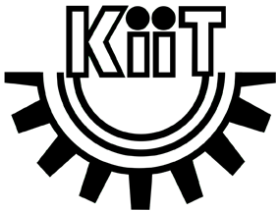
ETOLA: HEURISTIC CLUSTERING OF SHORT TEXT

Submitted in partial fulfillment of the requirements
for the Degree of

Bachelor of Technology
in
Computer Science and Engineering

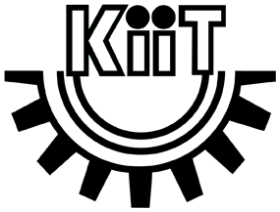
By
Nibir Nayan Bora
(Roll No.: 805090)

Under the guidance of
Bhabani Shankar Prasad Mishra



School of Computer Engineering
KIIT University
Bhubaneswar, Orissa 751 024

Spring Semester, 2012



CERTIFICATE

This is to certify that the thesis entitled "**Etola: Heuristic Clustering of Short Text**", submitted by **Nibir Nayan Bora** in partial fulfillment of the requirements for the degree of **Bachelor of Technology** in **Computer Science and Engineering** at the School of Computer Engineering, KIIT University, Bhubaneswar, India, is a record of research work carried out by him during the academic year 2011-2012 under my supervision. The matter embodied in this thesis is original and has not been submitted for the award of any other degree.

Date:

Dr. Bhabani Shankar Prasad Mishra
(Project Guide)

Signature of Examiner:

1.

2.

Abstract

Document clustering deals with assigning documents to groups (called clusters) in accordance with the general clustering rule, '*high intra-cluster document similarity and low inter-cluster document similarity*'. In this study, we propose a novel heuristics for clustering news headlines. News headlines are grammatically and semantically different from larger bodies of text, like blog posts and reviews. Based on the heuristics, we implemented versions of the *frequent term-based* and *frequent noun-based* clustering algorithms. Both these algorithms, along with k-means, regular frequent term and frequent noun clustering were evaluated using five datasets - Reuters343, Reuters2388 (news headlines), CICLing-2002, Hep-ex and KnCr (scientific abstracts). On interpreting the results based on common external cluster quality evaluation measures (*purity*, *entropy* and *F measure*), it was found that the heuristics performed at par with, or even better than, traditional clustering algorithms and few other intuitive algorithms, when tested using the datasets comprising of news headlines. However, on using the datasets comprising of scientific abstracts, the results were not favorable.

Acknowledgement

The author would like to thank Dr. Bhabani Shankar Prasad Mishra, Associate Professor, School of Computer Engineering, KIIT University, Bhubaneswar, India for his supervision throughout the course of the project.

The author would also like to thank Dr. Anil Kumar Singh and Dr. Sriram Chaudhury of the School of Computer Engineering, KIIT University, Bhubaneswar, India for his valuable comments and suggestions.

The author also extends his gratefulness to the Dean and other faculty members of the School of Computer Engineering, KIIT University, Bhubaneswar, India for providing the opportunity and the facilities to carry out this project.

Nibir Nayan Bora

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

1.1 Objective

A large number of news organizations are involved in reporting news from every nook and corner of the world. With the onset of the Information Era[1] (or the Digital Age), and with the establishment of the world wide web, all these organizations started making their stories accessible over the internet. Now, any common user will be overwhelmed by the amount of news he could get access to. Moreover, there is no way to know which among these news posts are redundant. Wouldn't it be wonderful if all these news stories are available at a single location with similar news posts grouped together? This has been the motivation for our study.

Cluster analysis or *clustering* is the task of assigning a set of objects into groups (called *clusters*) so that the objects in the same cluster are more similar (in some sense or another) to each other than to those in other clusters. Cluster analysis finds applications in numerous fields. For example identify families in plant and animal species, market and demographic research, finding customer groups and communities, image analysis etc.

Document clustering (also referred to as *text categorization*) is closely related to the concept of data clustering. Document clustering is a more specific tech-

[1] http://en.wikipedia.org/wiki/Information_Age

nique for unsupervised document organization, where generally high dimensional documents are the objects in consideration. A large number of studies has investigated document clustering as a methodology for improving search and retrieval, automatic topic identification, document browsing, as well as the primitive task of classification.

1.2 Domain

News headlines are different from previously studied larger bodies of text, such as search results, blog posts and reviews, or even more recently studied short text like scientific abstracts. They are constricted, succinct (often limited to a single sentence) and do not guarantee correctness of grammar. This results in very small dimensionality of a corpus, allowing us to try modified versions of the regular clustering methods as well as various heuristics, while clustering news headlines. Clustering headlines instead of the news posts themselves could provide a significant gain in execution time. Further, an efficient algorithm for clustering of news headlines may find many applications, like creating groups of similar news from different sources[2].

1.3 Study

In this study, we propose a novel heuristics which could be incorporated into frequent term-based clustering to yield better results. We implemented heuristic versions of both frequent term and frequent noun clustering algorithms. These algorithms were then evaluated using two datasets (*Reuters343* and *Reuters2388*) of 343 and 2388 news headlines pertaining to 26 topics (extracted from the Reuters corpus). We also tested our heuristics on three other short text corpuses namely CICLing-2002, hep-ex, and KnCr (comprising of scientific abstracts). The results were then interpreted using commonly used cluster quality measures (*purity*,

[2]Such an approach is used by Google News.

entropy, *F measure*). Since we are concerned with the clustering quality of the algorithms and do not investigate clustering with an application in mind, the evaluation parameters are external measures which assess the results of the clustering algorithms against a set pre-defined set of classes (or categories).

1.4 Structure

Chapter 2 is a summary of different studies in document clustering, classification of clustering algorithms, and various innovative algorithms. Our heuristics is introduced in Chapter 3 and the clustering algorithms, defined. Chapter 4 accounts for the datasets used to evaluate the results and the evaluation parameters. Results of the studies are discussed in Chapter 5 and a final conclusion is drawn in Chapter 6.

Chapter 2

Background

2.1 Classification of Clustering Algorithms

Clustering algorithms can broadly be classified as partitional or hierarchical [7]. Partitional algorithms partition the document space into a specific number of groups, using the iterative relocation principle. *K-means*, *k-medioids*, *bisecting k-means* are a few examples of partitional algorithms. Hierarchical clustering algorithms, on the other hand, begin with all the documents in the document space as individual clusters an iteratively merge the most similar clusters. In contrast to this bottom-up approach (also called the *agglomerative* approach), hierarchical clustering can be top down (or the *divisive* approach). While it has often been argued that hierarchical clustering yields better quality clusters, partitional methods are preferred often because of their linear time complexity.

Carpineto et al. [3] proposed an alternate classification of clustering algorithms based on how well they are prepared to produce sensible, comprehensive, and compact cluster labels. However, this classification type is based on clustering as a technique to improve browsing of search results. According to them, a clustering algorithm could be data-centric, description-aware or description-centric. A data-centric algorithm (e.g. partitional, hierarchical etc.) essentially emphasize on the quality of clusters produced without regarding an appropriate cluster descriptor. Description-aware algorithms (e.g. Suffix Tree Clustering [24]) try to

ensure that the construction of cluster descriptions is that feasible and it yields results interpretable to a human. Description-centric algorithms take into account both quality of clustering and descriptions, prioritizing the quality of the latter over the first. A point to be noted is that most of the traditional texts clustering techniques are adopted from mathematical models which operate on numeric data. In such cases, describing the identified classes is unimportant, or sometimes impossible, hence the algorithms usually fall under data-centric algorithms.

2.2 An Account of few Clustering Algorithms

K-means & Bisecting K-means

K-means [2] [9], a partitional clustering algorithm having time complexity of $O(n)$ is most commonly used. It starts by randomly selecting k data points (or documents) from the corpus (or the document space), which act as the initial cluster centroids. Each document is then assigned to the centroid to which it is closest to. The centroids are recalculated (often as a weighted average of all vectors in the cluster) and the process is repeated until there is no change in the shape of the clusters (or the change falls below a threshold). K-means essentially gives overlapping clusters which are spherical in shape. A variation of this algorithm, bisecting k-means [23] has been shown to perform better by Steinbach et al. [23]. Bisecting k-means follows by considering the entire document space as a single cluster and in each step applying k-means multiple times to find two clusters. The split with the highest average pairwise similarity is retained. This process is repeated until k clusters are obtained.

Agglomerative Hierarchical Clustering

Agglomerative Hierarchical Clustering (AHC) follows a bottom-up strategy which starts by placing each object in its own cluster and then merges these atomic clusters into larger and larger clusters, until all of the objects are in a single cluster

or until certain termination conditions are satisfied. In both agglomerative and divisive hierarchical clustering, the user can specify the desired number of clusters as a termination condition. Based on how the similarity between clusters is measured, agglomerative hierarchical clustering takes different forms - single-link (cluster similarity is equal to the similarity of their most similar pair), complete-link (similarity of the least similar pair) and average-link (similarity is the average of all similarities between pairs of elements across clusters).

Frequent Term-Based clustering

Beil et al. [1] discussed two variants of frequent term-based clustering, one partitional and the other hierarchical. Both the algorithms are based on a greedy technique to identify frequent term sets. These sets are assigned an entropy overlap value each, based on which, the final clustering is chosen. The partitional variant of the algorithm, Frequent Term-based Clustering (FTC), produces flat non overlapping clusters. The hierarchical, Hierarchical Frequent Term-based Clustering (FTC), version of the algorithm follows a top-down approach, running FTC on each level.

Scatter/Gather

Cutting et al. [5] introduced two linear time partitional clustering algorithms - Buckshot and Fractionation, both being used as methods to choose initial centers. After k centers have been found, each document in the corpus is assigned to one of these centers using a *Assign-to-Nearest* algorithm. They also discussed three refinement techniques, namely Iterated Assign-to-Nearest, Split and Join, which is applied to the initial clustering of the corpus. In Buckshot, a small random sample of the documents (of size $\sqrt{kn}$) is chosen, on which a pre-chosen cluster subroutine is applied. The algorithm runs at time complexity $O(kn)$, however, it is not deterministic. Fractionation initially breaks the corpus into N/m buckets of fixed size $m > k$. The cluster subroutine is then applied to each of these buckets

separately to agglomerate individuals into document groups. These groups are then treated as individuals and the entire process repeated until only k groups remain. Cutting et al. also pointed out that Buckshot is a more faster clustering algorithm whereas Fractionation produced a better clustering.

Suffix Tree Clustering

Zamir and Etzioni [24] introduced another interesting linear time clustering algorithm, Suffix Tree Clustering (STC). One notable difference between STC and previously discussed algorithms is that STC does not treat a document as a set of words, but as an ordered sequence of words. This algorithm proceeds by constructing a suffix tree of all the sentences of all the documents in the corpus. Each node of the suffix tree represents a group of documents and a phrase that is common to all. This is referred to as a *base cluster*. Each base cluster is then assigned a score that is a function of the number of documents it contains, and the words that make up its phrase. Further, base clusters with a high overlap in their document sets are merged. The final clusters are reported by sorting the base clusters based on their scores.

2.3 Other Trends in Document Clustering

Apart from these, there are a number of proposed clustering algorithms ([6], [16]) each with their benefits. One popular aspect of investigating document clustering as an application is in improving web search results, towards which many studies have been directed ([13], [24], [25], [5], [4]).

However, our heuristic is motivated by the recent trend in clustering short-text. Shrestha et al. [22], in their study of clustering short text evaluated variants of hierarchical agglomerative clustering and spectral clustering [12] on four different short text corpora - CICLing-2002, Hep-ex, and KnCr [17] and LDC (paragraphs of news posts concerning the *Death of Diana*). Few other studies have dealt with

clustering abstracts instead of full text in papers [14] [18] and other short text domains [19] [10]. Koller and Sahami [10] implemented hierarchical classification using very few words. Zamir and Etzioni [24] also found that clusters based on snippets were almost as good as clusters created using full text of web documents.

Chapter 3

Clustering Algorithm

In this chapter we introduce the novel heuristics that we propose, and discuss in details the clustering algorithms being used. This has been followed up with an account of the refinement process.

3.1 The Hypothesis

The heuristics we propose is intended to be used for modifying the general frequent term-based clustering algorithm, which forms clusters by first finding the most frequent terms in the corpus and then bundling the documents containing each of these terms into a cluster. This is based on the premise that each document is related to a topic with is manifested by the presence of a related term (*key term*). We aim at making the process of finding these key terms easy.

Our hypothesis states, "*the lesser the number of terms in a document, the easier it is to identify the key term*". Evidently, this obligates the assumption that each document contains only one such key term. To understand this hypothesis, consider two news headlines:

H1: *BALDRIGE PREDICTS SOLID U.S. HOUSING GROWTH*

H2: *JANUARY HOUSING SALES DROP, REALTY GROUP SAYS*

Both these headlines belong to the class "HOUSING". After preprocessing, H1 is left with 4 terms, while H2 has 6 terms. Table 3.1 shows the terms (and

Table 3.1: Illustration of the hypothesis. (Terms are stemmed to their roots)

Entire corpus		H1 *(4 terms)*		H2 *(6 terms)*	
Term	Freq.	Term	Freq.	Term	Freq.
januari	12			januari	**12**
sale	10			sale	10
hous	9	*hous*	**9**	*hous*	9
group	6			group	6
growth	5	growth	5		
baldrig	2	baldrig	2		
drop	2			drop	2
solid	2	solid	2		
realti	1			realti	1

corresponding frequencies in the entire corpus) of both H1 and H2 as well as the list of terms in H1 and H2 combined, in order of their occurrence in the corpus. On proceeding with the general frequent term-based clustering algorithm, it is noticed that `januari` will be chosen as a key term, followed by `sale` and so on. However, there are no classes `januari` or `sale` in the corpus. While applying our hypothesis, we apply the frequent term-based clustering algorithm, to each document separately and try to find a key term in each (although term frequency in the entire corpus is considered). Now, on processing H2 first, `januari` will be chosen as a key term, which again is incorrect. The correct key term, `hous` comes third in H2. On the other hand, processing H1 (which has 4 terms) directly identifies `hous` as the key term, which is correct. Thus, it is easier to identify the key term from a document with less number of terms.

3.2 Preprocessing & Document Representation

Before feeding our corpus of headlines to the clustering algorithms, common preprocessing steps are applied and each feature is separated. We consider only *unigram* features, and use *feature presence* measure to represent in the *vector space model*.

Following are a list of preprocessing steps used in our experiments:

- Removal of non-linguistic elements like URLs (e.g. *http://bit.ly/aDkhG*).
- Removal of stop words. Stop words are relatively common words used in a language. e.g. *the*, *is*, *at*, *which*, *on*.
- Case Normalization. The entire document is converted to either upper or lower case.
- Stem words to their roots, so that grammatical inflections are removed. e.g. *loved*, *lover* and *lovely* are all reduced to the same token *love*. We use Porter Stemming algorithm [20] for this purpose.

In the vector space model all the documents are represented in the corpus in a matrix form. One of rows or columns in then matrix represents the document IDs, and the other represent all unique features occurring in the corpus. The intersection of rows and columns in the matrix represent a weighted measure of the feature in the corresponding document. Two common feature weighing techniques are *Term Frequency*Inverse Document Frequency* (TF*IDF) and *Feature Presence*. In preliminary experiments, we found that considering *term frequency*inverse document frequency* had an adverse effect as the total dimensionality of the corpus was substantially low.

Feature Presence (FP) is a straight forward method of feature weighing where the number of times a term/feature occurs in the document is considered. A variant of this technique is to use a binary value at the intersections of documents and terms. In this case, instead of considering the term count, the FP value is set to 1 if a certain terms occurs in the document (irrespective of the number of times it occurs), or 0 if it doesn't. We use the binary feature presence metric.

3.3 The Heuristics Algorithms

We built two heuristic algorithms - Frequent Term *plus*, which is the heuristic version of the frequent term-based clustering algorithm and Frequent Noun *plus*, which has the only modification that all non-nouns are removed in preprocessing.

While implementing the heuristics in clustering algorithms, the number of terms (after preprocessing) in each document is counted. The documents are then arranged in reverse order of number of terms in each. For each document, an attempt is made to identify the key term. Whenever a key term is found, all unclustered documents containing the key term are gathered to form a cluster. While finding a key term, each term is checked if it occurs more in unclustered documents. This is followed till the third most occurring term in a document.

Both the Frequent Term *plus* and Frequent Noun *plus* clustering algorithms are shown next. Figure 3.1 shows the flowchart for Frequent Term *plus* clustering algorithm.

Algorithm 3.1 Frequent Term *plus* algorithm

Require: *featureList*; {List of features and their frequencies in the corpus}
Require: *documentList*; {List of documents and the number of features contained in each}

```
1: reverseSort(documentList);
2: for all document in documentList do
3:    sort(document) {Sort terms in document in order of frequency in corpus}
4:    repeat
5:       Select a term from the document
6:       if countTermIn(term, unclusteredDocs) > countTermIn(term, clusteredDocs) then
7:          createCluster(term)
8:          update(unclusteredDocs)
9:       else
10:         Skip term
11:      end if
12:   until Key term found or 3 iterations over
13: end for
14: applyRefinement()
```

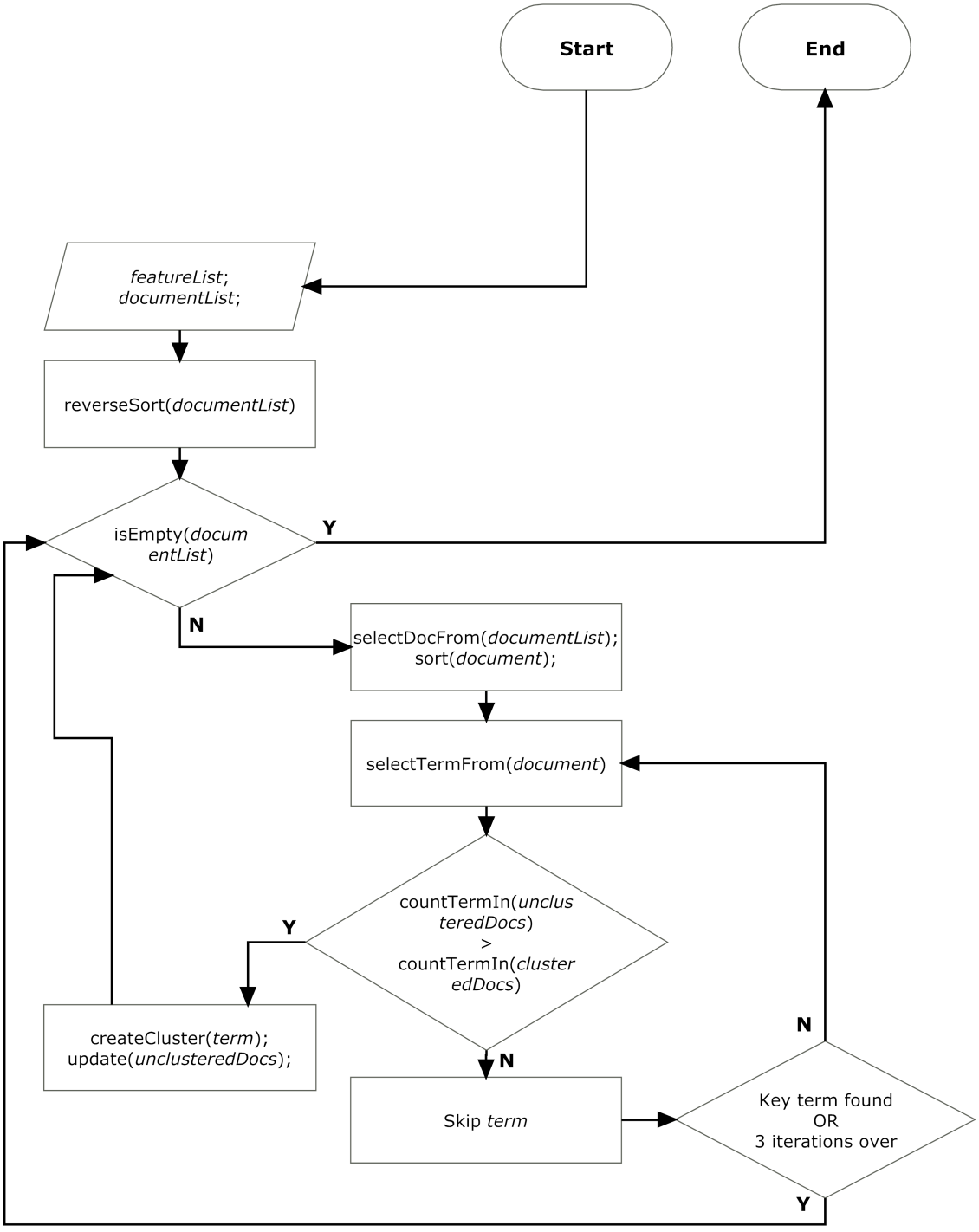


Figure 3.1: Flowchart for Frequent Term *plus* algorithm

Algorithm 3.2 Frequent Noun *plus* algorithm

Require: *nounList*; {List of noun features and their frequencies in the corpus}
Require: *documentList*; {List of documents and the number of noun features contained in each}

```
1: reverseSort(documentList);
2: for all document in documentList do
3:    sort(document) {Sort noun terms in document in order of frequency in corpus}
4:    repeat
5:       Select a noun from the document
6:       if countNounIn(noun, unclusteredDocs) > countNounIn(noun, clustered-
         Docs) then
7:          createCluster(noun)
8:          update(unclusteredDocs)
9:       else
10:         Skip noun
11:      end if
12:   until Key term found or 3 iterations over
13: end for
14: applyRefinement()
```

3.4 Refinement

Refinement is applied to the clustering, produced by an algorithm, to possibly increase the quality of the clusters. In out experiments, all clusters with three or less number of documents are marked as unclustered, and these documents assigned to the most similar cluster. Further, k-means algorithm is applied to the remaining clusters, taking them as the initial seeds.

Algorithm 3.3 Refinement algorithm: applyRefinement()

Require: *clusters*; {Initial clustering produced by clustering algorithms}

```
1: for all cluster in clusters do
2:    if size(cluster) <= 3 then
3:       unclusteredDocs ← breakCluster(cluster)
4:       removeCluster(cluster, clusters)
5:    end if
6: end for
7: for all document in unclusteredDocs do
8:    assignToClosestCluster(clusters)
9: end for
10: kMeans(clusters)
```

Chapter 4

Evaluation

The evaluation technique used in this study focuses on the overall quality of the clusters produced by the algorithm. This is essentially an external measure of cluster quality, as indicated by Steinbach et al. [23] and in [15], which requires the corpus to be pre-categorized into classes by a human. We evaluate our heuristic algorithms using five different datasets, and compare the results with general frequent term-based clustering (as well as the frequent noun version) and traditional k-means algorithm. Comparisons are also made with few well known related studies.

4.1 The Datasets

We use five different datasets to evaluate the results of our clustering algorithms. Since our study was motivated by the unique nature of news headlines, the first two corpuses (Reuters343 and Reuters2388) consists purely of news headlines. Both these datasets were created by extracting headlines from the 90 category split of Reuters-21578 corpus used by Joachims [8]. The other three corpora namely CICLing-2002, Hep-ex, and KnCr, were created from scientific abstracts, and have been used previously for short text clustering ([17], [22]).

Reuters343

This dataset contains a total of 343 headlines split into 26 classes. Although the Reuters corpus is quite large, many headlines are redundant (even if the news post itself is not). This dataset is hand manipulated to remove redundant headlines, and ensured that each headline contains a key term. Further, all headlines in a particular class are related to a single topic (e.g. *coconut*). This will be our primary dataset. Table 4.1 is the list of all classes in the dataset, and the number of headlines contained in each. Table 4.2 shows a few headlines from the dataset and their corresponding class.

Table 4.1: List of classes in Reuters343 and Reuters2388 datasets and corresponding class distributions.

Sl. No.	Class	Reuters 343	Reuters 2388	Sl. No.	Class	Reuters 343	Reuters 2388
1	gold	24	94	14	tin	12	18
2	coffee	23	111	15	palm-oil	11	30
3	alum	22	35	16	zinc	10	21
4	ship	21	197	17	cotton	10	39
5	sugar	21	126	18	retail	8	23
6	trade	20	369	19	money	8	538
7	wheat	19	212	20	soybean	8	78
8	cocoa	19	55	21	gas	8	37
9	corn	19	181	22	housing	7	16
10	barley	15	37	23	rapeseed	5	18
11	copper	14	47	24	coconut	5	4
12	rubber	13	37	25	fuel	5	13
13	jobs	12	46	26	cattle	4	6

Table 4.2: Example of headlines from the Reuters343 dataset.

Headline	Topic
U.S. cotton certificate expiration date extended	cotton
China trying to increase cotton output, paper says	cotton
January housing sales drop, realty group says	housing
Quebec February housing starts fall	housing
Pakistan not seen as major wheat exporter	wheat
Weather hurting Yugoslav wheat - USDA Report	wheat

Reuters2388

This dataset contains a total of 2,388 headlines split into 26 classes (the same classes as the Reuters343 headlines). Headlines from all news posts in the corresponding classes of the 90 category split of Reuters-21578 corpus are considered in this dataset, without any hand manipulation. Again, all headlines in a particular class are related to a single topic (although the key term may not be present explicitly in the headline). Table 4.1 is the list of all classes in the dataset, and the number of headlines contained in each.

CICLing-2012

This is a small corpus consisting of 48 abstracts in the domain of computational linguistics collected from the CICLing 2002 conference. This corpus has 4 classes of 48 abstracts and the abstracts are evenly distributed among the 4 classes which is as follows: $\{11, 15, 11, 11\}$.

Hep-ex

This corpus contains 2,922 abstracts collected by the University of Jaén, Spain on the domain of Physics from the data server of the CERN. These abstracts are related to 9 categories. The distribution of the abstracts among the 9 classes is highly uneven and is as follows: $\{2623, 271, 18, 3, 1, 1, 1, 1, 1\}$

KnCr

This corpus contains abstracts from the cancer domain of the medical field and collected from the MEDLINE documents [19]. It contains 900 abstracts and they are related to 16 categories. The abstracts are distributed among the 16 classes as follows: $\{169, 160, 119, 99, 66, 64, 51, 31, 30, 29, 22, 20, 14, 12, 8, 6\}$

4.2 Cluster Quality Measures

The three cluster quality measures used to interpret the output of the algorithms are discussed next. The formulas illustrated as defined for the set of classes $C = \{c1, c2, ..., c_i\}$ and set of clusters $K = \{k1, k2, ..., k_j\}$. n is the total number of documents in the corpus, n_i the size of class i, n_j the size of cluster j and n_{ij} is the number of documents of class i in cluster j.

Purity

To compute purity, each cluster is assigned to the class which is most frequent in the cluster, and then the accuracy of this assignment is measured by counting the number of correctly assigned documents and dividing by the total number of documents in the corpus, as shown in [15]. Mathematically, purity of a clustering K with respect to pre-labeled classes C is calculated as:

$$purity(K, C) = \frac{1}{n} \sum_{j} \max_{i} |k_j \cup c_i|$$

Purity values range between 0 and 1, with the purity measure of a good clustering approaching 1 and vice versa. It has also been pointed out in [15] that high purity can be easily achieved if the number of clusters is large, or purity is 1 if each document gets its own cluster. Thus, purity measure cannot alone be used to trade off the quality of the clustering against the number of clusters.

Entropy

Entropy, a measure introduced by Shannon [21], is an external evaluation method which accounts for the 'goodness' of a flat clustering, as indicated by Steinbach et al. [23]. However, just like purity measure, maximum entropy is achieved when each cluster contains only one document.

The calculate the entropy measure of a clustering K, we first calculate p_{ij}, the probability that a member of cluster j belongs to class i. The total entropy of the

clustering is calculated as below:

$$E_K = \frac{1}{n} \sum_j n_j \sum_i -p_{ij} log(p_{ij})$$

F measure

F measure is another external quality measure which combines the precision and recall ideas from information retrieval. Precision and recall of a cluster for each given class is calculated as $Precision(i, j) = \frac{n_{ij}}{n_j}$ and $Recall(i, j) = \frac{n_{ij}}{n_i}$

F measure of cluster j and class i is calculated using the formula given below:

$$F(i, j) = \frac{2 * Recall(i, j) * Precision(i, j)}{Recall(i, j) + Precision(i, j)}$$

The overall F measure of the clustering is calculated using the expression suggested by Larsen and Aone [11], as illustrated below:

$$F_K = \frac{1}{n} \sum_i n_i \max_j F(i, j)$$

Chapter 5

Results

5.1 Results

For evaluation of our proposed heuristics, we implemented four algorithms, in addition to k-means clustering algorithm. The five algorithms in comparison are:

- K-means (*regular*)
- Frequent-term (*regular*) [FT]
- Frequent-term *plus* (*heuristic*) [FT*plus*]
- Frequent-nouns (*regular*) [FN]
- Frequent-nouns *plus* (*heuristic*) [FN*plus*]

The results discussed in this section were obtained on performing the experiments on an Intel Core i3 machine with clock frequency 2.1GHz and 1GB of primary memory.

To start with, Table 5.1 summarizes the total clusters generated, total correct clusters, incorrect clusters and total number of documents correctly clustered by each of the five algorithms with the Reuters343 dataset. FT*plus* and FN*plus* algorithms both correctly identified 22 topics out of 26 in the dataset, which was the highest. On the other hand, k-means algorithm could identify only 17 topics

Table 5.1: Summary of clusters generated. (Reuters343 dataset)

Clustering algorithm	k-means	FT	FT*plus*	FN	FN*plus*
Total clusters generated	26	29	34	28	33
Correct clusters	17	19	**22**	20	**22**
Incorrect clusters	9	10	*12*	8	11
Number of documents correctly clustered	238	226	255	241	**272**

in its best case. In terms of number of documents correctly clustered, FN*plus* had the most.

Table 5.2 and Figure 5.1 summarizes the cluster quality evaluation measures obtained for each of the five algorithms using the Reuters343 dataset. To determine the quality of the clusters produced in terms of these values, note that for purity measure, the closer it is to 1, the better is the clustering. Similarly, for F measure, the larger the F measure, the better is the cluster quality. However, in case of entropy, it is just the opposite. The lesser the entropy measure, the better is the clustering produced. In short, the cluster quality is directly proportional to the purity and F measure values, wheras it is inversely proportional to the entropy measure.

Similarly, Tables 5.3 to 5.6 and Figures 5.2 to 5.5 summarizes the cluster quality evaluation measures obtained for each of the algorithms using the Reuters2388, CICLing-2002, Hep-ex and KnCr datasets respectively. For the later three datasets it was not possible to evaluate the FN and FN*plus* algorithms, as these datasets were available in preprocessed (and stemmed) form.

As evident in Figure 5.1 (using Reuters343 dataset), frequent noun *plus* produced the best cluster quality. Both the heuristic algorithms, frequent term *plus* and frequent noun *plus* showed improved performance compared to the regular frequent term and frequent noun clustering algorithms. Moreover, k-means clustering produced the poorest cluster quality. On observing the evaluation measures, it has been noticed that the purity and F measure values increase in the order

Table 5.2: Cluster quality measures. (Reuters343 dataset)

Clustering algorithm	k-means	FT	FT*plus*	FN	FN*plus*
Purity	0.57	0.63	0.64	0.72	**0.83**
Entropy	1.09	0.83	0.59	0.55	**0.27**
F measure	0.50	0.62	0.68	0.76	**0.86**

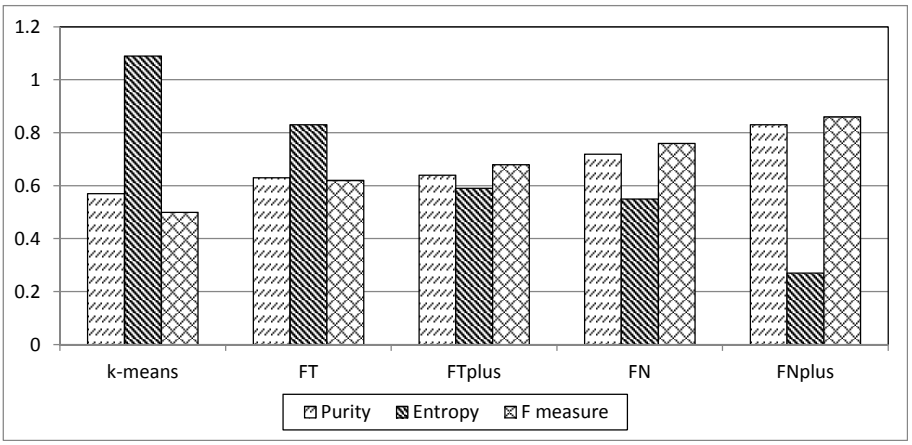


Figure 5.1: Cluster quality measures. (Reuters343 dataset)

Table 5.3: Cluster quality measures. (Reuters2388 dataset)

Clustering algorithm	k-means	FT	FT*plus*	FN	FN*plus*
Purity	0.4	0.34	0.3	0.34	0.28
Entropy	1.51	1.12	1.01	0.93	0.91
F measure	0.44	0.4	0.38	0.42	0.38

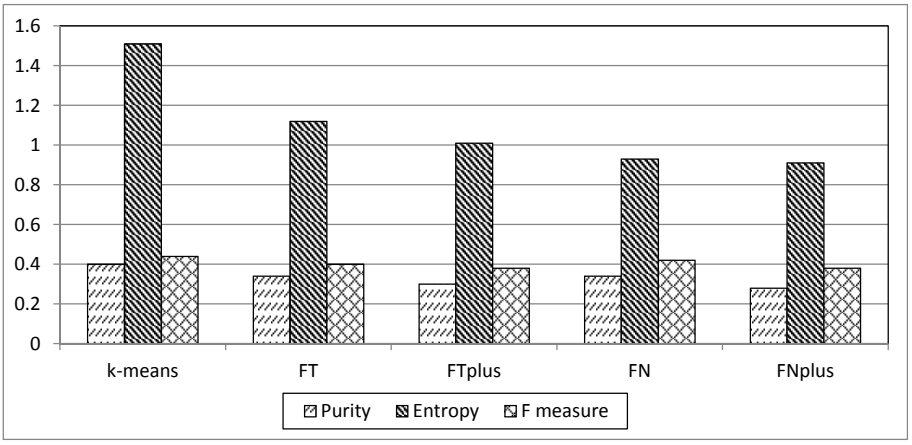


Figure 5.2: Cluster quality measures. (Reuters2388 dataset)

Table 5.4: Cluster quality measures. (CICLing-2002 dataset)

Clustering algorithm	k-means	FT	FT*plus*
Purity	0.58	0.52	0.56
Entropy	1.18	1.02	1.17
F measure	0.49	0.42	0.37

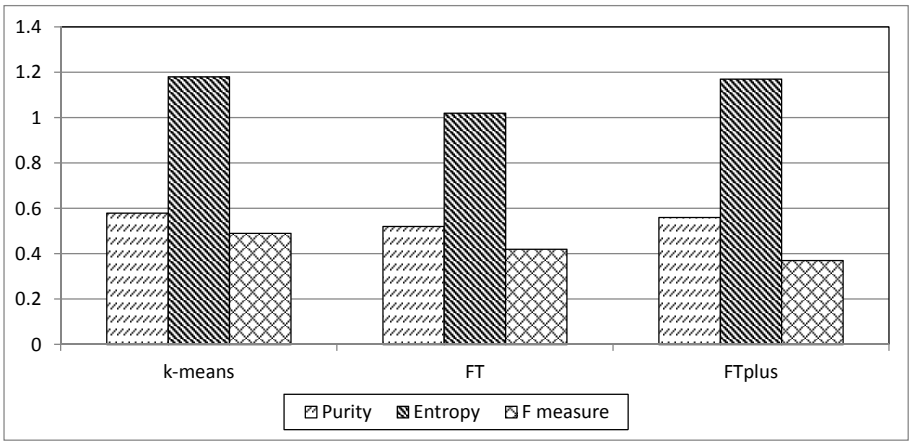


Figure 5.3: Cluster quality measures. (CICLing-2002 dataset)

Table 5.5: Cluster quality measures. (Hep-ex dataset)

Clustering algorithm	k-means	FT	FT*plus*
Purity	0.22	0.07	0.07
Entropy	0.22	0.13	0.1
F measure	0.33	0.13	0.12

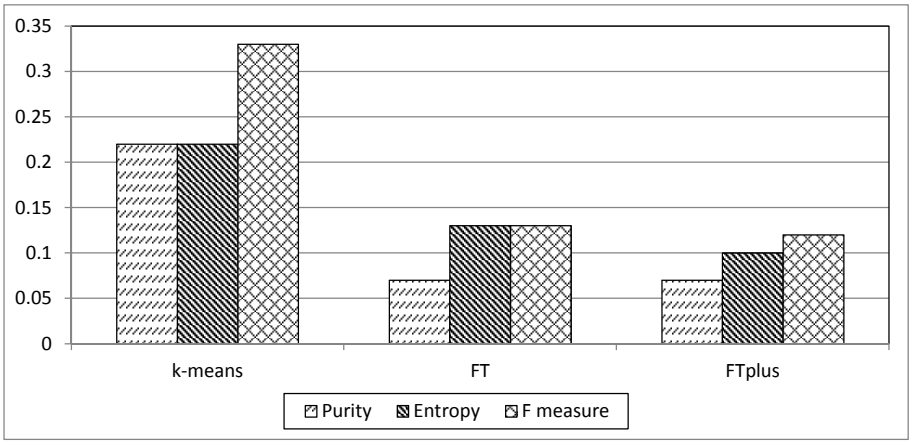


Figure 5.4: Cluster quality measures. (Hep-ex dataset)

Table 5.6: Cluster quality measures. (KnCr dataset)

Clustering algorithm	k-means	FT	FT*plus*
Purity	0.18	0.31	0.12
Entropy	0.82	1.03	0.83
F measure	0.18	0.18	0.14

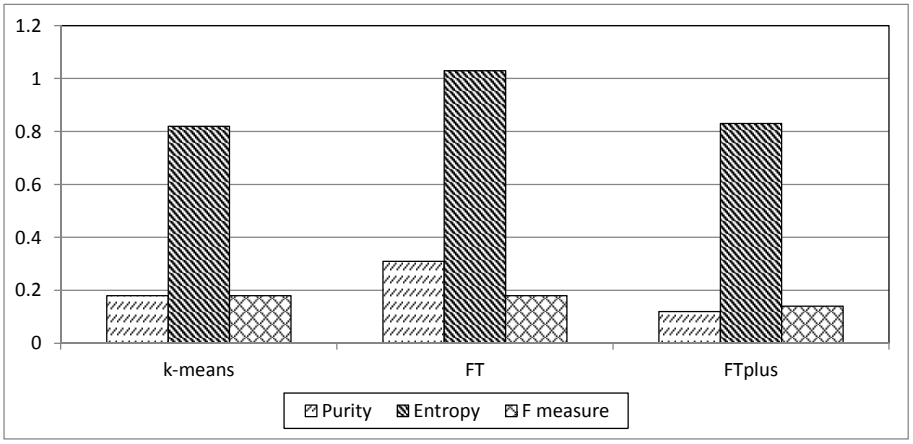


Figure 5.5: Cluster quality measures. (KnCr dataset)

k-means, FT, FT*plus*, FN, FN*plus*, and entropy measure decreases in the same order. The quality of the clusters produced by each of these algorithms can hence be arranged in this order.

In Figure 5.2 (using Reuters2388 dataset), although the entropy measure decreases in the order k-means, FT, FT*plus*, FN, FN*plus*, the purity and F measure values remain almost same in each case. Judging only in terms of entopy value, FN*plus* produces the best clustering. A thing to be noted in this regard is that, the lesser is the entropy value, the less out-of-place documents are in the produced clusters (relate to entropy in chemical atoms).

Using the other three datasets (CICLing-2002, Hep-ex, and KnCr), the results were a little absurd. With the CICLing-2002 dataset (Figure 5.3) and KnCr dataset (Figure 5.5) each of the purity, entropy and F measure values remained almost constant (with a little drop or rise in FT) for all algorithms. In case of the Hep-ex dataset (Figure 5.4) k-means was the best performer in terms of all evaluation measure. In fact, FT and FT*plus* performed worse.

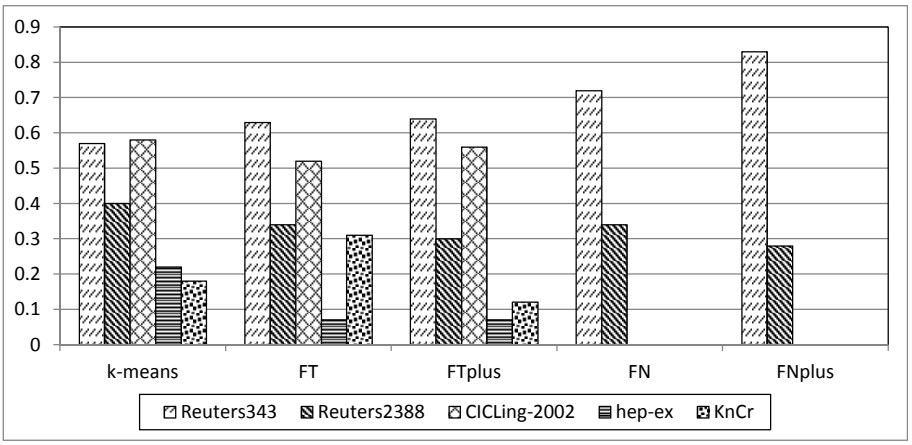


Figure 5.6: Purity comparison on all datasets.

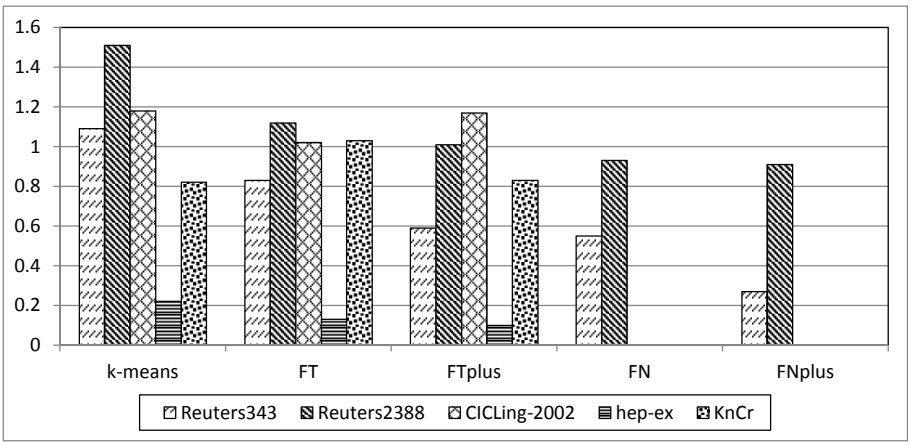


Figure 5.7: Entropy comparison on all datasets.

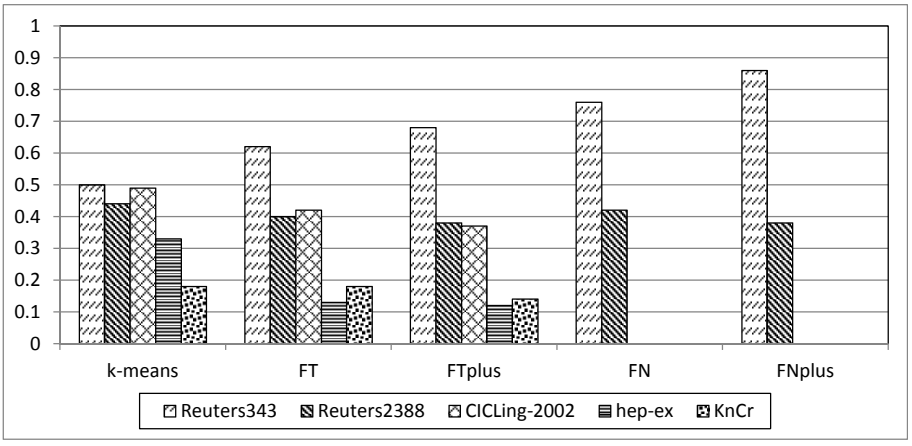


Figure 5.8: F measure comparison on all datasets.

Figure 5.6, 5.7 and 5.8 shows the purity, entropy and F measure scores for all algorithms on all datasets. Apparently, the Reuters343 dataset shows the most eminent results, suggesting FN*plus* to be the best clustering algorithm.

5.2 Discussions

Although the average F measure values in most studies [11] [23] lies around 0.5 and 0.6, our study project a much higher score (using Reuters343 dataset). This may be attributed to two significant dimensions of our study, which makes it different from many related work - *1.* News headlines are essentially small text, compared to larger bodies of text like news posts and reviews. *2.* The corpus used for evaluation is small and hand manipulated. However, Beil et al. [1] reports a F measure similar to that of FT*plus* with their implementation of bisecting k-means on a corpus of papers related to aeronautical system, medicine and information retrieval. They also reported entropy values similar to the ones achieved by us, although on evaluating using the Reuters corpus, the entropy was much higher. Steinbach et al. [23] also reported much lower entropy values in their original implementation of bisecting k-means algorithm. Thus, it can be inferred that our heuristics perform comparably (or even better) with traditional clustering methods as well as other intuitive algorithms.

While the results obtained using the Reuters343 dataset indicated enhanced performance with the heuristic algorithms, the results were not consistent with the Reuters2388 dataset. A possible explanation of this outcome is that, while our heuristics was aimed at identifying the key term corresponding to a class and subsequently building a cluster using it, the Reuters2388 corpus comprises of numerous headlines which do not contain the same key term as other documents in the class (and sometimes do not contain a key term at all). As each document contains a limited number of features, finding a secondary key term is difficult and thus leads to a certain degree of noise.

The absurd cluster quality measures obtained while using the CICLing-2002, Hep-ex and KnCr datasets can be explained as a result of the different nature of these datasets. Unlike news headlines, these datasets comprises of scientific abstracts, which are full-fledged paragraphs with multiple complete sentences. This leads to a higher dimensionality of the corpus. Further, this can also be attributed to the uneven class distribution of documents in these datasets.

Chapter 6

Conclusion

We proposed a novel heuristics which could be incorporated into the regular frequent term-based clustering and frequent noun-based clustering algorithms to produce better results. The heuristics was implemented and compared with traditional k-means, frequent term and frequent noun clustering. Five datasets were used for evaluating these clustering algorithms - Reuters343, Reuters2388 (news headlines), CICLing-2002, Hep-ex and KnCr (scientific abstracts), and the results were interpreted based on three common external cluster quality evaluation measures - purity, entropy and F measure. On investigating the results, the following conclusions were drawn:

1. Using a small noiseless dataset of news headlines (Reuters343) our heuristic clustering algorithms performed at par with, or even better than, few intuitive algorithms like bisecting k-means [23], buckshot and fractionation [5]. The general order of increasing performance was k-means, FT, FT*plus*, FN, FN*plus*, based on all three evaluation measures.

2. With a larger and noisier dataset of news headlines (Reuters2388) the performance variations were mild. However, based on a single evaluation measure (entropy), the general order of increasing performance remained same.

3. While using datasets comprising of scientific abstracts (CICLing-2002, Hep-ex and KnCr) there was no apparent performance increase. In fact, in one

case (using Hep-ex dataset), the cluster quality of frequent term and frequent term *plus* algorithms degraded over k-means.

Future work may include finding a method to decrease the dimensionality of short text corpuses. The heuristics may also be extended to news headlines and description (instead of headlines only) to address the problem of noisy headlines. Further, *word sense disambiguation* can be done as news headlines are indited by experts and they usually follow a variety of linguistic styles. Lastly, the problem of selecting an appropriate cluster descriptor will be addressed.

References

[1] F. Beil, M. Ester, and X. Xu. Frequent term-based text clustering. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 436–442, New York, NY, USA, 2002. ACM.

[2] P. S. Bradley and U. M. Fayyad. Refining initial points for k-means clustering. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 91–99, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[3] C. Carpineto, S. Osiński, G. Romano, and D. Weiss. A survey of web clustering engines. *ACM Comput. Surv.*, 41(3):17:1–17:38, July 2009.

[4] D. R. Cutting, D. R. Karger, and J. O. Pedersen. Constant interaction-time scatter/gather browsing of very large document collections. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '93, pages 126–134, New York, NY, USA, 1993. ACM.

[5] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/gather: a cluster-based approach to browsing large document collections. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '92, pages 318–329, New York, NY, USA, 1992. ACM.

[6] S. Guha, R. Rastogi, and K. Shim. Rock: A robust clustering algorithm for categorical attributes. In *In Proc.ofthe15thInt.Conf.onDataEngineering*, 2000.

[7] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. The Morgan Kaufmann Series in Data Management Systems. Elsevier, 2006.

[8] T. Joachims. Text categorization with support vector machines: Learning with many relevant features, 1998.

[9] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(7):881–892, July 2002.

[10] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 170–178, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[11] B. Larsen and C. Aone. Fast and effective text mining using linear-time document clustering. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '99, pages 16–22, New York, NY, USA, 1999. ACM.

[12] U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, Dec. 2007.

[13] Y. S. Maarek, R. Fagin, I. Z. Ben-Shaul, and D. Pelleg. Ephemeral document clustering for web applications. Technical report, IBM RESEARCH REPORT RJ 10186, 2000.

[14] P. Makagonov, M. Alexandrov, and A. Gelbukh. Clustering abstracts instead of full texts. *In: Text, Speech, Dialog, LNAI N 3206, Springer*, 2004:129–135, 2004.

[15] C. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[16] P. Pantel and D. Lin. Document clustering with committees. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '02, pages 199–206, New York, NY, USA, 2002. ACM.

[17] D. Pinto, J.-M. Benedí, and P. Rosso. Clustering narrow-domain short texts by using the kullback-leibler distance. In A. F. Gelbukh, editor, *Proceedings of the*

8th International Conference on Computational Linguistics and Intelligent Text Processing, volume 4394 of *CICLing'07*, pages 611–622, Berlin, Heidelberg, 2007. Springer.

[18] D. Pinto, H. Jiménez-Salazar, and P. Rosso. Clustering abstracts of scientific texts using the transition point technique. In *Proceedings of the 7th international conference on Computational Linguistics and Intelligent Text Processing*, CICLing'06, pages 536–546, Berlin, Heidelberg, 2006. Springer-Verlag.

[19] D. Pinto and P. Rosso. Kncr: A short-text narrow-domain sub-corpus of medline. In *Proc. of the TLH 2006 Conference*, Advances in Computer Science, pages 266–269, 2006.

[20] M. F. Porter. Readings in information retrieval. chapter An algorithm for suffix stripping, pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.

[21] C. E. Shannon. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(1):3–55, Jan. 2001.

[22] P. Shrestha, C. Jacquin, and B. Daille. Clustering short text and its evaluation. In *Proceedings of the 13th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing'12, pages 169–180, Berlin, Heidelberg, 2012. Springer.

[23] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *In KDD Workshop on Text Mining*, 2000.

[24] O. Zamir and O. Etzioni. Web document clustering: a feasibility demonstration. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 46–54, New York, NY, USA, 1998. ACM.

[25] O. Zamir and O. Etzioni. Grouper: a dynamic clustering interface to web search results. In *Proceedings of the eighth international conference on World Wide Web*, WWW '99, pages 1361–1374, New York, NY, USA, 1999. Elsevier North-Holland, Inc.