Learning and Nonlinear Models - Journal of the Brazilian Society on Computational Intelligence (SBIC), Vol. 12, Iss. 1, pp. 17-37, 2014

ANALYSIS OF DOMAIN INDEPENDENT STATISTICAL KEYWORD EXTRACTION METHODS FOR INCREMENTAL CLUSTERING

Rafael Geraldeli Rossi[1], **Ricardo Marcondes Marcacini**[1,2], **Solange Oliveira Rezende**[1]
[1]Institute of Mathematics and Computer Science - University of Sao Paulo
[2]Federal University of Mato Grosso do Sul
{ragero,solange}@icmc.usp.br, ricardo.marcacini@ufms.br

Abstract – Incremental clustering is a very useful approach to organize dynamic text collections. Due to the time/space restrictions for incremental clustering, the textual documents must be preprocessed to maintain only their most important information. Domain independent statistical keyword extraction methods are useful in this scenario, since they analyze only the content of each document individually instead of all document collection, are fast and language independent. However, different methods have different assumptions about the properties of keywords in a text, and different methods extract different set of keywords. Different ways to structure a textual document for keyword extraction can also modify the set of extracted keywords. Furthermore, extracting a small number of keywords might degrade the incremental clustering quality and a large number of keywords might increase the clustering process speed. In this article we analyze different ways to structure a textual document for keyword extraction, different domain independent keyword extraction methods, and the impact of the number of keywords on the incremental clustering quality. We also define a framework for domain independent statistical keyword extraction which allows the user set different configurations in each step of the framework. This allows the user tunes the automatic keyword extraction according to its needs or some evaluation measure. A thorough experimental evaluation with several textual collections showed that the domain independent statistical keyword extraction methods obtains competitive results to the use of all terms or even selecting terms analyzing all the text collection. This is a promising evidence that favors computationally efficient methods for preprocessing in text streams or large textual collections.

Keywords – Automatic keyword extraction, incremental clustering, text preprocessing

1 Introduction

Organizing dynamic text collection, in which new textual documents are constantly published, is a current challenge [1, 2]. Traditional clustering approaches consider that the text collection is static, and the clustering process is repeated when new documents are inserted into the textual collection. This is computationally costly, mainly for situations when frequent updates in the clustering are required. Incremental clustering methods are useful in this context since they allow to update existing clusters instead of generating them for each new document or set of documents [3].

The process of incremental clustering might become slow, due to the calculation of similarities considering the large number of different words contained in a (dynamic) text collection, and might consequently consume a large amount of memory. Moreover, the documents of a domain share several general words, which can affect the similarity computation and produce spurious results.

To solve or reduce the problems mentioned above, only the document keywords, which describe or summarize the document content concisely, should be used instead of all the words [4]. Unfortunately, most of the documents do not have associated keywords. Thus, methods to automatically extract keywords are necessary, since the manual extraction of keywords from the documents in a collection or stream is unfeasible.

Some keyword extraction methods need to analyze the entire document collection. These methods are considered "domain dependent". Some examples of these methods are: TF-IDF (Term Frequency - Inverse document Frequency) [5], Mutual Information [6] and Log-likelihood [7]. The domain dependent methods are not feasible for streams or large document collections, since they need to keep all the words in the memory to extract keywords. Furthermore, most domain dependent methods are supervised, i.e., they require labels for the documents, which are not provided in text clustering tasks.

On the other hand, there are keyword extraction methods that analyze only the content of each document individually. These methods does not require labeled documents and does not need to analyze the entire document collection, i.e., they are "domain independent". Examples of these methods are: TF-ISF (Term Frequency - Inverse Sentence Frequency) [8], CSI (Co-occurrence Statistical Information) [9], TextRank [10], and Eccentricity-Based [11]. The keyword extraction from single documents is very useful for i) large collections, in which the load of the entire collection in memory to extract the keywords is sometimes impossible, and ii) incremental collections, in which the analysis of the entire collection to extract keywords for each new document is unfeasible. Thus, in this article we focus on domain independent statistical keyword extraction methods which satisfy the requirements of incremental clustering applications. Moreover, these methods can be applied to documents written in any language and are faster than linguistic methods.

Different sets of keywords are extracted by different domain independent keyword extraction methods since they have different assumptions about the properties of the keywords. Furthermore, different ways to structure a text for keyword extraction can also change the extracted sets of keywords. Thus, an analysis about which way to structure texts and which method is more appropriate for the incremental clustering task is necessary. Moreover, a study about the number of keywords to improve or maintain the quality of the incremental clustering is also necessary, since the use of a little number of keywords might not maintain the quality of the incremental clustering and a large number of keywords might not have impact in the speed of the process.

In summary, our contributions are threefold:

- We analyze different domain independent methods for keyword extraction and analyze the impact of the different number of keywords extracted from documents in the quality of the incremental clustering.
- We define a set of steps which must be applied for domain independent keyword extraction. We organized these steps in a framework, allowing the user sets different configurations in each step to tune the keyword extraction.
- We show that the domain independent statistical keyword extraction methods obtains similar results to the domain-dependent methods through a statistical analysis upon the experimental evaluation with several textual collections. This is a promising evidence that favors computationally efficient methods for preprocessing in large textual collections.

The remainder of this article is organized as follows. Section 2 describes the algorithm used for incremental clustering. Section 3 details the proposed framework and the domain independent statistical keyword extraction methods used in this article. Section 4 presents the incremental clustering results obtained by different domain independent statistical keyword extraction methods applied to different text structures with different number of keywords. Finally, Section 5 presents the conclusions and future work.

2 Incremental Clustering

Incremental clustering is based on the assumption that the allocation of a new document in an existing cluster structure can be done without storing all documents in main memory [12]. Formally, given a sequence of documents $d_1, d_2, ..., d_n$, a partition C_{h+1} can be generated considering a previous partition C_h and the new document d_i [13]. Moreover, the incremental clustering algorithms store only the cluster representatives (centroids) in main memory [3].

Leader-Follower algorithm[1] [14], described in Algorithm 1, is one of the simplest and commonly used methods for incremental clustering [1, 3]. In this algorithm, the similarity of a new document with the existing clusters is calculated (lines 4–10). If the similarity is higher than a user's threshold (m), the new document is allocated to the closest cluster (lines 11–13). Otherwise, a new cluster is created for the new document (lines 14–18). Usually the centroids of the clusters are used to compute the similarities. When a new document is allocated to a cluster, the respective centroid is adjusted (lines 13 and 17).

Algorithm 1: Leader-Follower (Incremental Clustering Algorithm)

Input :
$X_{inc} = \{x_1, x_2, ...\}$: text documents source
α: Minimum dissimilarity threshold
Output:
$P = \{G_1, G_2, ..., G_k\}$: data partition with k clusters

1 **foreach** *new document* $x \in X_{inc}$ **do**
  // searching the nearest cluster
2 $MinDist \leftarrow +\infty$
3 $G_{sel} \leftarrow \emptyset$
4 **foreach** *cluster* $G_i \in P$ **do**
5 $Dist \leftarrow d(x, G_i)$
6 **if** $Dist < MinDist$ **then**
7 $MinDist \leftarrow Dist$
8 $G_{sel} \leftarrow G_i$
9 **end**
10 **end**
  // clustering the document x
11 **if** $MinDist < \alpha$ **then**
12 allocate x in the nearest cluster G_{sel}
13 adjust the cluster centroid of the G_{sel}
14 **else**
15 create new cluster G_{new} into data partition P
16 allocate x in the cluster G_{new}
17 adjust the cluster centroid of the G_{new}
18 **end**
19 **end**

[1] A Java implementation of the Leader-Follower algorithm used in this article is available at `http://sites.labic.icmc.usp.br/sket/supplement/leader-follower.java`

Due do the time/space restrictions for incremental clustering, it is important to preprocess the documents to maintain only the most important information contained into them. In this way, domain independent keyword extraction methods are useful in this scenario.

3 Domain Independent Statistical Keyword Extraction

Domain Independent Statistical Keyword Extraction methods can be applied using the same set of steps. Here we propose a framework for domain independent statistical keyword extraction which allows the user tune the keyword extraction through the setting of pre-processing steps, keyword extraction methods, and number of keywords.

The proposed framework contains 4 steps: i) preprocessing and structuring the textual document, ii) generating scores for each term extracted from the document, iii) sorting the term scores, and iv) extracting the first k terms of the sorted term scores as keywords. Figure 1 presents the inputs and outputs generated in each step of the proposed framework.

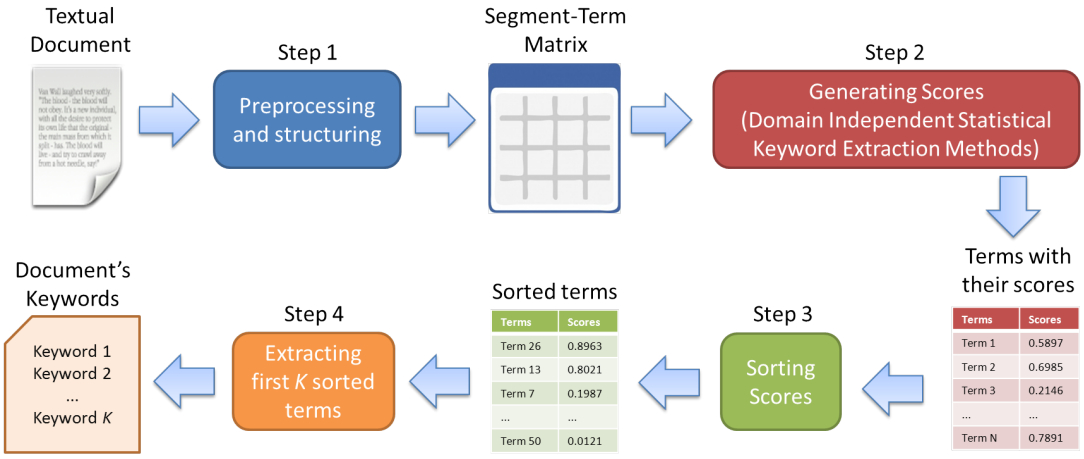


Figure 1: Proposed framework for statistical keyword extraction.

In the next subsections we detail each step of the proposed framework for domain independent statistical keyword extraction and a general overview to illustrate the differences among different keyword extraction methods. We implemented a tool in Java language referred to Statistical Keyword Extraction Tool[2] (SKET), which contains all the steps and methods presented in this article.

3.1 Preprocessing and Structuring Textual Document

Keywords extraction methods consider as input a text represented in a structured format. A segment-term matrix can be used to structure texts for any domain independent keyword extraction methods. In this matrix, each row corresponds to a segment of the text, such as as sentence, paragraph or window of words, and each term corresponds to a column. A term can be a single word or a set/sequence of words [5, 15]. We denote the set of terms as $\mathcal{T} = \{t_1, t_2, \ldots, t_N\}$ and the set of segments as $\mathcal{S} = \{s_1, s_2, \ldots, s_M\}$. If a term t_i occurs in a segment s_j, the value 1 is assigned to the corresponding cell of the matrix (o_{t_i,s_j}) and 0 otherwise. Table 1 shows an example of a segment-term matrix.

Table 1: Segment-term matrix.

	Term 1	*Term 2*	$\cdots$	*Term N*
Segment 1	o_{t_1,s_1}	o_{t_2,s_1}	$\cdots$	o_{t_N,s_1}
Segment 2	o_{t_1,s_2}	o_{t_2,s_2}	$\cdots$	o_{t_N,s_2}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Segment M	o_{t_1,s_M}	o_{t_2,s_M}	$\cdots$	o_{t_N,s_M}

Common Text Mining preprocessing steps [16], as the removal of stopwords and unnecessary characters, and word simplification, can be used to generate the segment-term matrix. The cut of terms which occur below a minimum number of segments can also be performed [11].

The segment-term matrix can be directly submitted to a set of keyword extraction methods [8, 9] or be used to generate a graph-based representation, which are used by another set of methods [10, 11]. A graph is defined as $G = \langle \mathcal{V}, \mathcal{E}, \mathcal{W} \rangle$, in which $\mathcal{V}$ represents the set of vertices, $\mathcal{E}$ represents the set of edges among the vertices and $\mathcal{W}$ represents the weights of the edges. The terms of the segment-term matrix are the vertices of the graph, i.e., $\mathcal{V} = \mathcal{T}$, and an edge between the terms t_i and t_j (e_{t_i,t_j}) is generated considering their frequency of cooccurrence ($co\text{-}occur(t_i, t_j)$). The weight of the edge is defined according to the

[2]Statistical Keyword Extraction Tool is available at `http://sites.labic.icmc.usp.br/sket/sket.zip`.

characteristic of the algorithm. Some algorithms consider that the edge weight must be small for terms with high co-occurrence. In this case, Equation 1 is used to set the edge weight. On the other hand, some algorithms consider that the edge weight must be high for terms with high co-occurrence. In this case, Equation 2 is used.

$$w_{t_i t_j} = \begin{cases} \frac{1}{co\text{-}occur(t_i,t_j)} & \text{if } e_{t_i,t_j} \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$w_{t_i t_j} = \begin{cases} 1 - \frac{1}{co\text{-}occur(t_i,t_j)} & \text{if } e_{t_i,t_j} \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

3.2 Generating a Score for Each Term of the Segment-Term Matrix

The difference among the keyword extraction methods is in the way that they compute the scores for the terms considering the segment-term matrix. Here we evaluated 5 statistical methods to compute the scores of the terms: (i) Most Frequent (MF), (ii) Term Frequency - Inverse Sentence Frequency (TF-ISF) [8], (iii) Co-occurrence Statistical Information (CSI) [9], (iv) Eccentricity-Based [11] and (v) TextRank [10]. The first three methods consider solely the segment-term matrix and the last two methods consider a graph representation as input. The next subsections present details about these methods.

3.2.1 Most Frequent

A simple measure to automatically extract keywords is to consider the most frequent (MF) terms as keywords. The score of the term t_i is obtained by counting the number of occurrences of the term in the segment-term matrix, i.e.:

$$score(t_i) = freq(t_i) = \sum_{s_j \in \mathcal{S}} o_{t_i,s_j}. \tag{3}$$

3.2.2 Term Frequency - Inverse Sentence Frequency

TF-ISF (Term Frequency - Inverse Sentence Frequency) measure is proposed in [8]. The basic idea of TF-ISF is to determine the score of a term according to its frequency and its distribution through the sentences of the document. The score of a term decreases if a term occurs in a large number of sentences in the document, since this can be a common term and do not characterize the content of the document. This method was designed to consider the distribution of words in sentences, but here we also will apply TF-ISF considering the distribution of words in windows of terms. The TF-ISF score for a term t_i is:

$$score(t_i) = TF\text{-}ISF(t_i) = freq(t_i) * \log\left(\frac{|\mathcal{S}|}{freq(t_i)}\right). \tag{4}$$

3.2.3 Co-occurrence Statistical Information

CSI (Co-occurrence Statistical Information) measure [9] obtain scores for words using χ^2 measure [17]. χ^2 measures how much the observed frequencies are different from the expected frequencies. For the keyword extraction problem, the χ^2 measure for a term t_i is:

$$\chi^2(t_i) = \sum_{t_j \in \mathcal{T}} \frac{(co\text{-}occur(t_i,t_j) - co\text{-}occur(t_i)p(t_j))^2}{co\text{-}occur(t_i)p(t_j)}, \tag{5}$$

in which $p(t_j)$ is the probability of term t_j occurs in the segment-term matrix, $co\text{-}occur(t_i)$ is the total number of co-occurrences of the term t_i with terms $t_j \in \mathcal{T}$, $co\text{-}occur(t_i,t_j)$ corresponds to the observed frequency, and $co\text{-}occur(t_i)p(t_j)$ corresponds to the expected frequency.

Since a document is composed by sentences of varied length, terms that appear in long sentences tend to co-occur with more terms. Then, the authors of the CSI measure redefined $p(t_j)$ as the sum of the total number of terms in sentences in which t_j appears divided by the total number of terms in the document, and $co\text{-}occur(t_i)$ as the total number of terms in sentences in which t_i appears. Moreover, the value of the χ^2 measure can be influenced by non-important but adjunct terms. To make the method more robust to this type of situation, the authors of the CSI measure subtracts from the $\chi^2(t_i)$ the maximum χ^2 value for any $t_j \in \mathcal{T}$, i.e.:

$$score(t_i) = CSI(t_i) = \chi^2(t_i) - \arg\max_{t_j \in \mathcal{T}} \left\{ \frac{(co\text{-}occur(t_i,t_j) - co\text{-}occur(t_i)p(t_j))^2}{co\text{-}occur(t_i)p(t_j)} \right\}. \tag{6}$$

Using Equation 6, a low $CSI(t_i)$ value is generated if t_i co-occurs selectively with only one term. The measure presents a high value if t_i co-occurs selectively with more than one term, i.e., the term is relatively important in the document.

3.2.4 Eccentricity-Based

Eccentricity is a centrality measure, i.e., a measure which determines the importance of a vertice in a graph [11]. According to eccentricity measure, a node is central/important if its distance to the most distance vertice is small. The distance between a term t_i and term t_j ($d(t_i, t_j)$) is the sum of the edge weights on the shortest path from t_i to t_j in G. Thus, the eccentricity of a term t_i is:

$$score(t_i) = \epsilon(t_i) = \arg\max_{t_j \in \mathcal{T}} \{d(t_i, t_j)\} \ . \tag{7}$$

Equation 1 is used to weight the graph edges since the intent is to measure the distance of two nodes. In this case, terms with high co-occurence have a lower distance between them. We highlight that any other centrality measure can be used in place of eccentricity such as closeness, betweenness and clustering coefficient [18]. For instance, [19] used the vertice contribution to maintain a term network with a small world characteristic as centrality measure.

3.2.5 TextRank

TextRank algorithm [10] is based on PageRank algorithm [20], which defines the importance of a vertice in the graph considering the importance of its connected objects. The score of a term t_i using TextRank algorithm is:

$$score(t_i) = TR(t_i) = (1 - \lambda) + \lambda * \sum_{e_{t_i,t_j} \in \mathcal{E}} \frac{w_{t_i,t_j}}{\sum_{e_{t_j,t_k} \in \mathcal{E}} w_{t_j,t_k}} TR(t_j) \ , \tag{8}$$

in which λ is a damping factor that can be set between 0 and 1. Equation 8 is applied iteratively until convergence, that is, until the scores of the terms do not change too much or until a fixed number of iterations is reached. Equation 2 is used to weight the graph edges since high co-occurrence imply in high influence between the connected vertices.

3.3 Sorting Scores and Extracting Keywords

After generating a score for each term from a document, the scores must be sorted in descending order for keyword extraction. Any sorting algorithm can be used in this step [21]. The first k sorted terms correspond to the k most important terms in a document. Thus, these first k sorted terms are the keywords of a document.

3.4 General Overview of the Domain Independent Statistical Keyword Extraction Methods

Different sets of keywords are extracted by different methods since they have different assumptions about the properties of keywords. To illustrate this, we consider an example text composed by some paragraphs about data mining [22]. Table 2 presents the example text. We generate a sentence-term matrix and a graph based on this matrix. We only consider terms which occur in at least two sentences. Part of the sentence-term matrix is presented on Table 3 and the graph is illustrated on Figure 2.

We extract 10 keywords composed by single words considering the 5 different methods analyzed in this paper. The ranking of the extracted keywords is presented on Table 4. We notice that there are several common keywords for most methods. We can also notice that some important words for Data Mining domain as "*kdd*", "*machin*" and "*learn*" do not appear for all methods. On the other hand, important words as "*data*", "*mine*", "*analysi*", "*process*", "*database*" and "*pattern*" appear for almost all the methods.

Table 2: Example text to illustrate the differences among keyword extraction methods.

Data mining (the analysis step of the "Knowledge Discovery in Databases" process, or KDD), an interdisciplinary subfield of computer science, is the computational process of discovering patterns in large data sets involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems. The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. Aside from the raw analysis step, it involves database and data management aspects, data pre-processing, model and inference considerations, interestingness metrics, complexity considerations, post-processing of discovered structures, visualization, and on-line updating.
The actual data mining task is the automatic or semi-automatic analysis of large quantities of data to extract previously unknown interesting patterns such as groups of data records (cluster analysis), unusual records (anomaly detection) and dependencies (association rule mining). This usually involves using database techniques such as spatial indices. These patterns can then be seen as a kind of summary of the input data, and may be used in further analysis or, for example, in machine learning and predictive analytics. For example, the data mining step might identify multiple groups in the data, which can then be used to obtain more accurate prediction results by a decision support system. Neither the data collection, data preparation, nor result interpretation and reporting are part of the data mining step, but do belong to the overall KDD process as additional steps.
Data mining uses information from past data to analyze the outcome of a particular problem or situation that may arise. Data mining works to analyze data stored in data warehouses that are used to store that data that is being analyzed. That particular data may come from all parts of business, from the production to the management. Managers also use data mining to decide upon marketing strategies for their product. They can use data to compare and contrast among competitors. Data mining interprets its data into real time analysis that can be used to increase sales, promote new product, or delete product that is not value-added to the company.

Table 3: Sentence-term matrix extracted from example text.

data	mine	analysi	step	databas	process	kdd	comput	dicov	pattern	larg	set	involv	machin	learn	...
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	...
1	1	0	0	0	1	0	0	0	0	0	1	0	0	0	...
1	0	1	1	1	1	0	0	1	0	0	0	1	0	0	...
1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	...
1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	...
1	0	1	0	0	0	0	0	0	1	0	0	0	1	1	...
1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	...
1	1	0	1	0	1	1	0	0	0	0	0	0	0	0	...
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	...
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	...
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	...
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	...

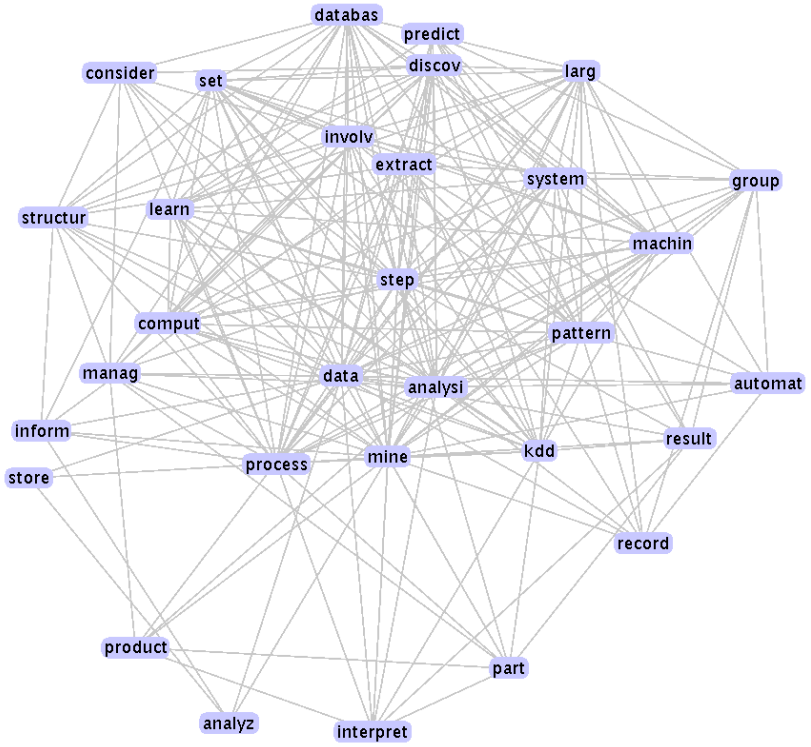


Figure 2: Graph extracted from Table 3.

Table 4: Keywords extracted by different methods.

Rank	MF	TF-ISF	CSI	Excentricity	TextRank
1^{st}	data	data	extract	data	data
2^{nd}	mine	mine	result	analysi	analysi
3^{rd}	analysi	analysi	group	step	process
4^{th}	process	step	analyz	process	step
5^{th}	step	databas	product	databas	databas
6^{th}	databas	process	part	pattern	involv
7^{th}	product	kdd	automat	involv	patter
8^{th}	pattern	comput	record	manage	discov
9^{th}	involv	discover	inform	productct	machine
10^{th}	manag	pattern	interpret	kdd	learn

We notice that the most frequent terms appear in top-ranked terms for MF and TF-ISF methods. There is a difference between these two methods only for less frequent terms. For instance, "*kdd*" and "*comput*" have frequency lower than "*pattern*" and "*involv*". However, "*patter*" and "*involv*" have their score decreased by TF-ISF method since they occur in more sentences than "*kdd*" and "*comput*".

CSI presents the most different set of keywords in comparison with other methods. CSI method is based on co-occurrence. Thus, even that a term t_i has a high frequency, the maximum number of co-occurrence is limited to the frequency of a co-ocurring

term t_j. When a term t_i has high co-occurrence with a term t_j, the second term of Equation 6 will have a high value and thus will decrease the score of a term t_i. Thus, terms with high frequency which co-occur with other frequent terms will have low scores. For this reason, terms such as "*data*" and "*mine*" do not appear in the top-ranked terms by CSI method.

The first terms ranked by Eccentricity and TextRank are practically the same. However, terms as "*manag*", "*product*", and "*kdd*" do not appear in TextRank since they are connected to terms with few connections and possibly not important terms. For instance, "*manage*" are connected with "*consider*", "*product*" and "*structur*", which have fewer connections than other vertices and thus receive/propagate a lower score than other vertices. Thus, terms as "*manage*" have their score decreased by TextRank method. On the other hand, the term "*manage*" is connected to term "*data*", wich is connected with all network vertex. Thus, "*manage*" reach other vertices of the network through term "*data*" with few steps, and then has their score increased by Eccentricy method.

In the next section we present what type of text structuring, method and number of keywords are advisable for incremental clustering considering collections with long texts and collections with short texts.

4 Experimental Evaluation

In this section we evaluate the impact in the incremental clustering quality provided by keywords extracted considering (i) different ways to structure texts for keyword extraction, (ii) different keyword extraction methods, and (iii) different number of keywords. The description of the textual document collections, the experiment configuration, evaluation criteria, results and discussion are presented in the next subsections.

4.1 Document Collections

We used 16 textual collections to analyze the effectiveness of the statistical keyword extraction methods. These textual collections are organized into two domains: (i) 8 textual collections composed by scientific articles and (ii) 8 textual collections composed by online news. The textual collections formed by scientific articles were extracted from proceedings of the ACM Digital Library[3]. These collections are composed by **long text documents**, which have a large number of segments. Each one of the ACM collections has 5 classes. Table 5 presents the number of classes, number of documents per class, and the total number of documents for each ACM collection. The number of documents in these collections ranges from 394 to 495 and the number of documents per class ranges from 50 to 105.

On the other hand, the textual collections formed by online news are composed predominantly by **short text documents** extracted from Estadão[4] online news, allowing to analyze the behavior of the keyword extraction methods considering texts with fewer segments than long text documents. Each one of the Estadão collections has 5 classes. Table 6 presents the number of classes, number of documents per class, and the total number of documents for each Estadão collection. The number of documents in these collections ranges from 394 to 495 and the number of documents per class ranges from 50 to 105.

Table 5: Description of the ACM textual document collections (long text documents) used in the experiments.

Collec.	Class	# Docs.	Total	Collec.	Class	# Docs.	Total
ACM-1	3D Technologies	91	401	ACM-5	Tangible and Embedded Interaction	81	471
	Visualization	72			Management of Data	96	
	Wireless Mobile Multimedia	82			User Interface Software and Technology	104	
	Solid and Physical Modeling	74			Information Technology Education	87	
	Software Engineering	82			Theory of Computing	103	
ACM-2	Rationality And Knowledge	86	411	ACM-6	Computational Geometry	89	439
	Simulation	84			Access Control Models and Technologies	90	
	Software Reusability	72			Computational Molecular Biology	71	
	Virtual Reality	83			Parallel Programming	96	
	Web Intelligence	86			Integrated Circuits and System Design	93	
ACM-3	Computer Architecture Education	78	424	ACM-7	Database Systems	104	471
	Networking And Communications Systems	75			Declarative Programming	101	
	Privacy in the Electronic Society	98			Parallel and Distributed Simulation	98	
	Software and Performance	81			Mobile Systems, Applications and Services	95	
	Web Information and Data Management	92			Network and System Support for Games	73	
ACM-4	Embedded Networked Sensor Systems	50	394	ACM-8	Mobile Ad Hoc Networking & Computing	90	495
	Information Retrieval	71			Knowledge Discovery and Data Mining	105	
	Parallel Algorithms and Architectures	98			Embedded Systems	102	
	Volume Visualization	104			Hypertext and Hypermedia	93	
	Web Accessibility	71			Microarchitecture	105	

4.2 Experiment Configuration and Evaluation Criteria

The segment term matrices were generated considering sentences or a sliding window of words as text segments. These are the two common ways to structure a text into a segment-term matrix [8–11]. As sentences we consider the set of words separated by a stop mark (".", "?" or "!"). We consider 2, 3, 4, 5, 10, 15, 20 and 30 as the size of sliding windows. The terms are single words stemming by Porter's algorithm [23]. We removed from the segment-term matrix the stopwords contained in the stoplist of

[3] http://dl.acm.org/
[4] http://www.estadao.com.br/

Table 6: Description of the Estadão textual document collections (short text documents) used in the experiments.

Collec.	Class	# Docs.	Total	Collec.	Class	# Docs.	Total
Estadão-1	Art & Entertainment-A	100	500	Estadão-5	Opinion	43	337
	Brazil	100			Middle East	34	
	Cities	100			Plannet	60	
	Economy-A	100			National-B	100	
	Sports-A	100			Technology	100	
Estadão-2	Science	100	442	Estadão-6	Sports-D	100	500
	Movie	42			Politics-A	100	
	Culture	100			São Paulo-A	100	
	Economy-B	100			Technology-A	100	
	General-A	100			Life-A	100	
Estadão-3	Economy-C	100	500	Estadão-7	São Paulo-B	100	500
	Sports-B	100			Health	100	
	General-B	100			Supplement	100	
	International-A	100			Economy-D	100	
	Culture	100			International-C	100	
Estadão-4	International-B	100	500	Estadão-8	Technology-B	100	500
	National-A	100			General-C	100	
	Business	100			National-C	100	
	Sports-C	100			Life-B	100	
	Art & Entertainment-B	100			Politics-B	100	

the SKET. For ACM collections we also removed terms which occur in only one text segment. This is a standard preprocessing step for keyword extractions [11] and reduces the computation time for keyword extraction in long text documents. On the other hand, we do not performed this step for Estadão collection since this removed most of the terms in the texts.

TextRank is the only method which requires the setting of parameter and stopping criteria. In this article we use $\lambda = 0.85$ [10, 20] and we consider as stopping criteria the maximum of 100 iterations or when the difference of the vertice scores in consecutive iterations is lesser than 0.00001.

The domain independent statistical keyword extraction methods were also compared to bag-of-words (BOW) approach, i.e., using all terms of the collection as keywords, and to TF-IDF (term frequency - inverse document frequency) ranking approach, which is a domain dependent keyword extraction method or feature selection method. TF-IDF approach select terms as keywords according to the highest TF-IDF scores, i.e.,

$$ScoreTFIDF(t_i) = \sum_{d_j \in \mathcal{D}} tfidf_{t_i,d_j}, \tag{9}$$

in which $tfidf_{t_i,d_j} = tf_{t_i,d_j} \times idf_{t_i}$, $idf_{t_i} = \log |\mathcal{D}|/df_{t_i}$, $|\mathcal{D}|$ is the number of documents in the collections, tf_{t_i,d_j} is the frequency of term t_i in document d_j, and df_{t_i} is the number of document in which term t_i occurs [24]. The number of keywords used in the TF-IDF method were based on the average number of terms generate for the different domain-dependent methods for each used number of keywords.

The minimum similarity threshold values m of the incremental clustering algorithm Leader-Follower was defined on the range $[0.05, 0.5]$ with 0.05 of increment. Since the results of the incremental clustering depend on the insertion order of the documents, the clustering process was repeated with different insertion orders. Moreover, the clustering process was repeated several times to attenuate random fluctuations of the evaluation results. The cosine measure was used to compute the similarities of the documents since this is a recommended measure for this type of data [25].

An adaptation of the $F\text{-}Score$ index [24] was used to evaluate the quality of the partitions generated by the incremental clustering algorithm. To compute the *F-Score* index, consider that:

- C is a partition;
- L_r represents a document set of a class r; and
- G_i represents a document set of a cluster i that belongs to the partition C.

Given a class L_r and a cluster G_i, Precision ($P(L_r, G_i)$) - Equation 10, Recall ($R(L_r, G_i)$) - Equation 11, and F ($F(L_r, G_i)$) - Equation 12, can be computed.

$$P(L_r, G_i) = \frac{|L_r \cap G_i|}{|G_i|} \tag{10}$$

$$R(L_r, G_i) = \frac{|L_r \cap G_i|}{|L_r|} \tag{11}$$

$$F(L_r, G_i) = \frac{2 * P(L_r, G_i) * R(L_r, G_i)}{P(L_r, G_i) + R(L_r, G_i)} \tag{12}$$

$F\text{-}Score$ selects for a certain class L_r the highest value obtained by some cluster of the partition C, as presented in Equation 13. Then, the *F-Score* of the partition is the sum of the F value for the classes of the collection (c) divided by the number of documents (n), as presented in Equation 14. The $F\text{-}Score$ value is 1 if the partition separates correctly the documents according to their class and 0 otherwise.

$$F(L_r) = \max_{G_i \in C} F(L_r, G_i) \tag{13}$$

$$F\text{-}Score = \sum_{r=1}^{c} \frac{|L_r|}{n} F(L_r) \tag{14}$$

4.3 Results

We consider four aspects to analyze the results obtained in the experimental evaluation: (i) the quality of the keywords extracted by each keyword extraction method; (ii) the effect of the number of keywords in the cluster quality; (iii) the effect of different ways to structure the text for keyword extraction; and (iv) the comparison with the use of all terms and domain-dependent keyword extraction methods, which are applied in **static scenarios** since they need to pre-process all the collection before the clustering task. Moreover, these aspects are discussed considering both long and short text documents, as presented in the following sections.

4.3.1 Statistical Keyword Extraction Methods for Long Text Documents

The collections composed by scientific articles (ACM-1, ACM-2, ACM-3, ACM-4, ACM-5, ACM-6, ACM-7, and ACM-8) allow fairly compare the effectiveness of all keyword extraction methods described in this paper, especially the methods based on sentences and sliding window[5]. Thus, we explore this scenario in detail to analyze and discuss the experimental results.

Due to the huge combinations of parameters, we adopt an approach based on statistical analysis to identify the configurations which presented significantly better results than the average [6]. We use the t-student test with 99% of confidence for this analysis. To illustrate the statistical analysis, we present on Figures 3(a) and 3(b) some results of the statistical tests carried out for each keyword extraction method considering the sentence and window approach to structure the texts. The cells with white color correspond to configurations with results below average. The cells with gray color correspond to configurations which obtained results above average. The cells with underlined values correspond to results above average and with statistically significant differences. With this approach we selected a subset of parameters with the best results for the analysis considering the four different aspects presented previously. We can notice that the use of more than 25 keyword usually provide results above average. The other textual collections showed a similar behavior with respect to parameters analysis.

Table 7 presents the average number of terms extracted by the different keyword extraction methods considering different number of keyword per document and the bag-of-words approach (All)[7]. Extracting 5 keywords per documents reduces about 98% the total number of terms and extracting 60 keywords per document reduces about 88%. The TF-IDF considering the number of features generated by 10 keywords obtained the best results for this approach.

Table 7: Average number of terms extracted by different keyword extraction methods considering different number of keywords per document.

# keywords	ACM-1	ACM-2	ACM-3	ACM-4	ACM-5	ACM-6	ACM-7	ACM-8
All	38827	46470	37287	51557	38574	51743	52540	47275
5	841	869	807	761	947	860	876	924
10	1476	1481	1398	1304	1555	1487	1519	1581
15	1969	1956	1861	1729	2016	1980	2022	2079
20	2374	2357	2248	2088	2412	2385	2438	2481
25	2713	2705	2586	2404	2741	2742	2786	2824
30	3020	3023	2863	2695	3032	3055	3093	3131
35	3130	3206	2954	2940	3452	3428	3363	3405
40	3374	3484	3179	3179	3711	3703	3627	3671
45	3616	3753	3387	3410	3942	3960	3882	3914
50	3836	4018	3589	3632	4173	4193	4113	4143
55	4046	4269	3783	3844	4386	4413	4337	4369
60	4260	4525	3977	4060	4585	4630	4566	4597

In Appendix A we show the graphics regarding the trade-off among *F-Score* and the different number the keywords extracted by each method. We can notice that the quality of the incremental clustering increases significantly when the number of keywords

[5]The representations of the ACM collections used in the experiments are available at `http://sites.labic.icmc.usp.br/sket/datasets/acm/`

[6]The results for ACM collection obtained by all the combinations of parameters are available at `http://sites.labic.icmc.usp.br/sket/results/f-scores_acm.pdf`

[7]All the number of terms extracted by the different keyword extraction methods and number of keyword for the ACM collections are available at `http://sites.labic.icmc.usp.br/sket/results/number_terms_acm.pdf`

	Datasets							
#Keywords	ACM-1	ACM-2	ACM-3	ACM-4	ACM-5	ACM-6	ACM-7	ACM-8
5	0.3492	0.2287	0.2619	0.3998	0.3105	0.3698	0.3142	0.2888
10	0.3239	0.3010	0.3472	0.6680	0.3957	0.4611	0.3548	0.3265
15	0.3249	0.3515	0.3962	0.7257	0.5006	0.4620	0.4004	0.3372
20	0.3230	0.3500	0.5339	0.8102	0.5252	0.5523	0.4872	0.3996
25	0.3561	0.5402	0.5782	0.7926	0.5148	0.5402	0.5153	0.4677
30	0.4154	0.6301	0.5971	0.8224	0.5709	0.6024	0.5353	0.4373
35	0.5736	0.5828	0.7251	0.8137	0.5447	0.5615	0.6227	0.4898
40	0.5574	0.5838	0.7435	0.8309	0.5302	0.6254	0.6002	0.5121
45	0.5700	0.6369	0.7166	0.8326	0.5660	0.6058	0.6082	0.5057
50	0.5592	0.6596	0.7328	0.8587	0.5621	0.6257	0.7041	0.5002
55	0.5999	0.6463	0.7614	0.8863	0.6009	0.5978	0.7118	0.5564
60	0.6246	0.6444	0.7334	0.8770	0.6125	0.6391	0.6641	0.5428

Method=CSI Window Size=2

	Datasets							
#Keywords	ACM-1	ACM-2	ACM-3	ACM-4	ACM-5	ACM-6	ACM-7	ACM-8
5	0.2427	0.2579	0.2228	0.2384	0.2777	0.2621	0.2711	0.2452
10	0.2452	0.2583	0.2234	0.3037	0.2850	0.2622	0.2540	0.2765
15	0.2338	0.2890	0.2588	0.3162	0.2755	0.2940	0.2767	0.3000
20	0.2561	0.2769	0.2556	0.3153	0.2678	0.2972	0.3048	0.3066
25	0.2783	0.3130	0.2811	0.3504	0.2745	0.2915	0.3061	0.3289
30	0.3091	0.3717	0.2854	0.3755	0.2819	0.3639	0.3317	0.3437
35	0.3466	0.4328	0.3280	0.3549	0.2863	0.3186	0.4199	0.3467
40	0.3965	0.4490	0.3842	0.4057	0.3018	0.3894	0.4521	0.3386
45	0.4230	0.5653	0.4020	0.4499	0.3047	0.3495	0.4617	0.3205
50	0.4261	0.4650	0.4026	0.5759	0.3218	0.3988	0.4651	0.3830
55	0.4679	0.4929	0.4703	0.5513	0.4268	0.4445	0.5196	0.3734
60	0.4703	0.4723	0.4662	0.5869	0.5096	0.5221	0.5125	0.4261

Method=CSI Sentence

	Datasets							
#Keywords	ACM-1	ACM-2	ACM-3	ACM-4	ACM-5	ACM-6	ACM-7	ACM-8
5	0.2726	0.3717	0.4008	0.6221	0.3813	0.3323	0.3650	0.3544
10	0.4562	0.4672	0.5113	0.7319	0.5533	0.6042	0.4940	0.3976
15	0.4401	0.5510	0.5493	0.8083	0.6224	0.6035	0.6441	0.4821
20	0.4930	0.5760	0.6654	0.7804	0.6160	0.6678	0.7025	0.4646
25	0.5427	0.5807	0.6535	0.7974	0.6487	0.6590	0.6697	0.4342
30	0.5225	0.6309	0.6877	0.8776	0.6496	0.6010	0.6984	0.4539
35	0.6106	0.6443	0.6676	0.8735	0.6127	0.6603	0.6362	0.5128
40	0.5730	0.6573	0.6648	0.8790	0.6035	0.6760	0.6474	0.5230
45	0.5900	0.6763	0.6509	0.8826	0.6241	0.6589	0.6624	0.5296
50	0.6085	0.7140	0.6612	0.8638	0.6191	0.6450	0.6612	0.5239
55	0.6234	0.6816	0.6773	0.8473	0.6504	0.6692	0.6655	0.5282
60	0.6289	0.7488	0.7109	0.8595	0.6803	0.7299	0.6941	0.5414

Method=ECC Window Size=30

	Datasets							
#Keywords	ACM-1	ACM-2	ACM-3	ACM-4	ACM-5	ACM-6	ACM-7	ACM-8
5	0.2611	0.2815	0.3007	0.4026	0.3625	0.2790	0.3778	0.2613
10	0.3026	0.3707	0.5601	0.5878	0.4050	0.3846	0.4834	0.3359
15	0.3839	0.4110	0.5052	0.5919	0.5307	0.5234	0.5953	0.4531
20	0.4589	0.5290	0.5935	0.6945	0.6469	0.6247	0.6531	0.4793
25	0.4690	0.5779	0.6118	0.7298	0.6017	0.7308	0.6912	0.4614
30	0.5315	0.5794	0.6605	0.7546	0.7060	0.7907	0.7679	0.4724
35	0.5392	0.6774	0.6530	0.7252	0.6576	0.7896	0.8186	0.5416
40	0.5453	0.6887	0.6898	0.7308	0.6388	0.7896	0.8432	0.4800
45	0.5374	0.7686	0.8110	0.7403	0.6497	0.7856	0.7871	0.5008
50	0.5330	0.7424	0.7892	0.7812	0.7287	0.8119	0.7997	0.5921
55	0.5237	0.7384	0.7833	0.7683	0.7686	0.8224	0.8302	0.6407
60	0.5583	0.7454	0.8223	0.7960	0.7668	0.7732	0.8305	0.6005

Method=ECC Sentence

	Datasets							
#Keywords	ACM-1	ACM-2	ACM-3	ACM-4	ACM-5	ACM-6	ACM-7	ACM-8
5	0.3196	0.4461	0.5783	0.6561	0.4495	0.4696	0.5731	0.3551
10	0.4771	0.4967	0.6407	0.6550	0.5770	0.6160	0.6993	0.4340
15	0.5310	0.6157	0.7512	0.7054	0.6068	0.7532	0.8023	0.4672
20	0.5229	0.6996	0.6955	0.7533	0.6351	0.7604	0.8463	0.5109
25	0.5603	0.6769	0.6480	0.7842	0.7019	0.7834	0.8823	0.5029
30	0.5316	0.7109	0.6859	0.7967	0.6489	0.7726	0.8878	0.5368
35	0.5259	0.6861	0.7319	0.8009	0.6726	0.6667	0.8443	0.5529
40	0.5691	0.6814	0.7205	0.7770	0.6512	0.6284	0.8701	0.5730
45	0.5755	0.6990	0.8261	0.8056	0.6479	0.6724	0.8777	0.5398
50	0.5797	0.7308	0.8457	0.8009	0.6235	0.6899	0.8831	0.6095
55	0.5847	0.7932	0.7320	0.8211	0.6425	0.7100	0.8831	0.6336
60	0.5514	0.7481	0.7317	0.7919	0.6694	0.6875	0.8854	0.6399

Method=MF Window Size=20

	Datasets							
#Keywords	ACM-1	ACM-2	ACM-3	ACM-4	ACM-5	ACM-6	ACM-7	ACM-8
5	0.3399	0.4457	0.5783	0.6561	0.4492	0.4668	0.5731	0.3552
10	0.4781	0.4959	0.6415	0.6550	0.5769	0.6150	0.6993	0.4340
15	0.5272	0.6140	0.7395	0.7054	0.6068	0.7532	0.8023	0.4674
20	0.5464	0.6995	0.7033	0.7511	0.6351	0.7603	0.8464	0.5113
25	0.5586	0.6737	0.6480	0.7842	0.7017	0.7711	0.8823	0.5029
30	0.5695	0.7131	0.6851	0.7968	0.6505	0.7726	0.8743	0.5374
35	0.5259	0.6883	0.7319	0.7947	0.7031	0.7521	0.8442	0.5570
40	0.5681	0.6812	0.7205	0.7763	0.7166	0.6602	0.8701	0.5730
45	0.5655	0.7007	0.8261	0.8056	0.6854	0.7606	0.8777	0.5398
50	0.5796	0.7162	0.8457	0.8078	0.6693	0.8017	0.8833	0.6085
55	0.5831	0.7742	0.7320	0.8127	0.6989	0.8024	0.8831	0.6340
60	0.5509	0.7567	0.7316	0.7919	0.7101	0.7308	0.8851	0.6378

Method=MF Sentence

	Datasets							
#Keywords	ACM-1	ACM-2	ACM-3	ACM-4	ACM-5	ACM-6	ACM-7	ACM-8
5	0.3113	0.4374	0.5277	0.6399	0.4134	0.4948	0.4120	0.4001
10	0.4159	0.5496	0.6017	0.7572	0.6011	0.6883	0.6176	0.4827
15	0.4827	0.6229	0.6844	0.7827	0.6158	0.7191	0.6900	0.4641
20	0.5230	0.6696	0.7645	0.8091	0.6339	0.6656	0.6840	0.4632
25	0.4893	0.6257	0.7198	0.7688	0.5877	0.6714	0.7002	0.4815
30	0.5083	0.6970	0.6919	0.8580	0.6182	0.6766	0.6587	0.5237
35	0.6011	0.7227	0.7040	0.8909	0.6422	0.6784	0.7098	0.5292
40	0.6073	0.6910	0.6839	0.8294	0.6487	0.6654	0.7112	0.5329
45	0.6053	0.6983	0.6801	0.8654	0.6402	0.6278	0.7144	0.5300
50	0.5628	0.6489	0.6726	0.8781	0.6432	0.6335	0.6432	0.5432
55	0.5922	0.6968	0.6753	0.8700	0.6476	0.6808	0.6510	0.5346
60	0.5977	0.7422	0.7262	0.8731	0.6513	0.7027	0.6219	0.5420

Method=TR Window Size=30

	Datasets							
#Keywords	ACM-1	ACM-2	ACM-3	ACM-4	ACM-5	ACM-6	ACM-7	ACM-8
5	0.3170	0.4224	0.4645	0.4388	0.4797	0.4246	0.4802	0.3672
10	0.4178	0.4849	0.6236	0.5912	0.6659	0.5699	0.5964	0.4233
15	0.4574	0.5820	0.6347	0.6582	0.5716	0.6906	0.7587	0.4550
20	0.5080	0.6088	0.6631	0.6685	0.6919	0.7072	0.7734	0.4661
25	0.5465	0.5856	0.6570	0.7134	0.7133	0.7487	0.8435	0.4974
30	0.5650	0.7070	0.6563	0.7847	0.6533	0.8042	0.8360	0.5293
35	0.5646	0.7265	0.7149	0.7473	0.6773	0.7970	0.8655	0.5848
40	0.5081	0.6897	0.6765	0.8009	0.7259	0.7866	0.8631	0.5617
45	0.5697	0.6897	0.7246	0.7690	0.7677	0.7775	0.8439	0.5666
50	0.5972	0.6912	0.7703	0.7771	0.7688	0.8383	0.8591	0.5715
55	0.5711	0.6718	0.7537	0.7702	0.7474	0.7282	0.8786	0.5935
60	0.5814	0.6875	0.7557	0.7800	0.7585	0.7379	0.8776	0.6082

Method=TR Sentence

	Datasets							
#Keywords	ACM-1	ACM-2	ACM-3	ACM-4	ACM-5	ACM-6	ACM-7	ACM-8
5	0.2797	0.3752	0.4364	0.4468	0.2768	0.2879	0.3714	0.2142
10	0.3527	0.3981	0.5164	0.5531	0.3148	0.3425	0.4931	0.2920
15	0.3820	0.3889	0.5011	0.5787	0.3311	0.3805	0.5280	0.3378
20	0.4503	0.5176	0.5237	0.7077	0.3971	0.4301	0.6757	0.3532
25	0.4595	0.5493	0.4996	0.7399	0.4879	0.5669	0.7138	0.4547
30	0.4619	0.5907	0.6250	0.6781	0.5916	0.6227	0.7531	0.4804
35	0.4781	0.6153	0.6292	0.6844	0.5432	0.6458	0.7590	0.5085
40	0.4182	0.7255	0.5038	0.7644	0.6012	0.6582	0.7711	0.5116
45	0.4605	0.6803	0.7104	0.7544	0.6146	0.6357	0.8055	0.5209
50	0.5252	0.6030	0.6614	0.7949	0.5950	0.6185	0.7415	0.5172
55	0.4394	0.7309	0.6800	0.7989	0.6079	0.6456	0.7432	0.5400
60	0.4520	0.7169	0.7612	0.8007	0.6279	0.6445	0.8222	0.5462

Method=TFISF Window Size=20

	Datasets							
#Keywords	ACM-1	ACM-2	ACM-3	ACM-4	ACM-5	ACM-6	ACM-7	ACM-8
5	0.2526	0.3752	0.4364	0.4468	0.2768	0.2881	0.3714	0.2125
10	0.3686	0.3981	0.5210	0.5531	0.3148	0.3434	0.4926	0.2921
15	0.3919	0.3870	0.4984	0.5787	0.3311	0.3803	0.5280	0.3376
20	0.4907	0.5176	0.5239	0.7077	0.3971	0.4302	0.6770	0.3525
25	0.4676	0.5519	0.5000	0.7399	0.4836	0.5766	0.7138	0.4549
30	0.5098	0.6029	0.6254	0.6781	0.6004	0.6227	0.7531	0.4789
35	0.4781	0.6153	0.6195	0.6844	0.5930	0.7852	0.7620	0.5050
40	0.4183	0.7255	0.5037	0.7644	0.6890	0.7236	0.7711	0.5116
45	0.4560	0.6805	0.7104	0.7544	0.7113	0.7732	0.8055	0.5255
50	0.5249	0.6108	0.6614	0.7949	0.6248	0.6704	0.7397	0.5172
55	0.4412	0.7309	0.6840	0.7927	0.6492	0.7342	0.7458	0.5381
60	0.4554	0.7169	0.7648	0.7975	0.7010	0.7078	0.8222	0.5456

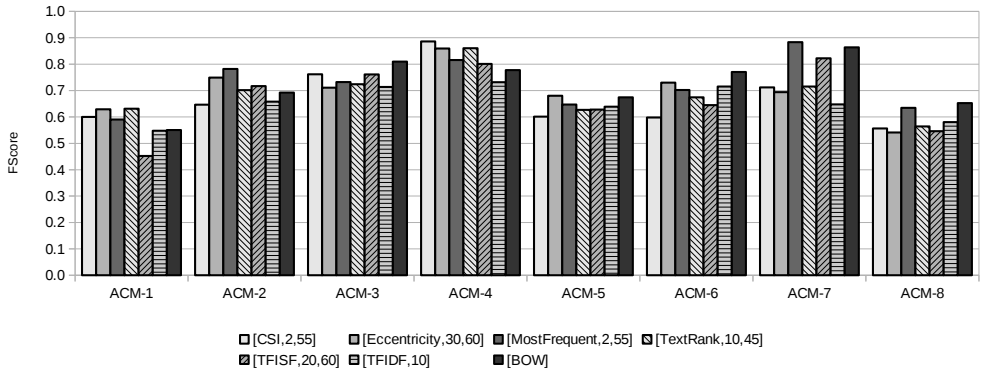
Method=TFISF Sentence

(a) (b)

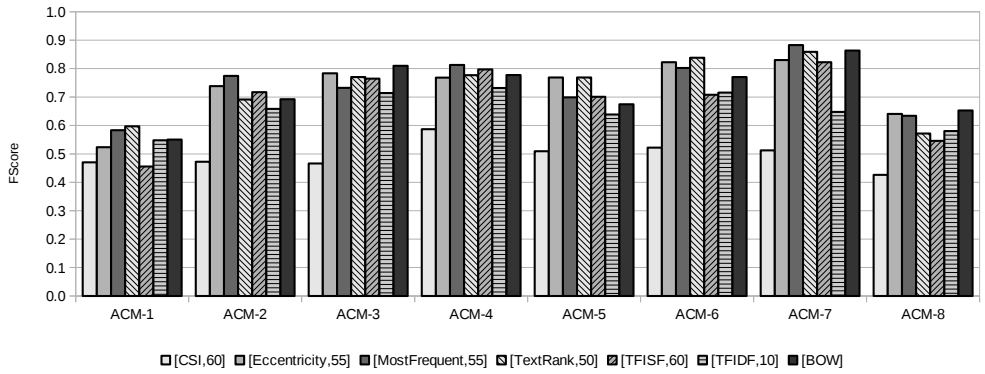
Figure 3: Statistical analysis of parameter settings for keyword extraction methods using (a) sliding window and (b) sentences. The cells with white color correspond to configurations with results below average. The cells with gray color correspond to configurations which obtained results above average. The cells with underlined values correspond to results above average and with statistically significant differences.

is between 20 and 35. There is a small decrease in the clustering quality when the number of keywords is between 5 and 15. On the other hand, a higher number of keywords, such as 40 or 60, does not provide a significant improvement in the clustering quality.

The best configurations for each method are presented on Figure 4. The first graphic (Figure 4(a)) shows the best results considering the windows approach to generate the segment-term matrix, and the second graphic (Figure 4(b)) shows the sentence approach. We use the notation [Method,#Keywords] to show the configurations using sentence approach and [Method,Window-Window Size,#Keywords] to show the configurations using windows approach. We can notice that the domain independent keyword extraction methods surpass the *f-scores* of BOW and TF-IDF for most of the ACM collection for both sentence and window mapping. The exception is CSI method using sentence mapping, which obtains lower *f-scores* than BOW and TF-IDF for all ACM collections.



(a) Window



(b) Sentence

Figure 4: Comparison of quality of incremental clustering considering the best configuration of each extraction method keywords by using (a) windows and (b) sentences.

We perform a statistical analysis using the Friedman statistical test with Nemenyi post-hoc test and 95% of confidence to verify if the domain-independent extraction methods in dynamic scenarios are able to provide equivalent results to domain-dependent methods in static scenarios. Figure 5 presents a graphical illustration of the statistical test. In this illustration the configurations are sorted according to the average ranking of the *F-Score* and the methods connected by a line do not present statistically significant differences among them. We can observe that there are no statistically significant differences among the domain-independent and the domain-dependent keyword extraction methods or even the BOW approach. However, some domain-independent methods obtained a better rank position than BOW but with a significantly reduction in the number of terms ([MF,Sentence,55],[MF,Windows-2,55],[ECC,Sentence,55] e [TR,Sentence,50]). We can notice that the sentence mapping provides better results than sliding window mapping in long text document since the same method using the sentence mapping obtained a better ranking than using window mapping. The exception is the CSI method which obtained better results with sliding window mapping.

The analyses presented in this section show that the domain-independent methods investigated in this article are potentially useful in dynamic scenarios with long text documents since they (i) keep the clustering quality, (ii) do not need to analyze the entire collection to extract keywords and (iii) reduces considerably the total number of terms in the text collections.

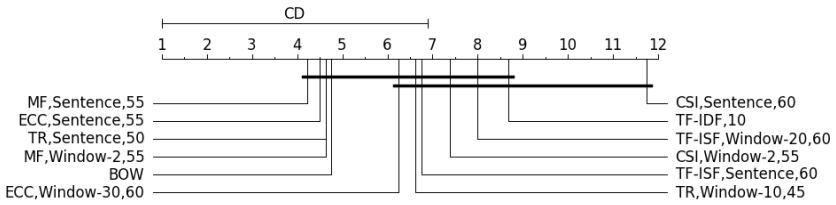


Figure 5: Statistical comparison between the best configurations of the keyword extraction methods (domain independent), TF-IDF (domain dependent), and BOW.

4.3.2 Statistical Keyword Extraction Methods for Short Text Documents

To analyze the behavior of the keyword extraction methods in this scenario, i.e., texts with fewer segments than long texts documents, we used 8 textual collections composed by online news from Estadão online news[8]. Table 8 presents the average number of terms extracted by the different keyword extraction methods considering different number of keyword per document and considering all features, i.e., without keyword extraction or feature selection[9]. We notice a reduction about 98% in the number of features using 5 keywords per document until 86% using 60 keyword per document.

Table 8: Average number of terms extracted by different keyword extraction methods considering different number of keywords per document.

# keywords	Estadão-1	Estadão-2	Estadão-3	Estadão-4	Estadão-5	Estadão-6	Estadão-7	Estadão-8
All	33909	28297	26212	28694	24146	28490	29894	24508
5	984	998	1008	1041	776	1026	1080	998
10	1587	1643	1645	1686	1286	1668	1721	1603
15	2077	2153	2159	2206	1683	2163	2244	2087
20	2503	2596	2589	2637	2032	2589	2679	2495
25	2877	2995	2975	3022	2344	2965	3076	2860
30	3213	3368	3337	3364	2635	3319	3435	3191
35	3531	3701	3669	3689	2907	3645	3768	3504
40	3833	4026	3984	4011	3174	3947	4078	3803
45	4120	4326	4274	4312	3424	4237	4381	4077
50	4398	4611	4545	4594	3652	4509	4657	4341
55	4654	4881	4812	4856	3866	4766	4928	4592
60	4904	5144	5064	5113	4071	5007	5185	4844

In Appendix B we show the graphics regarding the trade-off among *F-Score* and the different number the keywords extracted by each method. We can notice that the quality of incremental clustering increases significantly when the number of keywords is among 15 and 30. In few cases the clustering quality increases significantly for more than 30 keywords. A number of keywords fewer than 15 decreases the clustering quality for most of the collections.

The best configurations for each method are presented on Figure 6. The first graphic (Figure 6(a)) shows the best results considering the window mapping to generate the segment-term matrix, and the second graphic (Figure 6(b)) shows the sentence mapping. We can notice that all statistical keyword extraction methods were better than BOW using the window mapping for most of the collections. When using sentence mapping only CSI method obtained worst result than BOW. Most Frequent, TextRank, and Eccentricity methods provided the best clustering quality considering the keyword extraction methods.

In Figure 7 we present a graphical illustration of the Friedman statistical test with Nemenyi post-hoc text with 95% of confidence level considering the results presented on Figure 6. We can notice that the Most Frequent method and feature selection using TF-IDF presented better results with statistically significant differences than BOW. Furthermore, the window mapping provided better results than sentence mapping since all methods considering the window mapping obtained a better ranking than the use of sentence mapping.

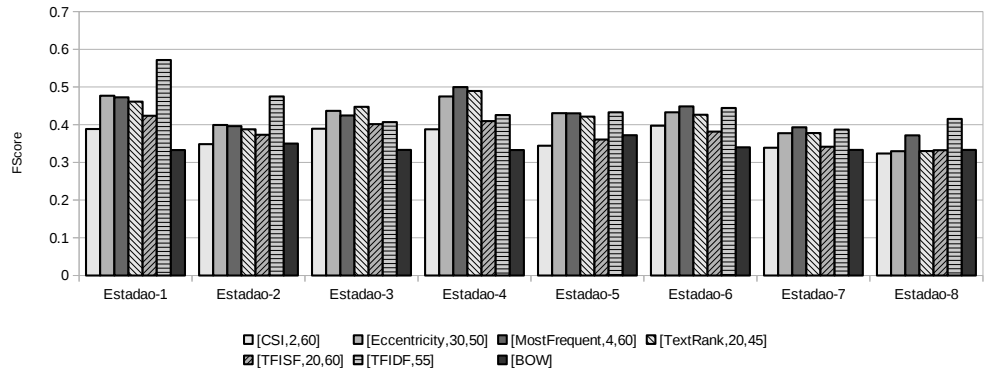
The analysis presented in this sections shows that the investigated domain independent keyword extraction methods are also useful in dynamic scenarios with short text documents since (i) most of them obtained a better clustering quality than BOW, (ii) obtained a clustering quality similar to TF-IDF but without analysing the entire collection, and (iii) reduced considerably the total number of terms in the collections with short text documents.

5 Conclusions and Future Work

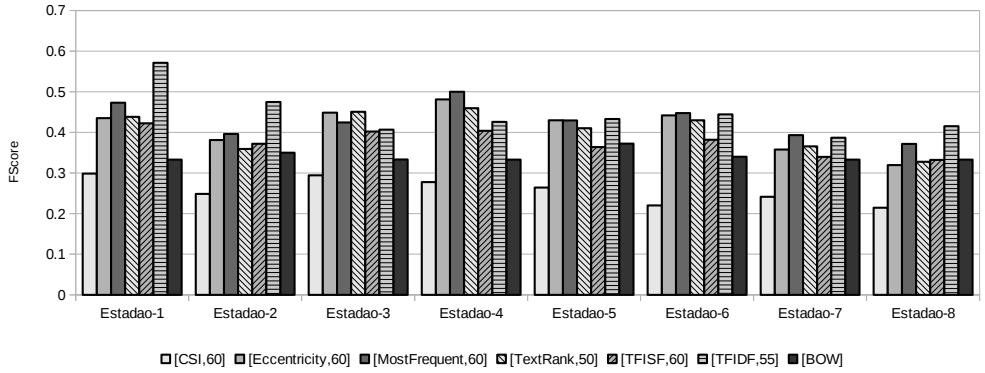
In this article we analyze the impact of the different statistical keyword extraction methods applied in different ways to structure texts and the different number of keywords extracted per document in the incremental clustering task. In general, the

[8]The representations of the Estadao collections used in the experiments are available at `http://sites.labic.icmc.usp.br/sket/datasets/estadao/`

[9]All the number of terms extracted by the different keyword extraction methods and number of keyword for the Estadão collections are available at `http://sites.labic.icmc.usp.br/sket/results/number_terms_estadao.pdf`



(a) Window



(b) Sentence

Figure 6: Comparison of quality of incremental clustering considering the best configuration of each extraction method keywords by using (a) windows and (b) sentences.

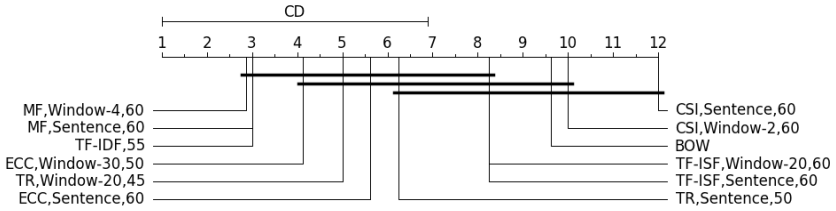


Figure 7: Statistical comparison between the best configurations of the keyword extraction methods (domain independent), TF-IDF (domain dependent), and BOW.

keyword extraction methods improved/maintained the quality of incremental clustering for collections with short or long text documents. Usually 35 keywords are enough to maintain the clustering quality for collections with long text documents and 15 keywords are enough for collections with short text documents. However, we observed that the highest number of keywords used in this article ([45-60]) obtained the best clustering quality.

The statistical keyword extraction methods presented equivalent results, except the method Co-occurrence Statistical Information, which presented inferior results in most cases mainly using sentence mapping. In general, the use of Most Frequent, Eccentricity and TextRank methods presented the best results. The sentence mapping to structure the texts for keyword extraction obtained the best results for collections with long text documents, and the windows mapping obtained the best results for collections with short text documents. Moreover, the domain independent keyword extraction methods obtained equivalent clustering quality but required fewer resources than the use of all terms or traditional feature selection methods.

As future works we intend to build a term graph using different correlation measures and apply other centrality measures to extract keywords. We also intend to apply the domain independent keyword extraction methods presented here in collections of other domains. An ensemble of the statistical keyword extraction methods will be also evaluated.

Acknowledgements

Grants 2010/20564-8, 2011/12823-6, 2011/19850-9 and 2014/08996-0, Sao Paulo Research Foundation (FAPESP).

REFERENCES

[1] C. C. Aggarwal and C. Zhai. “A Survey of Text Clustering Algorithms”. In *Mining Text Data*, pp. 77–128. Springer-Verlag, 2012.

[2] A. K. Jain. “Data clustering: 50 years beyond K-means”. *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[3] R. Xu and D. Wunsch. *Clustering*. Wiley-IEEE Press, IEEE Press Series on Computational Intelligence, 2008.

[4] J. Liu and J. Wang. “Keyword Extraction Using Language Network”. In *Natural Language Processing and Knowledge Engineering*, pp. 129–134. IEEE Computer Society, 2007.

[5] G. Salton and C. Buckley. “Term-weighting approaches in automatic text retrieval”. *Information Processing and Management*, vol. 24, no. 5, pp. 513–523, 1988.

[6] K. W. Church and P. Hanks. “Word association norms, mutual information, and lexicography”. *Computational Linguistics*, vol. 16, pp. 22–29, 1990.

[7] T. Dunning. “Accurate Methods for the Statistics of Surprise and Coincidence”. *Computational Linguistics*, vol. 19, pp. 61–74, 1993.

[8] A. P. E. C. B. Martins, T. A. S. Pardo and L. H. M. Rino. “Introdução à Sumarização Automática”. Technical Report RT-DC 002/2001, ICMC-USP, 2001. 38p.

[9] Y. Matsuo and M. Ishizuka. “Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information”. In *Florida Artificial Intelligence Research Society Conference*, pp. 392–396, 2003.

[10] R. Mihalcea and P. Tarau. “TextRank: Bringing Order into Texts”. In *Conference on Empirical Methods in Natural Language Processing*, pp. 404–411, 2004.

[11] G. K. Palshikar. “Keyword Extraction from a Single Document Using Centrality Measures”. In *Pattern Recognition and Machine Intelligence*, pp. 503–510, 2007.

[12] O. Maimon and L. Rokach. *Data Mining and Knowledge Discovery Handbook*. Springer-Verlag, 2005.

[13] C. Giraud. “A note on the utility of incremental learning”. *AI Communications*, vol. 13, pp. 215–223, 2000.

[14] A. K. Jain, M. N. Murty and P. J. Flynn. “Data clustering: a review”. *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[15] R. G. Rossi and S. O. Rezende. “Building a topic hierarchy using the bag-of-related-words representation”. In *Symposium on Document Engineering*, pp. 195–204, 2011.

[16] R. Feldman and J. Sanger. *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press, 2006.

[17] P. Greenwood and M. Nikulin. *A Guide to Chi-Squared Testing*. A Wiley-Interscience publication. Wiley, 1996.

[18] M. Newman. *Networks: An Introduction*. Oxford University Press, Inc., 2010.

[19] Y. Matsuo, Y. Ohsawa and M. Ishizuka. "KeyWorld: Extracting Keywords from a Document as a Small World". In *International Conference on Discovery Science*, pp. 271–281, 2001.

[20] S. Brin and L. Page. "The Anatomy of a Large-Scale Hypertextual Web Search Engine". *Computer Networks*, vol. 30, pp. 107–117, 1998.

[21] T. H. Cormen, C. Stein, R. L. Rivest and C. E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, second edition, 2001.

[22] Wikipedia. "Data Mining", 2013. `http://en.wikipedia.org/wiki/Data_mining`. Last modified on 6 July 2013.

[23] M. F. Porter. "An algorithm for suffix stripping". *Readings in Information Retrieval*, vol. 14, no. 3, pp. 130–137, 1980.

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

[25] P.-N. Tan, M. Steinbach and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

A Analysis of the Number of Keywords in the Quality of Incremental Clustering for Long Text Documents

Figures 8 – 15 present the trade-off among *F-Score* and the different number the keywords extracted by each method. The presented *F-Score* is the average value the of *F-Score* from the 30 best configurations from each method. The graphic shows a line which represents the use of all features (BOW).

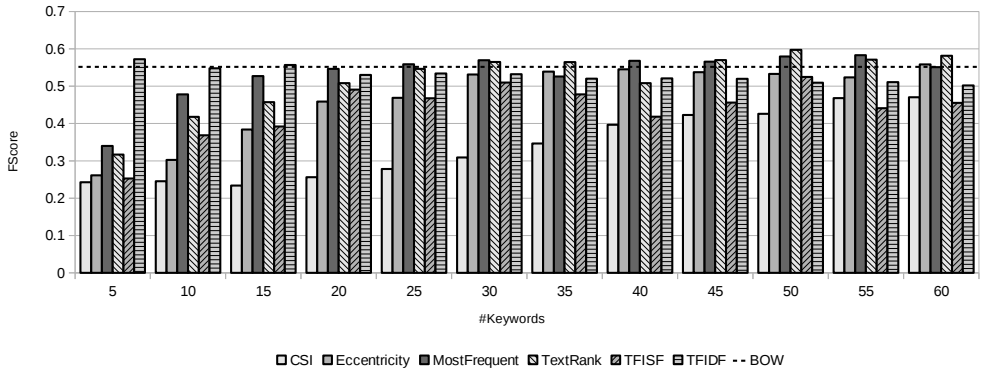


Figure 8: Analysis of the number of keywords in the quality of incremental clustering considering different methods for ACM-1.

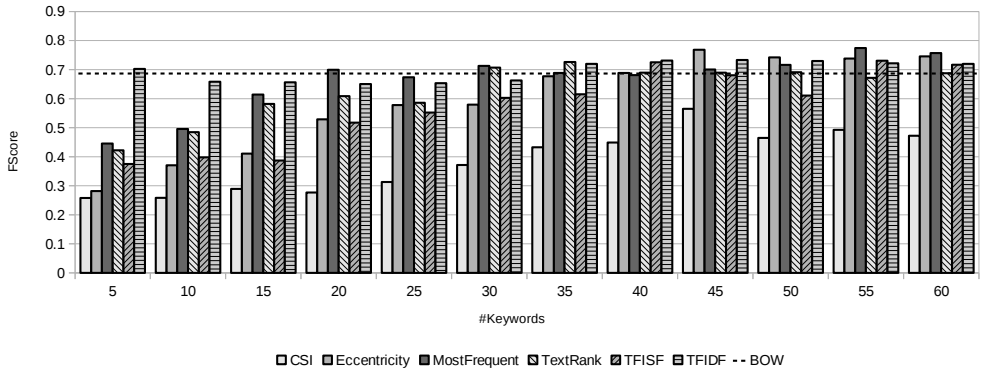


Figure 9: Analysis of the number of keywords in the quality of incremental clustering considering different methods for ACM-2.

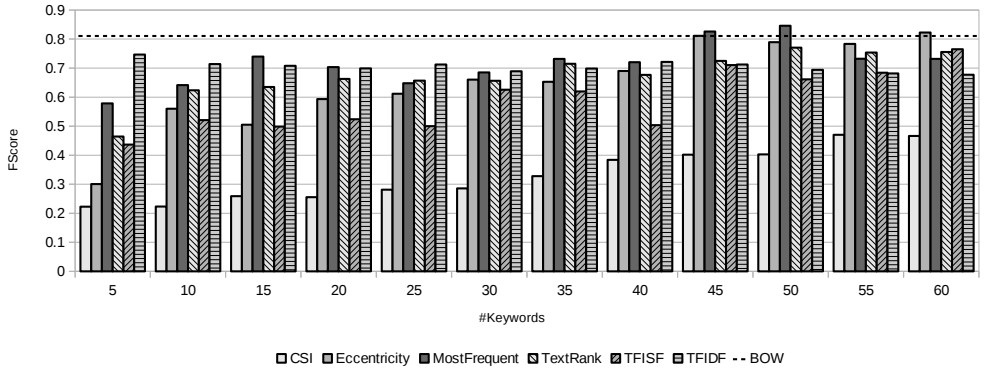


Figure 10: Analysis of the number of keywords in the quality of incremental clustering considering different methods for ACM-3.

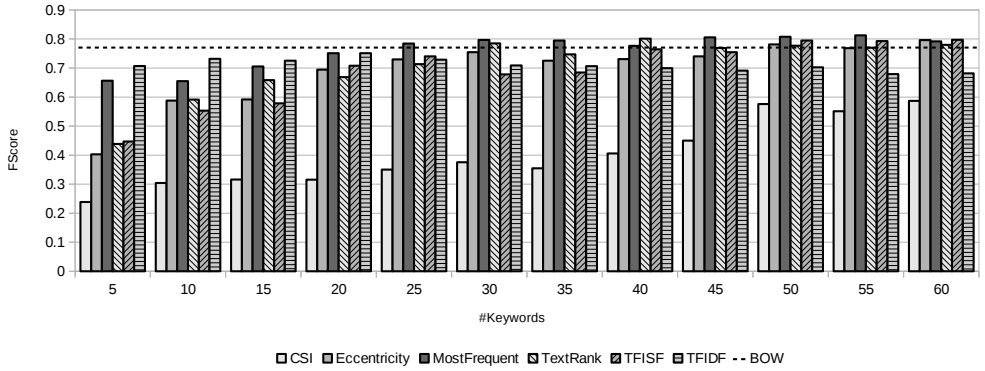


Figure 11: Analysis of the number of keywords in the quality of incremental clustering considering different methods for ACM-4.

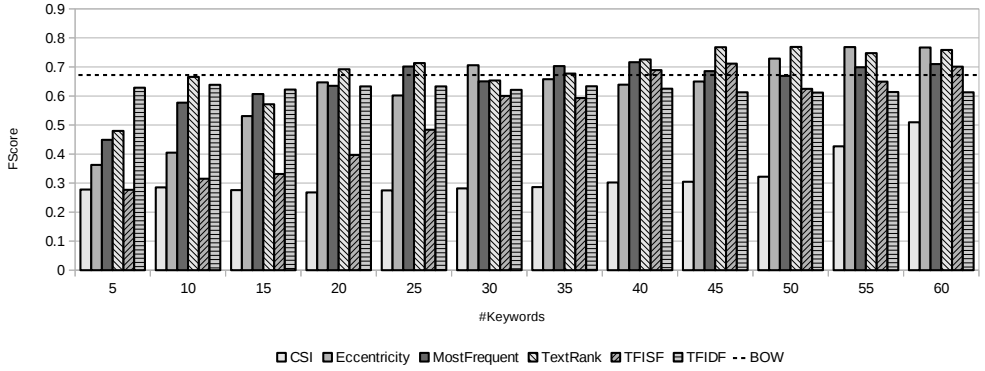


Figure 12: Analysis of the number of keywords in the quality of incremental clustering considering different methods for ACM-5.

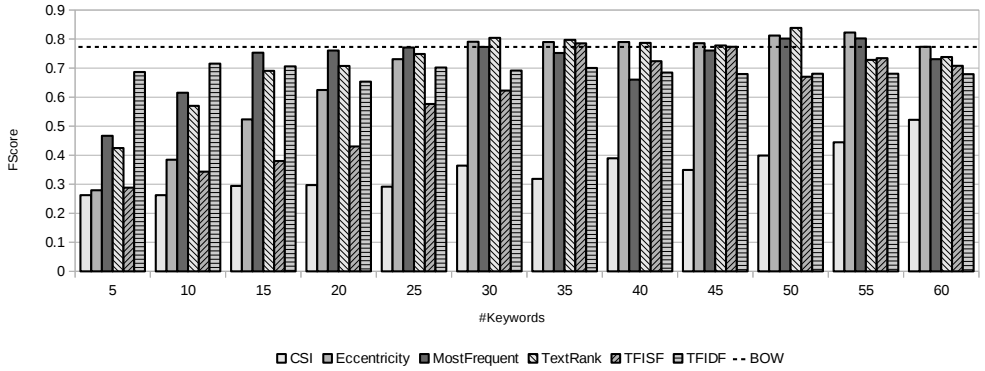


Figure 13: Analysis of the number of keywords in the quality of incremental clustering considering different methods for ACM-6.

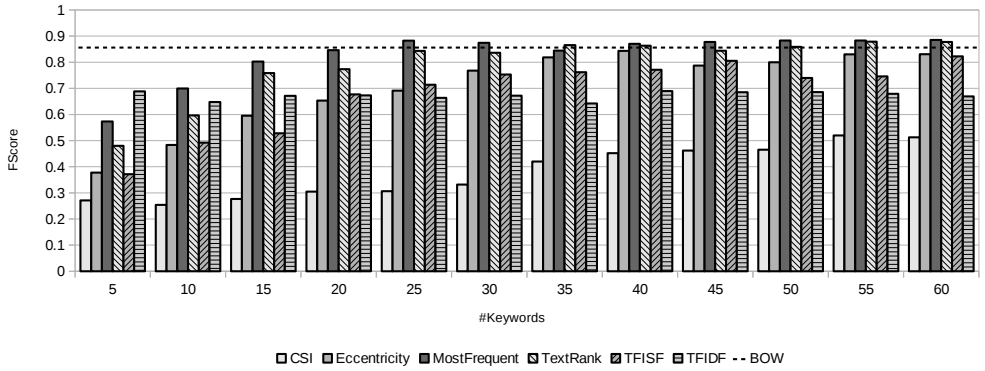


Figure 14: Analysis of the number of keywords in the quality of incremental clustering considering different methods for ACM-7.

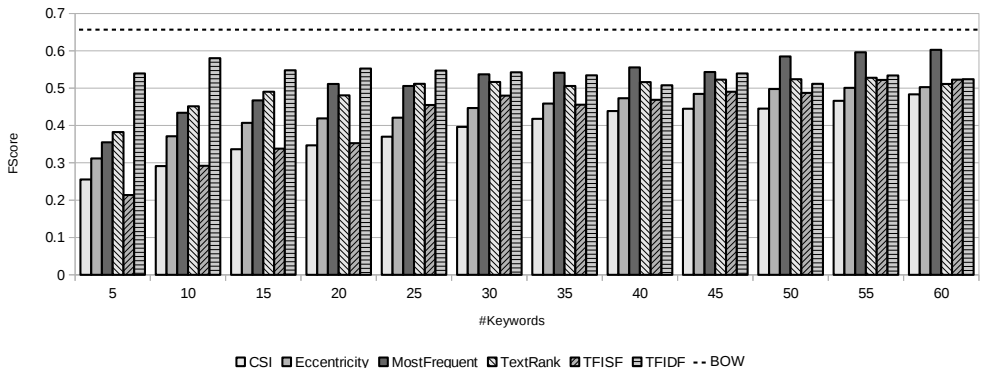


Figure 15: Analysis of the number of keywords in the quality of incremental clustering considering different methods for ACM-8.

B Analysis of the Number of Keywords in the Quality of Incremental Clustering for Short Text Documents

Figures 16 – 23 present the trade-off among *F-Score* and the different number the keywords extracted by each method. The presented *F-Score* is the average value the of *F-Score* from the 30 best configurations from each method. The graphic show a line which represents the use of all features (BOW).

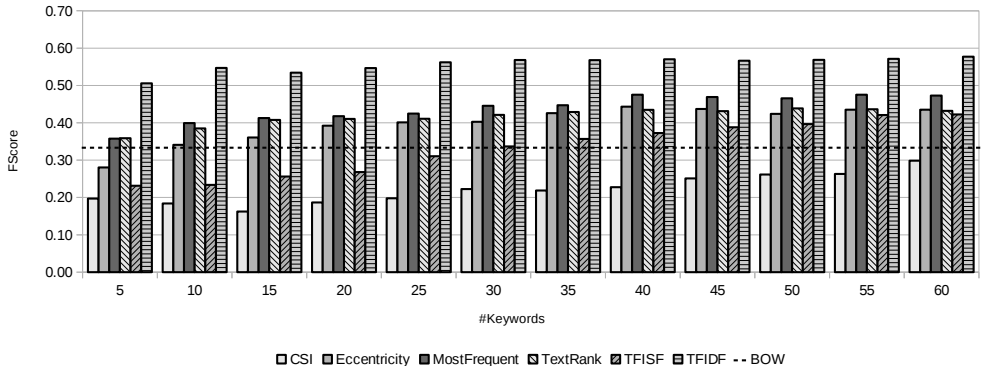


Figure 16: Analysis of the number of keywords in the quality of incremental clustering considering different methods for Estadao-1.

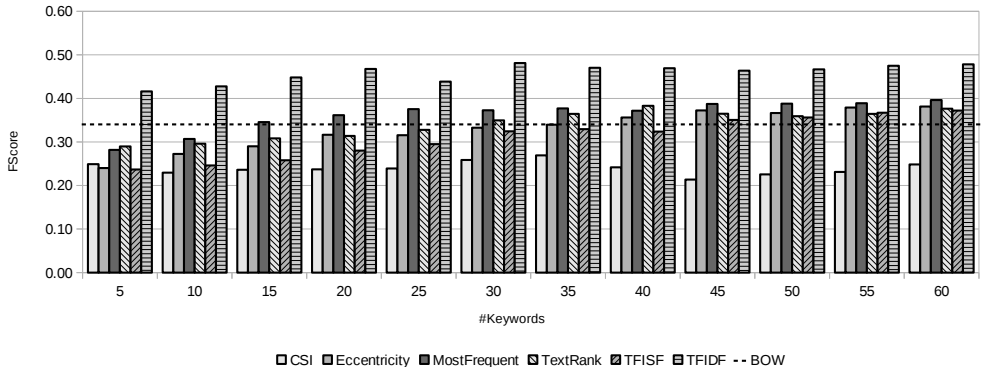


Figure 17: Analysis of the number of keywords in the quality of incremental clustering considering different methods for Estadao-2.

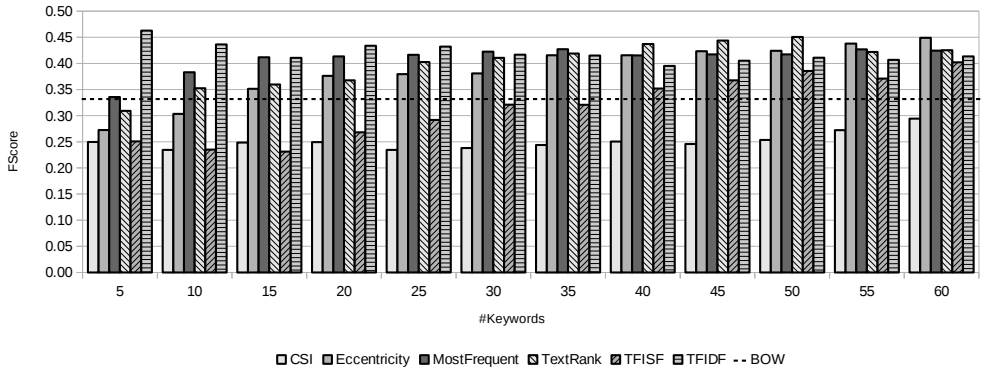


Figure 18: Analysis of the number of keywords in the quality of incremental clustering considering different methods for Estadao-3.

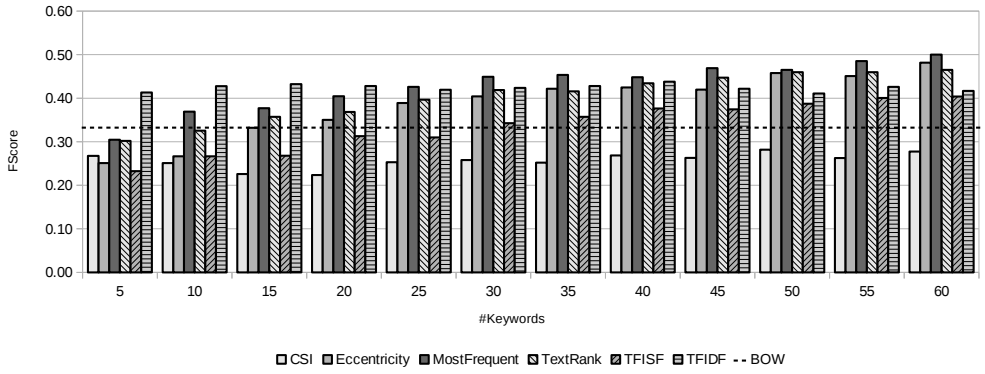


Figure 19: Analysis of the number of keywords in the quality of incremental clustering considering different methods for Estadao-4.

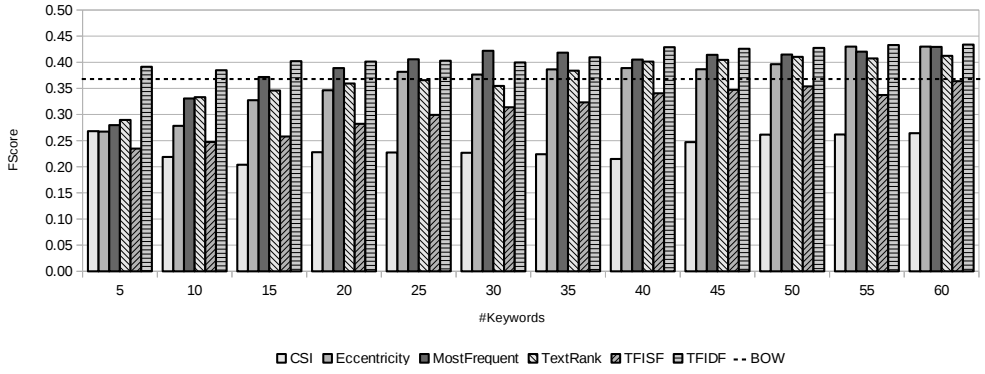


Figure 20: Analysis of the number of keywords in the quality of incremental clustering considering different methods for Estadao-5.

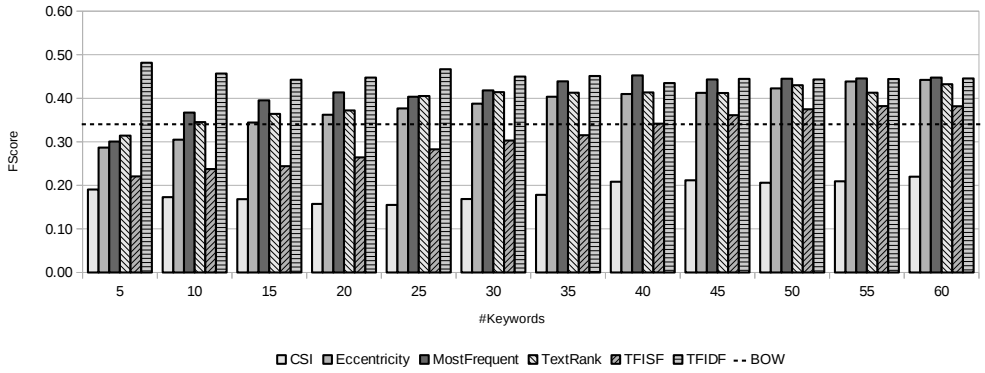


Figure 21: Analysis of the number of keywords in the quality of incremental clustering considering different methods for Estadao-6.

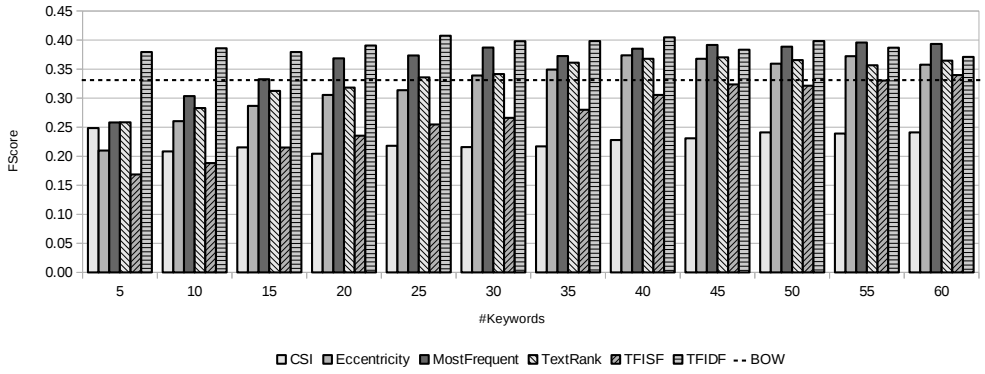


Figure 22: Analysis of the number of keywords in the quality of incremental clustering considering different methods for Estadao-7.

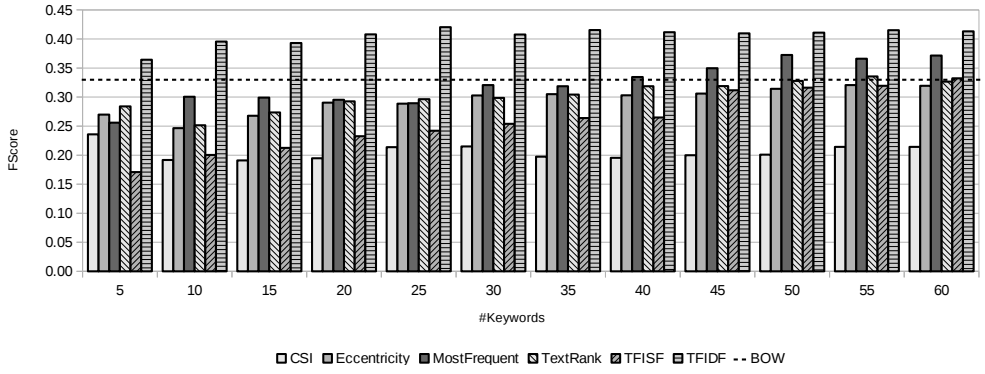


Figure 23: Analysis of the number of keywords in the quality of incremental clustering considering different methods for Estadao-8.