Natural Language Engineering **22** (5): 687–726. © Cambridge University Press 2015
doi:10.1017/S1351324915000273

Silhouette + attraction: A simple and effective method for text clustering†

MARCELO L. ERRECALDE[1],
LETICIA C. CAGNINA[1] and PAOLO ROSSO[2]

[1]*LIDIC, Universidad Nacional de San Luis, San Luis, Argentina*
e-mail: merreca@unsl.edu.ar, lcagnina@gmail.com
[2]*NLE Lab, PRHLT Research Center, Universitat Politècnica de València, Valencia, España*
e-mail: prosso@dsic.upv.es

(*Received 6 December 2013; revised 10 July 2015; accepted 13 July 2015; first published online 14 August 2015*)

Abstract

This article presents silhouette–attraction (*Sil–Att*), a simple and effective method for text clustering, which is based on two main concepts: the *silhouette coefficient* and the idea of *attraction*. The combination of both principles allows us to obtain a general technique that can be used either as a boosting method, which improves results of other clustering algorithms, or as an independent clustering algorithm. The experimental work shows that *Sil–Att* is able to obtain high-quality results on text corpora with very different characteristics. Furthermore, its stable performance on all the considered corpora is indicative that it is a very robust method. This is a very interesting positive aspect of *Sil–Att* with respect to the other algorithms used in the experiments, whose performances heavily depend on specific characteristics of the corpora being considered.

1 Introduction

Text clustering is the unsupervised assignment of documents to unknown categories. This task is more difficult than supervised text categorization because usually no information about categories and correctly categorized documents is provided in advance. Text clustering is a very important task due to the crucial role that textual information plays in our daily activities. Most of the information related to business is available in text documents and the huge amount of (textual) information that the Web makes available nowadays, offers an unlimited number of opportunities to use this information in relevant research problems and business applications.

†This research work has been partially funded by UNSL, CONICET (Argentina), DIANA-APPLICATIONS-Finding Hidden Knowledge in Texts: Applications (TIN2012-38603-C02-01) research project, and the WIQ-EI IRSES project (grant no. 269180) within the FP 7 Marie Curie People Framework on Web Information Quality Evaluation Initiative. The work of the third author was done also in the framework of the VLC/CAMPUS Microcluster on Multimodal Interaction in Intelligent Systems.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Text clustering, and clustering in general, has been traditionally used either for *understanding* or *utility* (Tan, Steinbach and Kumar 2005). The former refers to the idea of using text clustering in problems where the only aspect to be addressed is the organization and understanding of documents as an independent task. The latter considers text clustering as a pre/post-processing stage which serves as support of other relevant tasks, such as information retrieval and browsing (Cutting *et al.* 1992; Charikar *et al.* 2004; Liu and Croft 2004; Hearst 2006), text summarization (Larocca Neto *et al.* 2000; Takeda and Takasu 2007), topic discovery/identification (Stein and Meyer zu Eissen 2004; Pons-Porrata, Berlanga-Llavori and Ruiz-Shulcloper 2007) and supervised text classification (Kyriakopoulou 2008) among others. In both cases, any progress in obtaining simple and effective text clustering methods can have a direct effect in many relevant scientific and business problems.

A significant number of approaches have been proposed for text clustering in recent years (Steinbach, Karypis and Kumar 2000; Ng, Jordan and Weiss 2001; Zha *et al.* 2001; Berry 2003; Xu, Liu and Gong 2003; Jing 2005; Zhao, Karypis and Fayyad 2005). In particular, some recent bio-inspired proposals have gained increasing interest in *short*-text clustering. These approaches include algorithms based on *Particle Swarm Optimization* techniques (Cagnina *et al.* 2008; Ingaramo *et al.* 2009) and ant-behavior-based approaches (Errecalde, Ingaramo and Rosso 2010; Ingaramo, Errecalde and Rosso 2010a, 2010b).

However, despite this diversity of methods, most of them only work properly when documents, or document collections, have specific characteristics. In many cases, they use complex heuristics that exploit some peculiarities of the corpus under consideration, but their performances sharply degrade when they are used in other more general document collections. Therefore, the need for simple, general, effective and robust methods for clustering collections with widely varying characteristics becomes evident.

In this paper, we make a contribution in this area by proposing silhouette–attraction (*Sil–Att*), a robust method which is able to obtain high-quality results on text corpora with very different characteristics. In order to get a better understanding of our proposal, the present work will analyze some important aspects which were not considered in our preliminary studies with bio-inspired approaches. First of all, the two key mechanisms that seem to be essential to achieve good results in those works, the *silhouette coefficient* and the *attraction concept*, are examined. Then, a description of how these mechanisms can be combined in a simpler and effective method, *Sil–Att*, is proposed. *Sil–Att* focusses on the beneficial effects that the *Sil–Att* combination seems to obtain in iterative processes for text clustering tasks without considering any bio-inspired principles.

It is interesting to note that *Sil–Att* can be used as a boosting method which improves results generated by other clustering methods or it can be used as an effective and independent clustering algorithm. In this work, we present a detailed experimental study about the performance of *Sil–Att* on text corpora with very different characteristics. Our study also includes a detailed analysis of how often the *Sil–Att* method succeeds in improving previous clustering results and those situations in which the *Sil–Att* method does not obtain as good results as expected.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Finally, we present some interesting observations about the independence of *Sil–Att* on the quality of the initial clusterings and discuss how this aspect gives origin to the *Sil–Att* method as a complete clustering algorithm.

The rest of this paper is organized as follows. Section 2 introduces the two main concepts that were used in our work: the silhouette coefficient and the concept of attraction. Section 3 describes how these concepts were combined in a single method: *Sil–Att*. The experimental setup and the analysis of the results are provided in Section 4. Next, in Section 5 some related works are presented and the connections with our proposal are established. Finally, in Section 6 some general conclusions are drawn and possible future work is discussed.

2 Background

2.1 The silhouette coefficient

In realistic document clustering problems, information about categories and correctly categorized documents is not provided beforehand. An important consequence of this lack of information is that results cannot usually be evaluated with typical *external* measures like the *F-Measure* (FM) (van Rijsbergen 1979) or the *entropy* (Shannon 1948; Zhao and Karypis 2004), because 'correct' categorizations specified by a human expert are not available. Therefore, the quality of the resulting groups is evaluated with respect to *structural* properties expressed in different *Internal Clustering Validity Measures* (ICVMs). Classical ICVMs used as cluster validity measures include the *Dunn* (Dunn 1974; Bezdek and Pal 1995) and *Davies–Bouldin* (Davies and Bouldin 1979) indexes, the *Global Silhouette* (GS) coefficient (Rousseeuw 1987; Kaufman and Rousseeuw 1990) and new graph-based measures such as the *Expected Density Measure* and the *λ-Measure* (Stein, Meyer zu Eissen and Wißbrock 2003).

Most of researchers working on clustering problems are familiar with the use of these unsupervised measures of cluster validity as cluster validation tools. However, some recent works have proposed other uses of this kind of measures that include the hardness estimation of corpora for document clustering problems (Pinto and Rosso 2007; Errecalde, Ingaramo and Rosso 2008) and its use as objective functions in optimization-based clustering methods (Selim and Alsultan 1991; Zhao and Karypis 2004; Cagnina *et al.* 2008; Ingaramo *et al.* 2009).

Among the considerable number of ICVMs proposed up to now, the GS coefficient has shown good results as cluster validation method with respect to other well-known validity measures (Brun *et al.* 2007). Furthermore, the silhouette coefficient has also shown its potential for determining the optimal number of groups in a clustering problem (Rousseeuw 1987; Tan *et al.* 2005; Choi *et al.* 2011), estimating how difficult a corpus is for an arbitrary clustering algorithm (Errecalde *et al.* 2008), computing a target function to be optimized (Cagnina *et al.* 2008; Ingaramo *et al.* 2009), automatically determining a threshold for a similarity function (Bonato dos Santos *et al.* 2011) and as a key component in other internal process of clustering algorithms (Aranganayagi and Thangavel 2007; Errecalde *et al.* 2010; Ingaramo *et al.* 2010a).

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

The GS coefficient combines two key aspects to determine the quality of a given clustering: *cohesion* and *separation*. Cohesion measures how closely related are objects in a cluster whereas separation quantifies how distinct (well-separated) a cluster from other clusters is. The GS coefficient of a clustering is the average *cluster silhouette* of all the obtained groups. The cluster silhouette of a cluster C also is an average silhouette coefficient but, in this case, of all objects belonging to C. Therefore, the fundamental component of this measure is the formula used to determine the silhouette coefficient value of any arbitrary object i, that will be referred as $s(i)$ and it is defined in (1).

$$s(i) = \frac{b(i) - a(i)}{max(a(i), b(i))} \tag{1}$$

with $-1 \leq s(i) \leq 1$. The $a(i)$ value denotes the average dissimilarity of the object i to the remaining objects in its own cluster, and $b(i)$ is the average dissimilarity of the object i to all objects in the nearest cluster. From this formula, it can be observed that negative values for this measure are undesirable and that we want for this coefficient values as close to 1 as possible. Thus, for example, in Figure 1, two silhouette graphics representing clusterings of very different quality are shown. The graphic on the left shows that most of $s(i)$ values are near to 1, indicating an adequate membership level of nearly all elements to their assigned groups (high-quality clustering); on the other hand, the graphic on the right clearly shows a low quality clustering with a significant number of elements with low and negative values of $s(i)$.

In Section 3, it will be shown that silhouette coefficient information of a grouping not only is an appealing device to clearly visualize aspects related to the quality of groups; it can also be a fundamental tool to determine the order in which the documents of a grouping should be considered in the *Sil–Att* method proposed in this work.[1]

2.2 An attraction-based comparison

Iterative clustering approaches like AntTree (Azzag *et al.* 2003), K-means (MacQueen 1967) and K-medoids (Kaufman and Rousseeuw 1990) progressively add documents in different groups until all the documents have been considered (as *Sil–Att* does). These approaches have a *prototype-based view* of how this group should be selected in order to add a document. In this view, each cluster has associated an object (prototype) which will be used as a representative for the group being built. Thus, if p_j denotes the prototype of the cluster j, the document d will be incorporated in the group m whose prototype p_m is the most similar to d. It is important to notice that this prototype-based comparison has been used in classical clustering algorithms like K-means and K-medoids which take a 'center-based' view

[1] From now on, the expression '*silhouette coefficient information*' will denote general silhouette values that can correspond to a whole clustering, a particular group or a single object. The expression '*GS coefficient*' in contrast, will only be used in those cases where its use as an ICVM – which evaluates the quality of the whole clustering – needs to be emphasized.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

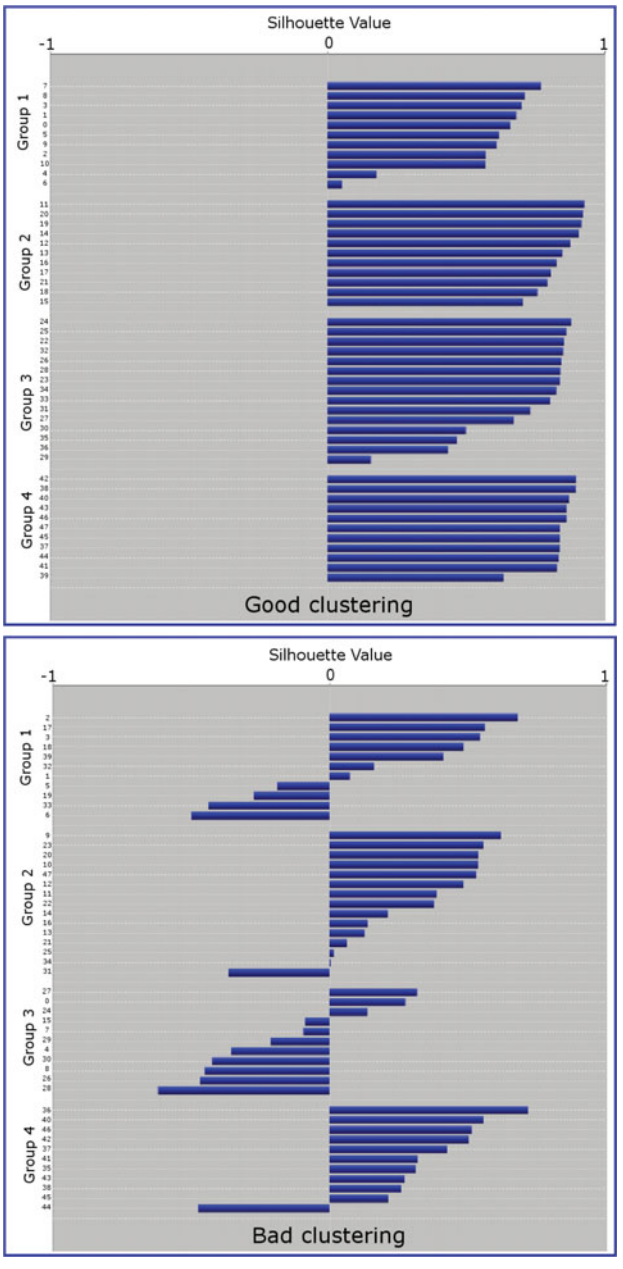


Fig. 1. (Colour online) Silhouette graphics. Examples of good and bad clusterings.

of prototypes, considering as representative of the cluster the centroid (K-means) and the medoid (K-medoid). It is also present in new 'biological' clustering approaches in which the role of prototypes (or representatives) are accomplished by 'creatures' with some particular characteristics like the ants connected to a support in the AntTree (Azzag *et al.* 2003) algorithm.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

An alternative view to this prototype-based approach, and the one used in *Sil–Att*, is to estimate the similarity between the document d being processed and the potential clusters where this document can be added, taking into account all the documents that have already been incorporated in the cluster instead of a prototype of this cluster. In that way, each group can be considered as exerting some kind of 'attraction' on the documents to be clustered. This idea of 'attraction' was already posed in Stein and Meyer zu Eissen (2002), where it was used as an efficient way of obtaining 'dense' groups. In the present work, we will give a more general sense to the concept of attraction by considering that the 'attraction level' that a group $\mathscr{G}$ exerts on an arbitrary document d, can be computed by *any plausible estimation of the quality of the group that would result if d were incorporated to* $\mathscr{G}$. This intuitive idea of 'attraction' is formalized in the following definition.

2.2.1 *Attraction of a group*

Let $\mathscr{G}$ be a group (set) of documents and d be an arbitrary document. Let $\mathscr{D}$ be the universe of possible documents to be considered. The *attraction* of a group $\mathscr{G}$ on the document d, denoted as $att(\mathscr{G}, d)$ is a mapping $att: 2^{\mathscr{D}} \times \mathscr{D} \rightarrow \mathbb{R}$ such that if $att(\mathscr{G}, d) = a$, $a \in \mathbb{R}$ is an estimation of the quality of the group $\mathscr{G} \cup \{d\}$.

To compute $att(\mathscr{G}, d)$, any ICVM that allows to estimate the quality of individual clusters can be applied to $\mathscr{G} \cup \{d\}$. An effective attraction measure (and the one used in this work) is the average similarity *Sim* between d and all the documents in $\mathscr{G}$ as shown in (2).

$$att(\mathscr{G}, d) = \frac{\sum_{d_i \in \mathscr{G}} Sim(d, d_i)}{|\mathscr{G}|} \tag{2}$$

The next section shows how the silhouette coefficient and the idea of attraction can be combined in a simple and effective method for text clustering.

3 The *Sil–Att* method

The *Sil–Att* is a simple and effective technique for text clustering that could also be used in more general clustering scenarios. The main idea behind this method is the combined use of silhouette coefficient information of a given clustering, with the incremental generation of groups based on the attraction that each group exerts on the document being clustered.

Figure 2 gives a concise description of the main steps involved in the *Sil–Att* algorithm. It starts considering an initial clustering $\mathscr{C}$, which can be given by the user or some previous clustering process (*Sil–Att* as a boosting method) or automatically generated by a random clustering process (*random_initial_cluster*()) when no clustering is provided as input ($\mathscr{C} = NULL$). In the first case, *Sil–Att* will attempt to improve the quality of the clustering supplied by the user (or other clustering algorithm). In the second one, *Sil–Att* will act as an independent clustering algorithm which will only require the initial grouping generated by *random_initial_cluster*(). Currently, this procedure is only intended to generate random (uniformly distributed) clusterings

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

function *Sil-Att*($\mathcal{C}$) **returns** a clustering $\mathcal{C}^\star$
input: $\mathcal{C} = \{C_1, \ldots, C_k\}$, an initial grouping or $NULL$

if ($\mathcal{C} = NULL$) **then** $\mathcal{C} = random_initial_cluster()$;
1. Generation of initial singletons
1.a. Create a set $\mathcal{Q} = \{q_1, \ldots, q_k\}$ of k data queues (one queue for each group $C_j \in \mathcal{C}$).
1.b. Sort each queue $q_j \in \mathcal{Q}$ in decreasing order according to the silhouette coefficient of its elements. Let $\mathcal{Q}' = \{q'_1, \ldots, q'_k\}$ be the resulting set of ordered queues.
1.c. Let $\mathcal{G}_\mathcal{F} = \{d_1^1, \ldots, d_k^1\}$ be the set formed by the first document d_i^1 of each queue $q'_i \in \mathcal{Q}'$. For each document $d_i^1 \in \mathcal{G}_\mathcal{F}$, remove d_i^1 from q'_i and set $\mathcal{G}_i = \{d_i^1\}$ (generate singleton $\mathcal{G}_i$).
2. Generation of the $\mathcal{L}$ list
2.a. Let $\mathcal{Q}'' = \{q''_1, \ldots, q''_k\}$ the set of queues resulting from the previous process of removing the first document d_i^1 of each queue in $\mathcal{Q}'$. Generate the $\mathcal{L}$ list by merging the queues in $\mathcal{Q}''$, taking one document from each q''_i following a round-robin policy.
3. Clustering process
3.a. Repeat
3.a.1 Select (simultaneously removing) the first document $\hat{d}$ from $\mathcal{L}$.
3.a.2 Let $\mathcal{G}^+$ the $\mathcal{G}_i$ with the highest $att(\mathcal{G}_i, \hat{d})$ value.
$$\mathcal{G}^+ \leftarrow \mathcal{G}^+ \cup \{\hat{d}\}$$
Until $\mathcal{L}$ is empty
Let $\mathcal{C}^\star = \{\mathcal{G}_1, \ldots, \mathcal{G}_k\}$ be the clustering obtained in Step 3.
if ($stop_condition(\mathcal{C}, \mathcal{C}^\star)$) **then return** $\mathcal{C}^\star$;
return *Sil-Att*($\mathcal{C}^\star$)

Fig. 2. The *Sil–Att* algorithm.

based on the set of documents and the number of groups (k) specified by the user.[2] However, other more elaborated approaches that also allow to specify non-uniform probability distributions for the different groups would be valid alternatives.

Once the initial clustering $\mathscr{C}$ is determined, three main processes take place in *Sil–Att*: (1) the generation of initial singletons, (2) the generation of the $\mathscr{L}$ list and (3) the incremental clustering process.

In the first process, the documents in each group $C_i \in \mathscr{C}$ are sorted according to their silhouette coefficient values, from highest to lowest. Then, the document d_i with the highest silhouette coefficient value of each group is selected as the initial representative of the C_i group, and a singleton $\mathscr{G}_i = \{d_i\}$ is generated for each $C_i \in \mathscr{C}$. After that, the $\mathscr{L}$ list with the remaining documents is generated considering again the silhouette coefficient information previously computed. The idea in this process is to take one document from each ordered queue in $\mathscr{Q}''$ following a round-robin policy, until all the queues are empty. Since the documents of each ordered queue are

[2] *Sil–Att*, the same as the remaining algorithms considered in the experimental work, received as input from the user the information about the number k of groups. When this information is not available, the silhouette coefficient could also be used for determining the optimal number of groups, as proposed in Rousseeuw (1987), Tan *et al.* (2005), Choi *et al.* (2011).

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

taken from head to tail, we can assure that the order that two documents had in an ordered queue $q \in \mathcal{Q}''$ will be preserved in the resulting combined $\mathcal{L}$ list. However, the silhouette coefficients of two documents taken from two different groups will not necessarily be in decreasing order in $\mathcal{L}$. As it was observed in some previous works related to our proposal (Azzag *et al.* 2003; Ingaramo *et al.* 2010b), the order in which these documents are later considered (determined by the $\mathcal{L}$ list) can directly affect the resulting clusters. For instance, in *Sil–Att*, the way the $\mathcal{L}$ list is generated favors a good 'mixture' of good representatives of each group in the first positions of $\mathcal{L}$ and a balanced number of documents in each group in the initial iterations of Step 3. In that way, when the 'difficult' documents (the ones with a low silhouette coefficient) need to be assigned to a particular group, this decision is made taking into account an acceptable and balanced number of the 'best' documents of each group (according to the silhouette coefficient).

Finally, the third process generates the new groups of the clustering by iteratively considering each document in $\mathcal{L}$ and placing this document in the *most appropriate* group. The decision of which of all the available groups will be considered as the most appropriated to include a current document $\hat{d}$ is based on the attraction level that each group exerts on $\hat{d}$. More formally, $\hat{d}$ will be incorporated in the group $\mathcal{G}^+$, such that

$$\mathcal{G}^+ = \arg\max_{\mathcal{G}_i} att(\mathcal{G}_i, \hat{d}). \tag{3}$$

Once the previous process has finished, the last step in the algorithm consists in determining whether the resulting group will be returned as final result of *Sil–Att* or it will be used as input clustering for a new execution of *Sil–Att*. The boolean *stop_condition*() function performs this decision considering for that task the initial clustering ($\mathcal{C}$) and the recently generated one ($\mathcal{C}^\star$).

As can be seen in Figure 2, two main aspects need to be specified by the user of *Sil–Att*: (i) the type of use of *Sil–Att* (as boosting method or as independent clustering method); and (ii) the criterion to be used in the *stop_condition*() function.

With respect to the first point, this is simply determined by providing a $NULL$ argument to *Sil–Att* when it must operate as a clustering method. From now on, this case will be denoted as *Sil–Att*($NULL$). On the other hand, if *Sil–Att* is intended to improve a clustering $\mathcal{C}_{Alg}$ obtained with some arbitrary clustering algorithm Alg, this usage of *Sil–Att* as boosting method will be denoted as *Sil–Att*($\mathcal{C}_{Alg}$).

The criterion used in the *stop_condition*() function to determine whether *Sil–Att* must continue iterating or not is another relevant aspect. The most simple alternative consists in making this function always return a constant $True$ value. In that case, *Sil–Att* will execute only *one* improvement processing step on an initial clustering $\mathcal{C}$. This use of *Sil–Att* will be denoted from now on as *Sil–Att*($\mathcal{C}$)[1]. Another alternative implementation for *stop_condition*() consists in using a general stop criterion that will be usually based on the recently generated clustering ($\mathcal{C}^\star$) and the previous one ($\mathcal{C}$). In this case, the usual criterion, and the one used in the present work, will consist in stopping the process when no change of elements among different groups is observed. However, other more general criteria could also be used; a simple alternative, for example, would consist in executing the process a maximum

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

number of iterations m specified by the user. When *stop_condition*() is implemented in that way, *Sil–Att* will iteratively perform an arbitrary number of improvement processing steps on an arbitrary initial clustering $\mathscr{C}$, until *stop_condition*() returns *True*. This iterative operating mode of *Sil–Att* will be denoted as $Sil\text{–}Att(\mathscr{C})^{\star}$.

From the above discussion, it is clear that *Sil–Att* can operate in four distinct operation modes, which result from the combination of both aspects:

(1) $Sil\text{–}Att(NULL)^{1}$: *Sil–Att* operates as an independent clustering algorithm that only performs one improvement step.
(2) $Sil\text{–}Att(NULL)^{\star}$: *Sil–Att* operates as an independent clustering algorithm which iteratively improves the clustering generated in the previous iteration until *stop_condition*() holds.
(3) $Sil\text{–}Att(\mathscr{C}_{Alg})^{1}$: *Sil–Att* improves a clustering $\mathscr{C}_{Alg}$, previously obtained with some arbitrary clustering algorithm Alg, performing only one improvement step.
(4) $Sil\text{–}Att(\mathscr{C}_{Alg})^{\star}$: *Sil–Att* takes a clustering $\mathscr{C}_{Alg}$ obtained with some arbitrary clustering algorithm Alg, and iteratively improves the clustering generated in the previous iteration until *stop_condition*() holds.

It is interesting to note that *Sil–Att* (and particularly the $Sil\text{–}Att()^{\star}$ variants) has some similarities with classical *iterative* clustering approaches such as K-means (MacQueen 1967), or K-medoids (Kaufman and Rousseeuw 1990; Ng and Han 1994). These algorithms start with some initial points representing the K different groups and strive to successively improve the existing set of clusters by changing these 'representative' objects (or *prototypes*). Classical prototypes are those most *centrally* located within their clusters such as the *centroids* (in K-means) or the *medoids* (in K-medoids). In that way, the task of finding K clusters in these iterative prototype-based approaches can be simply understood as the task of determining an appropriate representative object for each cluster (Ng and Han 1994).

Sil–Att also tries to improve the clustering obtained in a previous iteration but its manner of operating is completely different with respect to the way these algorithms work. First of all, *Sil–Att* is an iterative method but it cannot be considered a prototype-based approach. Algorithms like K-means or K-medoids decide which cluster a document should be added to, by only considering the distance of the document to the nearest prototype of each cluster. The *Sil–Att* method instead, considers the *attraction* that the whole *current* groups exert on the object being considered. This implies that, unlike prototype-based approaches which keep the prototypes fixed while the objects are being compared, the attraction of the different groups in *Sil–Att* changes as new documents are added to these clusters. This means that the order in which the objects are considered in the step 3 of Figure 2 (determined by the $\mathscr{L}$ list) can directly affect the resulting clustering of each iteration of the algorithm.

The above comment shows a key difference between *Sil–Att* and the iterative prototype-based approaches. Our proposal based on the silhouette coefficient not only determines which will be the first elements that the initial groups will have, it also determines the order in which the remaining documents will be considered and hence the resulting clustering in each iteration. This aspect is not usually considered

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

in prototype-based approaches which only focus on methods to obtain good initial prototypes (or 'seeds') (Hasan *et al.* 2009; Paterlini, Nascimento and Traina Jr. 2011) and only consider how to update the prototypes in each iteration.

The computational complexity of the *Sil–Att* algorithm can be analyzed in terms of the number of documents n to be clustered and the number of iterations required for the convergence I.[3] The main steps involved in the process carried out by *Sil–Att* algorithm are detailed as follows:

Step 1: The creation of the similarity matrix takes $\frac{(n)*(n-1)}{2}$. Then, this step uses $T_{Step1} : \frac{(n^2-n)}{2}$.
Step 2: Computation of the silhouette coefficients takes $\frac{(n)*(n-1)}{2}$. Then, this step takes $T_{Step2} : \frac{(n^2-n)}{2}$.
Step 3: Sorting of documents takes $n * \log n$. Then, step 3 uses $T_{Step3} : n * \log n$.
Step 4: Merging process to generate the $\mathscr{L}$ list takes n. Then, $T_{Step4} : n$.
Step 5: The cluster determination takes $\frac{(n)*(n-1)}{2}$. Then, this step uses $T_{Step5} : \frac{(n^2-n)}{2}$.

The similarity matrix is generated only once by computing the similarity for each pair of documents in the collection. This is possible for static corpus clustering tasks although for dynamic text collections, a different and more efficient (lower computational cost) implementation should be used. See Conclusions section for more details. Steps 2 to 5 are repeated I times, then the total computational complexity of *Sil–Att* is $T_{Sil-Att} : T_{Step1} + I * (T_{Step2} + T_{Step3} + T_{Step4} + T_{Step5}) = \frac{n^2-n}{2} + I * n^2 + I * n * \log n$ which is $\mathscr{O}(n^2)$.

4 Experimental setting and analysis of results

For the experimental work, fourteen corpora with different levels of complexity with respect to the size, length of documents and vocabulary overlapping were selected: Micro4News, EasyAbstracts, SEPLN-CICLing, CICLing-2002-A, CICLing-2002-F, R4, R6, R8-, R8+, R8B, JRC6, R8-Test, JRC-Full and R8-Train. Table 1 shows some general features of these corpora: corpus size expressed in K bytes, number of categories and documents ($|\mathscr{C}|$ and $|\mathscr{D}|$ respectively), total number of term occurrences in the collection ($|\mathscr{T}|$), vocabulary size ($|\mathscr{V}|$) and average number of term occurrences per document ($\overline{\mathscr{T}}_d$). *RH*, which stands for *Relative Hardness* (Pinto and Rosso 2007), is a specific measure which aims at estimating how related the topics corresponding to the different categories of the grouping are, that is, how difficult a corpus would be for a general clustering task. An alternative to estimate this aspect consists in using a simple vocabulary overlapping calculation among the vocabularies of the distinct categories of the corpus. Different set overlapping measures could be used for this purpose and, in the present work, the Jaccard coefficient among the vocabularies of the categories was used, resulting in the following definition: let $\mathscr{C}$ a corpus with n

[3] I value is often very small. In most of experimental instances, the number of iterations required for convergence ranged from 1 to 4.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Table 1. *Features of the corpora used in the experimental work: corpus size (CS) in Kb, number of categories ($|\mathscr{C}|$), number of documents ($|\mathscr{D}|$), total number of term occurrences ($|\mathscr{T}|$), vocabulary size ($|\mathscr{V}|$), average number of term occurrences per document ($\overline{\mathscr{T}}_d$) and Relative Hardness (RH)*

Corpora /Features	*CS*	$\lvert\mathscr{C}\rvert$	$\lvert\mathscr{D}\rvert$	$\lvert\mathscr{T}\rvert$	$\lvert\mathscr{V}\rvert$	$(\overline{\mathscr{T}}_d)$	*RH*
Micro4News	443	4	48	125,614	12,785	2,616.95	0.16
EasyAbstracts	44.9	4	48	9,261	2,169	192.93	0.18
SEPLN-CICLing	25	4	48	3,143	1,169	65.48	0.14
CICLing-2002-A	26.3	4	48	3,382	953	70.45	0.22
CICLing-2002-F	518	4	48	80,061	7,307	1,667.9	0.26
R4	184	4	266	27,623	4,578	166.4	0.19
R6	356	6	536	53,494	4,600	99.8	0.21
R8-	44.3	8	445	8,481	1,876	19.06	0.04
R8+	440	8	445	66,314	7,797	149.02	0.16
R8B	474	8	816	71,842	5,854	88.04	0.19
JRC6	9,185.2	6	563	1,420,558	68,219	2,523.19	0.11
R8-Test	767	8	2,189	208,099	11,975	95.06	0.08
JRC-Full	23,654.4	6	2,816	4,133,888	79,658	1,468	0.14
R8-Train	2,150.4	8	5,485	587,453	19,984	107.10	0.07

categories $Cat_1, \ldots, Cat_n$. The *Relative Hardness* of $\mathscr{C}$, $RH(\mathscr{C})$, is defined in (4).

$$RH(\mathscr{C}) = \frac{1}{n * (n-1)/2} * \sum_{j,k=1;j<k}^{n} sim(Cat_j, Cat_k) \tag{4}$$

where each category Cat_j is considered as the 'document' obtained by concatenating all the documents in Cat_j and the similarity (*sim*) between two categories Cat_j, Cat_k, is calculated as (5) shows.

$$sim(Cat_j, Cat_k) = \frac{|Cat_j \bigcap Cat_k|}{|Cat_j \bigcup Cat_k|} \tag{5}$$

The first five corpora (Micro4News, EasyAbstracts, SEPLN-CICLing, CICLing-2002-A, CICLing-2002-F) are small corpora with the same number of documents (48) and categories (4). These data sets were intensively used in different works (Alexandrov, Gelbukh and Rosso 2005; Errecalde *et al.* 2008; Ingaramo *et al.* 2009; Popova and Khodyrev 2011) that focused on specific characteristics of the corpora such as document lengths and its closeness respect to the topics considered in these documents. However, other features such as the number of groups and number of documents per group were maintained the same for all corpora in order to obtain comparable results. Although these studies were limited in general to small size corpora, this decision allowed a meticulous analysis of the features of each collection used in the experiments and a detailed understanding of the results obtained in each case that would be difficult to achieve otherwise with larger standard corpora. Three of these corpora (EasyAbstracts, SEPLN-CICLing and CICLing-2002-A) correspond to short-length documents (scientific abstracts) that mainly differ in the closeness among the topics of their categories. Thus, the EasyAbstracts corpus

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

with scientific abstracts on well-differentiated topics can be considered a medium complexity corpus, whereas the CICling-2002-A corpus with narrow domain abstracts is a relatively high complexity corpus. This corpus, generated with abstracts of articles presented at the *CICLing 2002* conference is a well-known short-text corpus that has been recognized in different works (Makagonov, Alexandrov and Gelbukh 2004; Alexandrov *et al.* 2005; Pinto, Benedí and Rosso 2007; Cagnina *et al.* 2008; Errecalde *et al.* 2008; Ingaramo *et al.* 2008, 2009; Popova and Khodyrev 2011) as a very difficult corpus. The remaining two small corpora, Micro4News and CICLing-2002-F, have longer documents than the previous ones, but Micro4News contains documents about well-differentiated topics (low complexity) whereas CICLing-2002-F and CICLing-2002-A have very related categories (high complexity).

The next five corpora (R4, R6, R8-, R8+, R8B) are subsets of the well-known R8-Test corpus, a subcollection of the Reuters-21578 dataset. These corpora were artificially generated in order to consider corpora with a different number of groups: four groups for R4, six groups for R6 and eight groups for R8B, R8- and R8+. R8B maintains the same groups as R8-Test but the number of documents per group does not significantly differ and it is more balanced. The last two eight-groups corpora differ in the lengths of their documents: R8- contains the shortest documents of R8-Test (approximately a twenty per cent of the documents in each group) whereas R8+ contains the largest documents of the same collection.

JRC6 refers to a sub-collection of JRC-Acquis (Steinberger *et al.* 2006), a popular corpus with legal documents and laws corresponding to different countries of the European Union. JRC6 consists of six groups of the original JRC-Acquis's about well-differentiated topics.

Finally, the last three corpora were used to test the performance of the algorithms in order to study their capabilities when dealing with larger amount of documents. The complete versions of R8-Test and R8-Train corpora were considered in this work. Also, a larger version of JRC6 corpus named JRC-Full containing a larger amount of short documents (in fact, all the short texts of six categories) was considered.[4]

As it can be observed from the previous description, most of the corpora used in the experimental work correspond to corpora with *short* documents. In fact, we are particularly interested in this kind of corpora because they constitute the most challenging scenario for text clustering as it has been recognized in different works (Makagonov *et al.* 2004; Alexandrov *et al.* 2005; Banerjee, Ramanathan and Gupta 2007; He *et al.* 2007; Pinto *et al.* 2007; Carullo, Binaghi and Gallo 2009; Hu *et al.* 2009; Cagnina *et al.* 2008; Errecalde *et al.* 2008; Ingaramo *et al.* 2008, 2009). However, in order to study if the results obtained with *Sil–Att* can be generalized to other more traditional corpora (with longer documents), other corpora such as Micro4News, CICLing-2002-F and JRC6 with lengths between 1,600 and 2,600 terms per document (in average) were also considered.

[4] These corpora can be accessed for research purposes at: https://sites.google.com/site/lcagnina/research.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

The documents were represented with the standard (normalized) *tf-idf* codification after a *stop-word* removing process. The popular *cosine measure* was used to estimate the similarity between two documents.

The results of *Sil–Att* were compared with those obtained by other five clustering algorithms: *K*-means (MacQueen 1967), *K*-MajorClust (Stein and Meyer zu Eissen 2002; Ingaramo *et al.* 2010a), Chameleon (Karypis, Han and Kumar 1999), CLUDIPSO (Cagnina *et al.* 2008) and sIB (Slonim, Friedman and Tishby 2002). These algorithms have been used in similar studies and are representative of different algorithmic principles of clustering. *K*-means, is a classical exemplar-based *iterative* algorithm and is probably one of the most popular clustering algorithms. MajorClust can be classified as a *density-based* algorithm with a cumulative attraction approach (Stein and Meyer zu Eissen 2002; Stein and Busch 2005). *K*-MajorClust (Ingaramo *et al.* 2010a), the variant used in this work, is based on the MajorClust algorithm but it was modified to generate exactly *K* groups. This modification allowed to make its results comparable to those obtained by the other algorithms which always generate clusterings with *K* groups. Chameleon is a *graph-based* clustering algorithm that is considered a good alternative when the clusters have different shapes, sizes and density (Tan *et al.* 2005). CLUDIPSO, a discrete Particle Swarm Optimization algorithm, is a *meta-search* algorithm which explicitly attempts to optimize a global criterion (objective function) that estimates the quality of clusterings. Previous works (Ingaramo *et al.* 2009; Cagnina *et al.* 2014) have showed the potential of CLUDIPSO when the GS coefficient was used as function to optimize, obtaining the best results in experiments with the four small size short-text corpora described at the beginning of this section: Micro4News, EasyAbstracts, SEPLN-CICLing and CICling-2002-A. The parameter settings for CLUDIPSO and the algorithms previously cited used in the comparisons correspond to the parameters empirically derived in Ingaramo *et al.* (2009). The *sequential* clustering algorithm sIB (Slonim *et al.* 2002) is based on the *Information Bottleneck* method. This popular approach finds clusters such that the mutual information between documents and clusters is maximized in a sequential update process as *K*-means does. The parameter setting for sIB corresponds to that proposed in Slonim *et al.* (2002) and selecting the adequate *k* value depending on the amount of the groups of each collection. Regarding the computational complexity of the algorithms evaluated in the experimental study, *K*-means is lineal in the amount of documents, that is $\mathcal{O}(n)$ (Manning, Raghavan and Schutze 2008) and $\mathcal{O}(n^2)$ for CHAMELEON (Karypis *et al.* 1999), CLUDIPSO (Cagnina *et al.* 2014) and *Sil–Att*. *K*-MajorClust is based on MajorClust algorithm but uses exactly *K* clusters. As the computational complexity of MajorClust (Karthikeyan, Peter and Chidambaranathan 2011) does not depend on the number of the groups, we conclude that the latter two have similar computational complexity, that is, $\mathcal{O}(n^2)$. Comparing all computational complexities, we conclude that all algorithms excepts *K*-means (the lowest) have similar computational complexity.

The quality of the results was evaluated by using the classical (external) *F*-measure on the clusterings that each algorithm generated in fifty independent runs

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

per collection.[5] The reported results correspond to the minimum (F_{min}), maximum (F_{max}) and average (F_{avg}) FM values. All the comparisons between *Sil–Att* and the other algorithms used in the experiments were carried out on the basis of statistical significance criteria. We first analyze in Section 4.1 the performance of *Sil–Att* as boosting method. Then, in Section 4.2, the results obtained by *Sil–Att* as an independent clustering algorithm are analyzed.

4.1 Sil–Att as a boosting method

In the experimental work, the first analyzed aspect was the *Sil–Att*'s performance as boosting method. Considering the *Sil–Att*() algorithm shown in Figure 2, that means to compare the result (clustering) $\mathscr{C}_{Alg}$ obtained with some arbitrary clustering algorithm *Alg*, and the result $\mathscr{C}_{Sil-Att(\mathscr{C}_{Alg})}$ obtained with *Sil–Att*() when $\mathscr{C}_{Alg}$ was used as input clustering. Two instances of the *Sil–Att*() algorithm were considered in the experiments with respect to the *stop_condition*() function: (i) $Sil\text{–}Att(\mathscr{C}_{Alg})^1$, where *stop_condition*() always returns *True* and, in consequence, *Sil–Att*() executes only one boosting processing step on the clustering $\mathscr{C}_{Alg}$ received from *Alg*; and (ii) $Sil\text{–}Att(\mathscr{C}_{Alg})^\star$ which iteratively performs an arbitrary number of boosting processing steps, until *stop_condition*() returns *True*; in this work, it occurs when no difference is observed between the clustering obtained in the current iteration and the obtained in the previous one ($\mathscr{C}^\star$ and $\mathscr{C}$ of Figure 2).

In order to keep notation as simple as possible when comparing an arbitrary method *Alg* and the improvements obtained by *Sil–Att*(), the results of $Sil\text{–}Att(\mathscr{C}_{Alg})^1$ will be directly referred as Alg^1, and those obtained with $Sil\text{–}Att(\mathscr{C}_{Alg})^\star$ as $Alg^\star$. Thus, for example, when *K*-means is the method to be compared, the results obtained with $Sil\text{–}Att(\mathscr{C}_{K-\text{means}})^1$ will be denoted as $K\text{-means}^1$, and those obtained with $Sil\text{–}Att(\mathscr{C}_{K-\text{means}})^\star$ as $K\text{-means}^\star$.

Tables 2–5 show the F_{min}, F_{max} and F_{avg} values that *K*-means, *K*-MajorClust, Chameleon and CLUDIPSO obtained with the first eleven corpora.[6] These tables also include the results obtained with $Sil\text{–}Att()^1$ and $Sil\text{–}Att()^\star$ taking as input the groupings generated by these algorithms. The values highlighted in bold indicate the best obtained results.

These results show the good performance that *Sil–Att* can obtain with text collections with very different characteristics. With the exception of the F_{avg} and F_{min} values obtained with CICling2002-F and the F_{max} value of JRC6, it achieves the highest F_{min}, F_{avg} and F_{max} values for all the corpora considered in our study, by improving the clusterings obtained with different algorithms. Thus, for instance, $Sil\text{–}Att()^\star$ obtains the highest F_{avg} value for R8+ by improving the clusterings

[5] The algorithms were executed on a Intel(R) Core (TM)2 Quad CPU 2.83 GHz 3 GB RAM.

[6] In this section, we do not include the results with the last three collections because Chameleon and CLUDIPSO were not able to obtain groupings (to be improved for *Sil–Att*()) because of the lack of RAM memory to complete the process. In order to make a fair comparison, we do not show the results for these larger corpora. sIB algorithm was not considered in this section because with the implementation used in the experimental study is not possible to obtain the groupings to be used by *Sil–Att*().

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Table 2. *Sil–Att()*[1] *and Sil–Att()** *as boosting method: F-measure values for Micro4News, EasyAbstracts and SEPLN-CICLing corpora*

Algorithms	Micro4News			EasyAbstracts			SEPLN-CICLing		
	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}
K-means	0.67	0.41	0.96	0.54	0.31	0.71	0.49	0.36	0.69
K-means[1]	0.84	0.67	**1**	0.76	0.46	0.96	0.63	0.44	0.83
K-means*	0.9	0.7	**1**	0.94	0.71	**1**	0.73	0.65	0.83
K-MajorClust	0.95	0.94	0.96	0.71	0.48	0.98	0.63	0.52	0.75
K-MajorClust[1]	0.97	0.96	**1**	0.82	0.71	0.98	0.68	0.61	0.83
K-MajorClust*	**0.98**	**0.97**	**1**	0.92	0.81	**1**	0.72	**0.71**	**0.88**
Chameleon	0.76	0.46	0.96	0.74	0.39	0.96	0.64	0.4	0.76
Chameleon[1]	0.85	0.71	0.96	0.91	0.62	0.98	0.69	0.53	0.77
Chameleon*	0.93	0.74	0.96	0.92	0.67	0.98	0.69	0.56	0.79
CLUDIPSO	0.93	0.85	**1**	0.92	0.85	0.98	0.72	0.58	0.85
CLUDIPSO[1]	0.96	0.88	**1**	**0.96**	0.92	0.98	**0.75**	0.63	0.85
CLUDIPSO*	0.96	0.89	**1**	**0.96**	**0.94**	0.98	**0.75**	0.65	0.85

Table 3. *Sil–Att()*[1] *and Sil–Att()** *as boosting method: F-measure values for CICLing2002-F, CICLing2002-A and JRC6 corpora*

Algorithms	CICLing2002-F			CICLing2002-A			JRC6		
	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}
K-means	0.48	0.36	0.65	0.45	0.35	0.6	0.52	0.4	**0.64**
K-means[1]	0.58	0.42	0.67	0.54	0.41	0.7	0.53	0.42	0.6
K-means*	0.63	0.52	0.71	0.6	0.47	0.73	0.54	0.45	0.59
K-MajorClust	0.51	0.48	0.6	0.39	0.36	0.48	0.44	0.33	0.55
K-MajorClust[1]	0.6	0.45	0.69	0.48	0.41	0.57	0.47	0.38	0.56
K-MajorClust*	0.64	0.5	0.69	0.61	0.46	0.73	0.51	0.48	0.54
Chameleon	0.48	0.4	0.6	0.46	0.38	0.52	0.38	0.31	0.56
Chameleon[1]	0.57	0.39	0.66	0.51	0.42	0.62	0.52	0.46	0.63
Chameleon*	0.62	0.49	0.69	0.59	0.48	0.71	0.54	**0.47**	0.63
CLUDIPSO	**0.67**	**0.66**	0.71	0.6	0.47	0.73	0.3	0.26	0.33
CLUDIPSO[1]	0.64	0.57	**0.74**	0.61	0.47	**0.75**	0.52	0.39	0.58
CLUDIPSO*	0.65	0.6	0.71	**0.63**	**0.51**	0.7	**0.55**	0.46	0.59

obtained by CLUDIPSO and the highest F_{max} value for EasyAbstracts by improving the clusterings obtained by K-means and K-MajorClust. It can also be appreciated in these tables that both versions of the boosting algorithm, *Sil–Att()*[1] and *Sil–Att()**, obtain in most of the cases considerable improvements on the original clusterings. Thus, for example, the F_{avg} value corresponding to the clusterings generated by K-means with the Micro4News collection ($F_{avg} = 0.67$), is considerably improved by *Sil–Att()*[1] ($F_{avg} = 0.84$) and *Sil–Att()** ($F_{avg} = 0.9$).

Another important aspect that can be analyzed in the previous results is the performance comparison between the iterative approach of *Sil–Att* (*Sil–Att()**) and the approach with only one execution of the algorithm (*Sil–Att()*[1]). Here, the benefits

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Table 4. *Sil–Att()*[1] *and Sil–Att()*[⋆] *as boosting method: F-measure values for* R8B, R8- *and* R8+ *corpora*

Algorithms	R8B			R8-			R8+		
	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}
K-means	0.63	0.47	0.76	0.64	0.55	0.72	0.60	0.46	0.72
K-means[1]	0.7	0.54	0.8	0.67	0.52	0.78	0.65	0.56	0.73
K-means[⋆]	**0.74**	**0.62**	0.85	0.68	0.61	0.78	0.65	0.56	0.73
K-MajorClust	0.23	0.2	0.25	0.61	0.49	0.7	0.57	0.45	0.69
K-MajorClust[1]	0.64	0.48	0.75	0.61	0.5	0.71	0.63	0.55	0.72
K-MajorClust[⋆]	0.71	0.59	**0.86**	**0.7**	0.58	0.75	0.64	0.57	0.68
Chameleon	0.54	0.29	0.71	0.57	0.41	0.75	0.48	0.4	0.6
Chameleon[1]	0.66	0.47	0.8	0.67	0.6	0.77	0.61	0.55	0.67
Chameleon[⋆]	0.7	0.52	0.81	0.68	**0.63**	0.75	0.65	0.6	0.69
CLUDIPSO	0.21	0.18	0.25	0.62	0.49	0.72	0.57	0.45	0.65
CLUDIPSO[1]	0.52	0.41	0.67	0.69	0.54	0.79	0.66	0.57	0.72
CLUDIPSO[⋆]	0.72	0.56	**0.86**	**0.7**	**0.63**	**0.8**	**0.68**	**0.63**	**0.74**

Table 5. *Sil–Att()*[1] *and Sil–Att()*[⋆] *as boosting method: F-measure values for* R4 *and* R6 *corpora*

Algorithms	R4			R6		
	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}
K-means	0.73	0.57	0.91	0.63	0.51	0.81
K-means[1]	0.77	0.58	**0.95**	0.68	0.56	0.83
K-means[⋆]	**0.78**	0.6	**0.95**	**0.71**	**0.58**	0.84
K-MajorClust	0.70	0.45	0.79	0.53	0.36	0.74
K-MajorClust[1]	0.7	0.46	0.84	0.64	0.51	0.73
K-MajorClust[⋆]	**0.78**	**0.7**	0.94	0.69	0.54	0.82
Chameleon	0.61	0.47	0.83	0.52	0.42	0.66
Chameleon[1]	0.69	0.6	0.87	0.59	0.44	0.74
Chameleon[⋆]	0.76	0.65	0.89	0.63	0.47	0.84
CLUDIPSO	0.64	0.48	0.75	0.31	0.26	0.38
CLUDIPSO[1]	0.71	0.53	0.85	0.53	0.4	0.69
CLUDIPSO[⋆]	0.74	0.53	0.89	0.68	0.57	**0.85**

of using the iterative approach seem to be evident, with a better performance of *Sil–Att()*[⋆] on *Sil–Att()*[1] in most of the considered experimental instances. As an example, when *Sil–Att()*[⋆] took as input the clusterings generated by *K*-means, its results were in most of the cases consistently better than those obtained by *Sil–Att()*[1] with the same clusterings, on the eleven considered corpora. The only exception is the F_{max} value of *Sil–Att()*[⋆] with JRC6 (0.59) which is worse than the F_{max} value obtained by *Sil–Att()*[1] (0.6) and even than the one obtained by *K*-means (0.64). However, it is worth noting that, despite the fact that *Sil–Att()*[1]'s results are in general not as good as those obtained by *Sil–Att()*[⋆], the differences between both algorithms are very small and *Sil–Att()*[1] also shows a competitive performance with

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

respect to the other four algorithms. In fact, if $Sil–Att()^{\star}$ were kept out of our analysis, $Sil–Att()^{1}$ would become the algorithm with the highest F_{min}, F_{avg} and F_{max} values in all the experimental instances, except in the same three cases previously mentioned where neither $Sil–Att()^{\star}$ nor $Sil–Att()^{1}$ can outperform the remaining algorithms.

The previous discussion about the very good performance of $Sil–Att()^{1}$ is not a secondary aspect. It offers evidence that the combination of the silhouette coefficient and the attraction concept can be a powerful tool that, in only one step, can achieve significant improvements on the original clusterings. Thus, it can be seen that it is possible to use *Sil–Att* as a relatively inexpensive mechanism that can obtain very competitive results in only one improvement step. However, if higher quality results are required, it can keep iterating with high chances of improving the obtained performance. In the rest of this sub-section, we will only focus on the results obtained with $Sil–Att()^{\star}$ in the comparisons with the other algorithms. However, a similar study could also be carried out using $Sil–Att()^{1}$.

In the previously shown results it could be seen that, despite the excellent performance of *Sil–Att* in almost all experimental instances, there are some few cases where $Sil–Att()^{\star}$ does not improve (and it can even slightly deteriorate) the results obtained with $Sil–Att()^{1}$ or the initial clusterings generated by the other algorithms. This suggests that, although $Sil–Att()^{\star}$ can achieve significant improvements (on *average*) on all the considered algorithms, and the highest F_{min} and F_{max} value obtained on most of the corpora, a deeper analysis is required in order to also consider the particular *improvements* (or the *deteriorations*) that this algorithm makes on each clustering that it receives as input. In other words, it would be interesting to analyze *how often* (and in *what extent*) we can expect to observe an improvement in the quality of the clusterings provided to $Sil–Att()^{\star}$. Tables 6 and 7 give some insights on this issue, by presenting in Table 6 the *improvement percentage* (IP) and the *improvement magnitude* (IM) obtained with $Sil–Att()^{\star}$, whereas Table 7 gives the *deterioration percentage* (DP) and the *deterioration magnitude* (DM) that $Sil–Att()^{\star}$ produced on the original clusterings. The *percentage* of cases where $Sil–Att()^{\star}$ produced clusterings with the *same quality* as the clusterings received as input (SQP) can be directly derived from the two previous percentages. Thus, for example, in Table 6 it can be seen that $Sil–Att()^{\star}$ produced an improvement in the ninety-two per cent of the cases when received the clusterings generated by Chameleon on the R4 collection, giving FM values which are (on average) a 0.17 higher than the FM values obtained with Chameleon. In this case, the values presented in Table 7 indicate that $DP = 7\%$ and $DM = 0.02$ meaning that in one per cent of the experiments with this algorithm and this collection, $Sil–Att()^{\star}$ gave results of the same quality ($SQP = 1\%$).

With the exception of the Chameleon-SEPLN-CICLing and CLUDIPSO-SEPLN-CICLing combinations, where $Sil–Att()^{\star}$ does not obtain significant improvements, the remaining experimental instances show the advantages of using $Sil–Att()^{\star}$ as a general boosting method. Thus, for example, on a total of forty-four experimental instances (algorithm-collection combinations), $Sil–Att()^{\star}$ obtained over eighty-two per cent of improvements in thirty-eight experimental instances and hundred per

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Table 6. *IP and IM values of Sil–Att()⋆ with respect to the original algorithms*

	K-means		*K*-MajorClust		Chameleon		CLUDIPSO	
Corpora /Algorithms	*IP*	*IM*	*IP*	*IM*	*IP*	*IM*	*IP*	*IM*
Micro4News	100	0.3	100	0.48	100	0.18	100	0.3
EasyAbstracts	100	0.44	100	0.49	83	0.15	100	0.05
SEPLN-CICLing	100	0.27	100	0.32	50	0.16	55	0.03
CICLing-2002-A	100	0.18	100	0.22	92	0.13	62	0.04
CICLing-2002-F	100	0.16	100	0.2	97	0.13	66	0.03
R4	96	0.06	100	0.39	92	0.17	98	0.1
R6	100	0.09	100	0.4	100	0.15	100	0.37
R8-	90	0.06	100	0.29	71	0.14	100	0.39
R8+	88	0.07	100	0.27	100	0.18	100	0.43
R8B	100	0.11	100	0.48	100	0.16	100	0.5
JRC6	70	0.04	86	0.08	100	0.16	100	0.22

Table 7. *DP and DM values of Sil–Att()⋆ with respect to the original algorithms*

	K-means		*K*-MajorClust		Chameleon		CLUDIPSO	
Corpora /Algorithms	*DP*	*DM*	*DP*	*DM*	*DP*	*DM*	*DP*	*DM*
Micro4News	0	0	0	0	0	0	0	0
EasyAbstracts	0	0	0	0	16	0.05	0	0
SEPLN-CICLing	0	0	0	0	50	0.04	44	0.03
CICLing-2002-A	0	0	0	0	7	0.03	38	0.01
CICLing-2002-F	0	0	0	0	2	0.05	34	0.02
R4	4	0.03	0	0	7	0.02	1	0.01
R6	0	0	0	0	0	0	0	0
R8-	10	0.03	0	0	28	0.001	0	0
R8+	12	0.02	0	0	0	0	0	0
R8B	0	0	0	0	0	0	0	0
JRC6	30	0.03	13	0.01	0	0	0	0

cent of improvements in twenty-nine cases. This excellent performance of *Sil–Att()⋆* can be easily appreciated in Figure 3, where the *IP* (white bar), *DP* (black bar) and *SQP* (gray bar) values are compared but considering in this case the improvements/deteriorations obtained in each of the eleven corpora.

From the information shown in Tables 6 and 7 and, in Figure 3, it can be concluded that *Sil–Att()⋆* shows a remarkable performance in corpora such as Micro4News, R6 and R8B, where it obtains hundred per cent of improvements. However, in corpora such as SEPLN-CICLing, CICLing-2002-A, CICLing-2002-F and JRC6 the obtained *DP* values indicate that the performance of *Sil–Att()⋆* with respect to the other algorithms, although improved, needs to be further investigated. In both cases, a more rigorous analysis about the statistical significance of those results is required.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

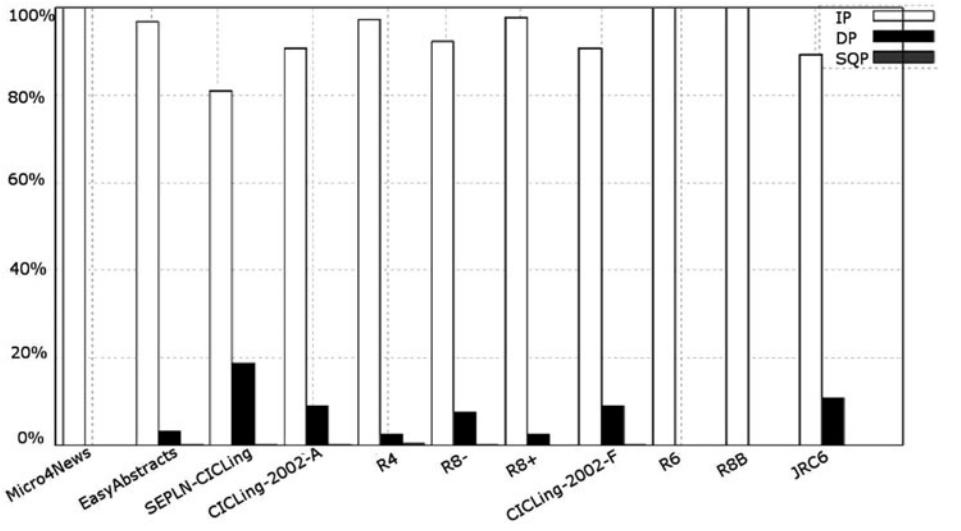


Fig. 3. IP, DP and SQP values per collection of $Sil\text{–}Att()^{\star}$ with respect to the original algorithms.

In this analytical study, the first considered aspect was assessing whether or not the distributional assumptions (*independence* of observations, *normality* of sampling distribution and *equal* variance) required by the analysis of variance ANOVA (Fisher 1925) were violated by the results obtained in the experiments. In this case, the (non-parametric) Levene's test (Levene 1960) showed that the variances obtained in each collection were significantly different and, therefore, the homogeneity of variance assumption was broken in each case.

An alternative approach when the ANOVA's assumptions are violated, is to use a non-parametric approach analogue to ANOVA such as the Kruskal–Wallis test (Kruskal and Wallis 1952). The results obtained with this test allowed to assert that there were significant differences in the results obtained in the eleven considered corpora. Then, the next step was to apply multiple Tukey's tests (Tukey 1953; Barnette and McLean 1998) to determine which were the specific experimental instances where there were significant differences between the results obtained with the original algorithms and those obtained with $Sil\text{–}Att()^{\star}$. However, before starting this analysis, it can be useful to analyze some side-by-side boxplots with the best (Figures 4 and 5) and the worst (Figures 6 and 7) results according to the results previously presented in Tables 6 and 7 and in Figure 3.[7] For each collection and arbitrary algorithm Alg considered in the experiments, the boxplots of $Sil\text{–}Att()^{\star}$ obtained with the initial clusters generated by Alg are shown immediately to the right of the boxplots corresponding to Alg. This allows an easier comparison of $Sil\text{–}Att()^{\star}$ with respect to the clustering algorithms used to generate its initial groupings.

[7] Boxplots (Tukey 1977) are descriptive tools for displaying statistical information such as dispersion, quartiles, median, etc.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

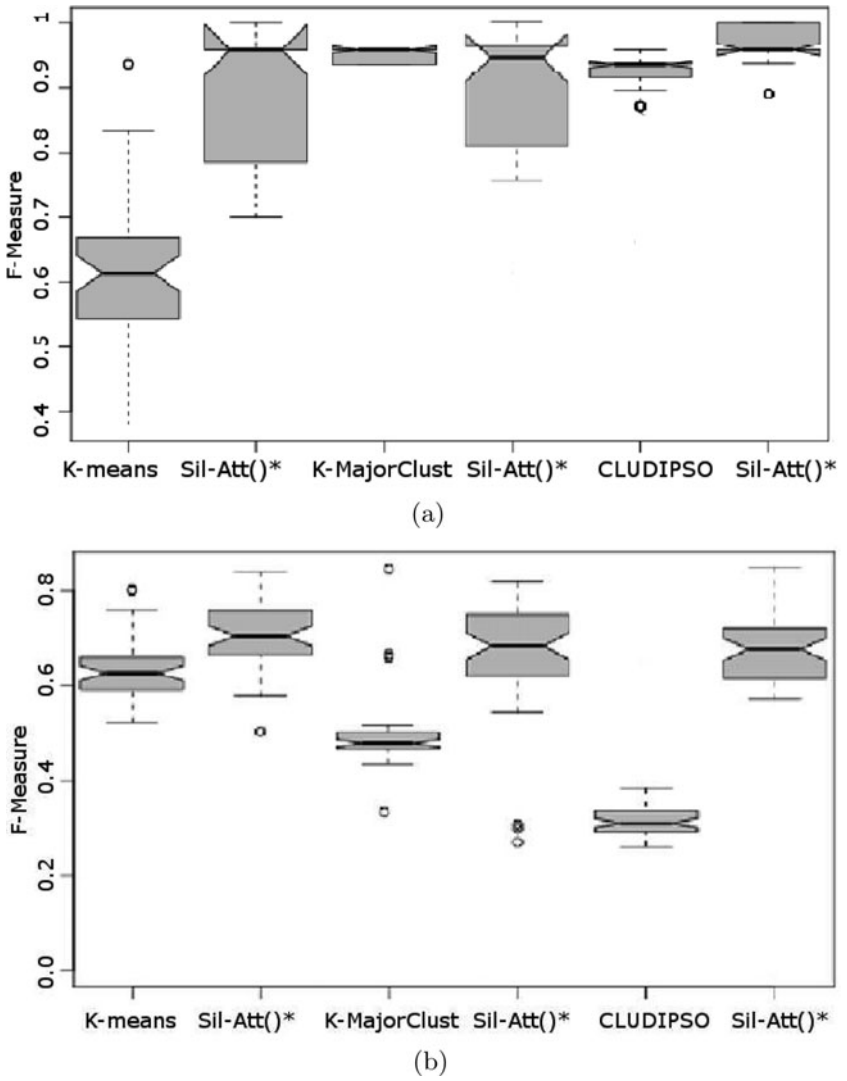


Fig. 4. Best results of *Sil–Att*()* as boosting method. (a) Micro4News. (b) R6.

Our first observation about the boxplots shown in Figures 4 and 5 is that the median values obtained with *Sil–Att*()* are, in all the corpora and algorithms considered, better than the median values of the original algorithms that provided to *Sil–Att*()* its initial clusterings. Moreover, the 'notches' of the boxplots corresponding to *Sil–Att*()* never overlap the notches of the boxplots obtained with the original algorithms. In comparisons of boxplot graphics, that is usually considered as a firm evidence of a significant difference between the data being compared. Another important aspect is related to the upper limits of the *Sil–Att*()*'s boxplots. They all achieve the highest values in all the corpora and algorithms considered showing that *Sil–Att*()* is able to obtain very high-quality results as boosting method, in corpora with very different characteristics. Finally, it is very interesting to note that

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

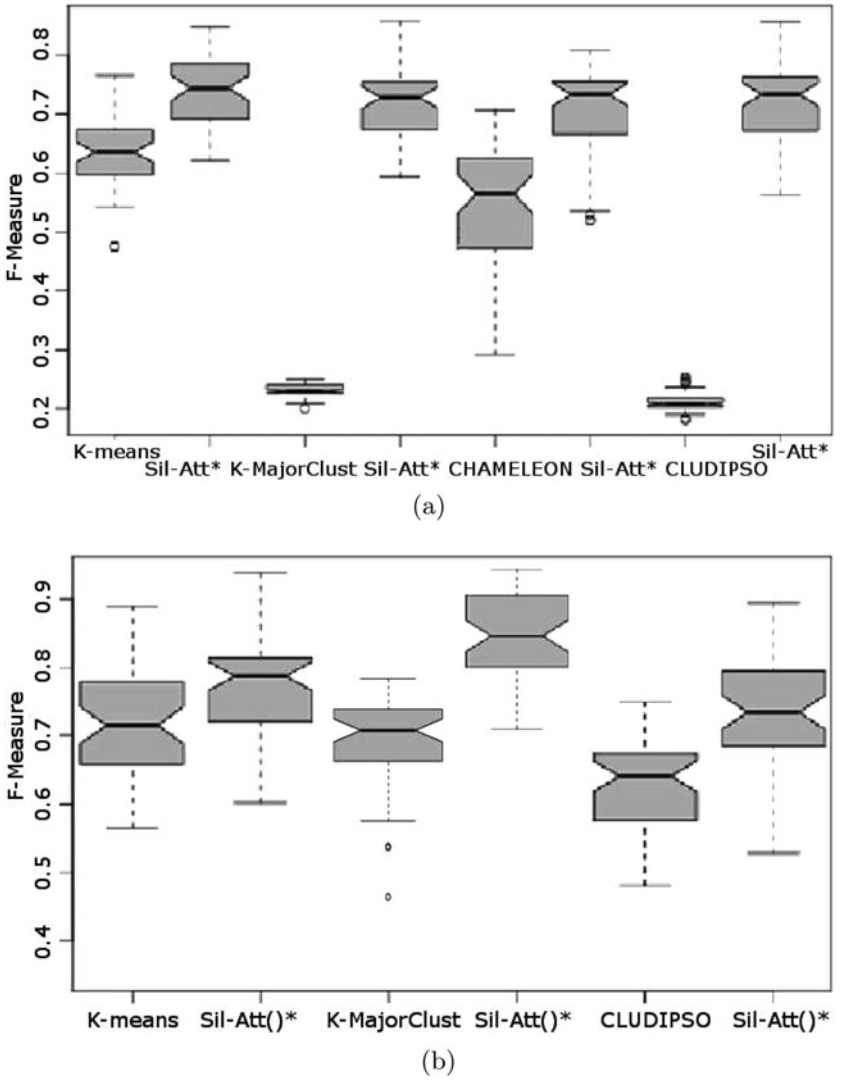


Fig. 5. Best results of *Sil–Att*()* as boosting method. (a) R8B. (b) R4.

in each of the four considered corpora, *Sil–Att*()* obtains very similar boxplots, *independently* of the cluster's quality received as input. This is evident, for instance, in corpora such as R8B where *Sil–Att*()* obtains very similar boxplots with good quality clusters generated with K-means but also with very bad input clusterings generated with K-MajorClust.

With respect to the boxplots shown in Figures 6 and 7, it is interesting to note that the results which were considered as 'the worst' results obtained by *Sil–Att*()*, actually correspond to situations where *Sil–Att*()* does not show a *significant difference* with respect to the original clusterings. This is the case, for example, of the results obtained by CLUDIPSO with SEPLN-CICLing, CICLing-2002-A and CICLing-2002-F, and the K-means' results with JRC6. In all these cases, it can

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

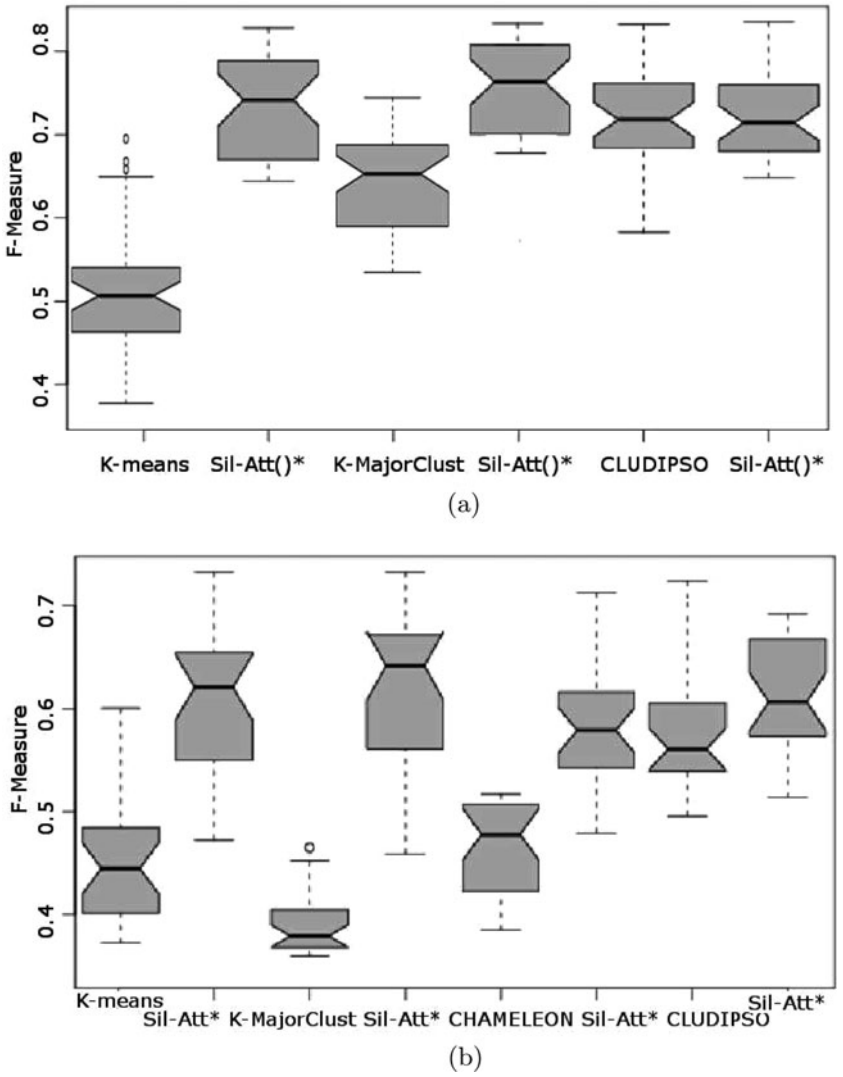


Fig. 6. Worst results of *Sil–Att*()* as boosting method. (a) SEPLN-CICLing. (b) CICLing-2002-A.

be appreciated a slightly better (or similar) performance of *Sil–Att*()*, but it cannot be assured that it clearly outperforms the original algorithms. However, *Sil–Att*()* still shows significant differences in the remaining algorithm-collection combinations and, in some cases, such as the results obtained by CLUDIPSO with JRC6, the advantages of using *Sil–Att*()* are evident.

The above graphical results shown in Figures 4–7 were also confirmed by using the Tukey's test of multiple comparisons which showed significant differences between the performance of *Sil–Att*()* and the original algorithms except for the CLUDIPSO's results with the five small size corpora and the *K*-means' results with JRC6. In order to analyze this aspect, in Table 8 we show only the cases for which there is not significant difference (that is, $p > 0.05$). This is not a minor

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

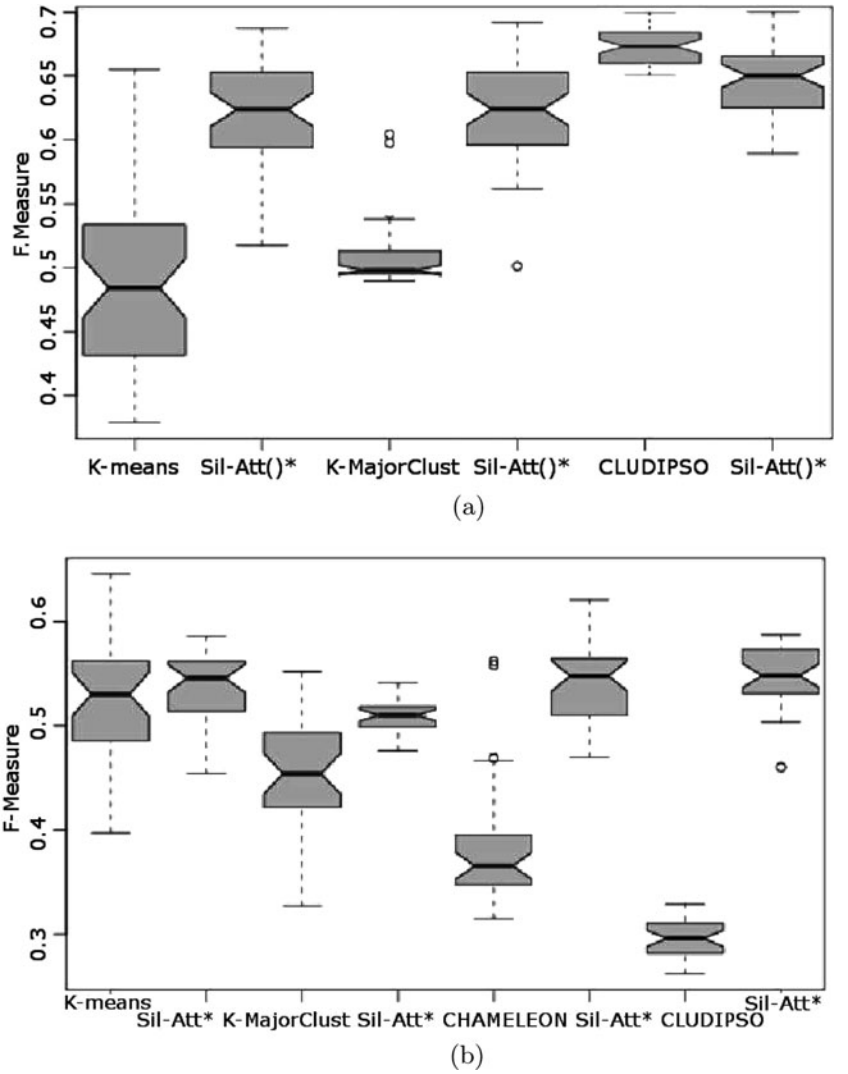


Fig. 7. Worst results of *Sil–Att*()* as boosting method. (a) CICLing-2002-F. (b) JRC6.

aspect, if we consider that on a total of forty-four possible algorithm-collection combinations, *Sil–Att*()* outperformed the remaining algorithms with significant differences in thirty-eight cases and, in the remaining algorithms, it obtained better (or comparable) results but they cannot be considered as statistically significant.

4.2 Sil–Att as an independent clustering method

The two main concepts that support *Sil–Att*, the *silhouette coefficient* information and the idea of *attraction*, were firstly used in two bio-inspired approaches for improving the clusterings obtained by other algorithms in short-text clustering problems (Errecalde *et al.* 2010). Those studies, as well as the results obtained in the previous section, showed some interesting aspects:

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Table 8. *Tukey's tests: (Sil–Att()⋆ versus CLUDIPSO) and (Sil–Att()⋆ vs. K-means). Results with Micro4News, EasyAbstracts, SEPLN-CICLing, CICLing-2002-A,CICLing-2002-F, R4 and JRC6*

	CLUDIPSO	*K*-means
Corpora	*p* value	*p* value
Micro4News	0.6	...
EasyAbstracts	0.1	...
SEPLN-CICLing	1	...
CICLing-2002-A	0.79	...
CICLing-2002-F	0.42	...
JRC6	...	0.4

(1) The approach seems to obtain significant improvements in almost all the considered experimental instances, *independently* of the particular characteristics of the collection being processed. That behavior can be clearly appreciated in Figures 4–7, where *Sil–Att*()⋆ consistently improves (or at least maintains) the quality of the input clusterings, and shows competitive results in *all* the considered corpora.
(2) Those previous results also seem to be *independent* of the clustering algorithms used for generating the input clusterings.

In order to gain a deeper understanding about these aspects, the first addressed issue was analyzing to what extent the quality of the clusterings generated by *Sil–Att* and the quality of the input clusterings are related. In other words, *can we say that the performance of Sil–Att() directly depends on how bad/good the initial clusterings are*?

A simple alternative to attempt answering this question consists in considering some correlation measure such as the *Spearman Correlation* (Spearman 1904) and use it for comparing the quality of the clusterings that *Sil–Att* receives as input and the quality of the clusterings that it generates as output. This was the study carried out in this work, taking as quality measures the GS coefficient and the popular FM. The correlation study considered all the possible combinations using both quality measures on the input/output clusterings. The Spearman correlation values indicate that we are not able to affirm that a correlation between the quality of the input and output clusterings exists (value <0.75).

From the previous results, an obvious question that naturally arises is: *can Sil–Att obtain acceptable quality results taking as input randomly generated clusterings?* That is a crucial aspect because it would become *Sil–Att* a truly independent clustering algorithm which would not require of another clustering algorithm to generate the initial clusterings. This was the idea that gave origin to the $Sil\text{–}Att(NULL)^1$ and the $Sil\text{–}Att(NULL)^\star$ versions of the *Sil–Att*() algorithm presented in Figure 2.

In order to analyze how robust *Sil–Att* is to random initial clusterings, fifty experimental instances of $Sil\text{–}Att(NULL)^1$ and $Sil\text{–}Att(NULL)^\star$ were tested, using as *random_initial_cluster*() function, a simple process that randomly determines the group of each document (denoted Rand-Clust). The results obtained with

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Table 9. *Sil–Att*$(NULL)^1$ *and Sil–Att*$(NULL)^\star$*'s results for Micro4News, EasyAbstracts and SEPLN-CICLing corpora*

Algorithms	Micro4News			EasyAbstracts			SEPLN-CICLing		
	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}
Rand-Clust	0.38	0.31	0.5	0.38	0.32	0.45	0.38	0.3	0.47
Sil–Att$(NULL)^1$	0.87	0.73	**1**	0.76	0.54	0.96	0.63	0.48	0.77
Sil–Att$(NULL)^\star$	0.9	0.73	**1**	**0.92**	0.67	**1**	**0.73**	**0.65**	0.84
K-means	0.67	0.41	0.96	0.54	0.31	0.71	0.49	0.36	0.69
K-MajorClust	**0.95**	**0.94**	0.96	0.71	0.48	0.98	0.63	0.52	0.75
Chameleon	0.76	0.46	0.96	0.74	0.39	0.96	0.64	0.4	0.76
CLUDIPSO	0.93	0.85	**1**	**0.92**	**0.85**	0.98	0.72	0.58	**0.85**
sIB	0.7	0.61	0.72	0.72	0.71	0.75	0.47	0.45	0.54

Table 10. *Sil–Att*$(NULL)^1$ *and Sil–Att*$(NULL)^\star$*'s results for CICLing2002-F, CICLing2002-A and JRC6 corpora*

Algorithms	CICLing2002-F			CICLing2002-A			JRC6		
	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}
Rand-Clust	0.38	0.31	0.46	0.39	0.31	0.52	0.23	0.21	0.25
Sil–Att$(NULL)^1$	0.56	0.43	0.71	0.54	0.42	0.71	0.51	0.4	0.59
Sil–Att$(NULL)^\star$	0.64	0.44	**0.72**	**0.6**	0.46	**0.75**	**0.53**	**0.44**	0.59
K-means	0.48	0.36	0.65	0.45	0.35	0.6	0.52	0.4	**0.64**
K-MajorClust	0.51	0.48	0.6	0.39	0.36	0.48	0.44	0.33	0.55
Chameleon	0.48	0.4	0.6	0.46	0.38	0.52	0.38	0.31	0.56
CLUDIPSO	**0.67**	**0.66**	0.71	**0.6**	**0.47**	0.73	0.3	0.26	0.33
sIB	0.43	0.36	0.49	0.45	0.38	0.51	0.48	0.42	0.51

Rand-Clust were compared with those of the algorithms considered in the previous section. Tables 9–13 show these results. In those, it is possible to appreciate that *Sil–Att*$(NULL)^\star$ is robust despite the low quality of the initial clusterings. In fact, *Sil–Att*$(NULL)^\star$ obtained in most of the considered corpora the best F_{min}, F_{max} or F_{avg} values and, in the remaining cases, it achieved results comparable to the best results obtained with the other algorithms. Thus, for example, in eleven corpora it obtained the best F_{max} value and, in the remaining three corpora, the second best value, with a minimal difference with respect to the best obtained value. In eleven corpora, it obtained the best F_{avg} value and, in nine of the fourteen corpora it obtained the best F_{min} value. This last result seems to be one weak aspect of *Sil–Att* as independent clustering method, due to the low quality clusterings obtained in some cases. However, as can be appreciated in Figures 8–13, the boxplots corresponding to *Sil–Att*$(NULL)^\star$ (abbreviated as *Sil–Att** for short) show a remarkable performance of this method in eleven of the considered corpora.

Following the same procedure that in the previous section, the first considered aspect was assessing whether or not the distributional assumptions required by

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Table 11. *Sil–Att(NULL)*[1] *and Sil–Att(NULL)***'s results for* R8B, R8- *and* R8+ *corpora*

Algorithms	R8B			R8-			R8+		
	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}
Rand-Clust	0.18	0.17	0.19	0.21	0.19	0.24	0.21	0.2	0.24
Sil–Att(NULL)[1]	0.65	0.55	0.79	0.63	0.52	0.73	0.64	0.54	0.7
Sil–Att(NULL)*	**0.71**	**0.63**	**0.86**	**0.66**	**0.57**	0.74	**0.65**	**0.57**	**0.72**
K-means	0.63	0.47	0.76	0.64	0.55	0.72	0.60	0.46	**0.72**
K-MajorClust	0.23	0.2	0.25	0.61	0.49	0.7	0.57	0.45	0.69
Chameleon	0.54	0.29	0.71	0.57	0.41	**0.75**	0.48	0.4	0.6
CLUDIPSO	0.21	0.18	0.25	0.62	0.49	0.72	0.57	0.45	0.65
sIB	0.48	0.45	0.52	0.26	0.23	0.31	0.28	0.27	0.29

Table 12. *Sil–Att(NULL)*[1] *and Sil–Att(NULL)***'s results for* R4 *and* R6 *corpora*

Algorithms	R4			R6		
	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}
Rand-Clust	0.32	0.29	0.35	0.23	0.2	0.25
Sil–Att(NULL)[1]	0.68	0.48	0.87	0.65	0.49	0.77
Sil–Att(NULL)*	**0.75**	0.54	**0.94**	**0.71**	**0.59**	**0.85**
K-means	0.73	**0.57**	0.91	0.63	0.51	0.81
K-MajorClust	0.70	0.45	0.79	0.53	0.36	0.74
Chameleon	0.61	0.47	0.83	0.52	0.42	0.66
CLUDIPSO	0.64	0.48	0.75	0.31	0.26	0.38
sIB	0.54	0.47	0.6	0.56	0.51	0.63

ANOVA were violated. The Levene's test values obtained in this case also showed that in all the considered cases, these values were *significant* and, in consequence, the ANOVA's assumptions were violated. Therefore, the Kruskal–Wallis test was newly used and the results showed that there were significant differences in the results obtained in the eleven considered corpora. Therefore, the next step was to apply multiple Tukey's tests to determine which are the specific experimental instances where these differences hold.

The same as in the previous study, the differences between the results obtained with *Sil–Att(NULL)** and the other algorithms were statistically significant in most of the considered corpora, clearly outperforming them in thirty-eight of the forty-four possible experimental instances. The values of the six cases where the Tukey's test did not reveal significant differences are shown in Table 14. In this table, it can be seen that this occurs when *Sil–Att(NULL)** is compared to CLUDIPSO in the four small corpora (Micro4News, EasyAbstracts, SEPLN-CICLing and CICLing-2002-A) and in two specific experimental instances where *K*-means obtains good results with the R4 and JRC6 corpora. Obviously, that do not mean that in these instances CLUDIPSO

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Table 13. *Sil–Att(NULL)*[1] *and Sil–Att(NULL)***'s results for R8-Test, JRC-Full and R8-Train corpora*

Algorithms	R8-Test			JRC-Full			R8-Train		
	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}	F_{avg}	F_{min}	F_{max}
Rand-Clust	0.19	0.17	0.2	0.20	0.19	0.20	0.18	0.17	0.20
Sil–Att(NULL)[1]	0.72	0.62	0.78	0.68	0.55	0.78	**0.62**	0.55	0.75
Sil–Att(NULL)*	**0.73**	**0.63**	**0.79**	**0.70**	**0.58**	**0.80**	**0.62**	**0.57**	**0.76**
K-means	0.67	0.54	0.71	0.50	0.44	0.56	0.60	0.45	0.74
K-MajorClust	0.53	0.44	0.62	0.47	0.46	0.5	0.55	0.45	0.59
Chameleon	0.56	0.53	0.59	0.47	0.38	0.47	NA[a]	NA	NA
CLUDIPSO	NA	NA	NA	NA	NA	NA	NA	NA	NA
sIB	0.46	0.43	0.51	0.56	0.41	0.71	0.43	0.42	0.44

[a]NA: Not Available result.

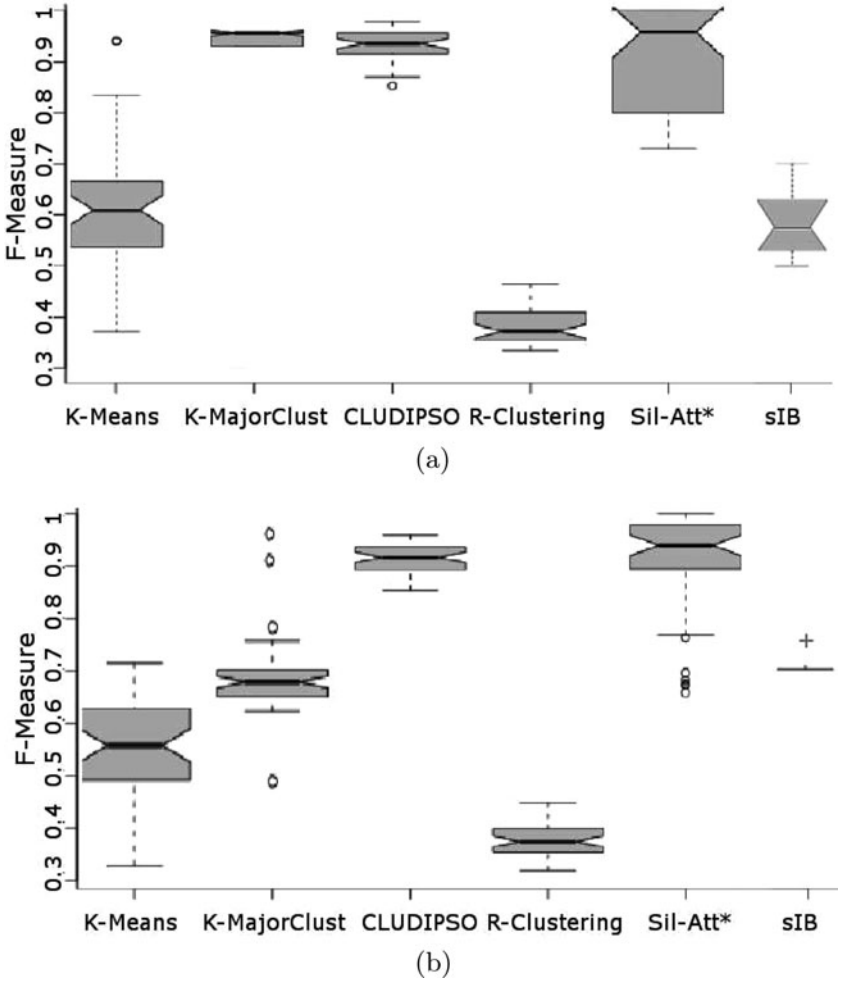


Fig. 8. Sil–Att as independent clustering method. (a) Micro4News. (b) EasyAbstracts

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

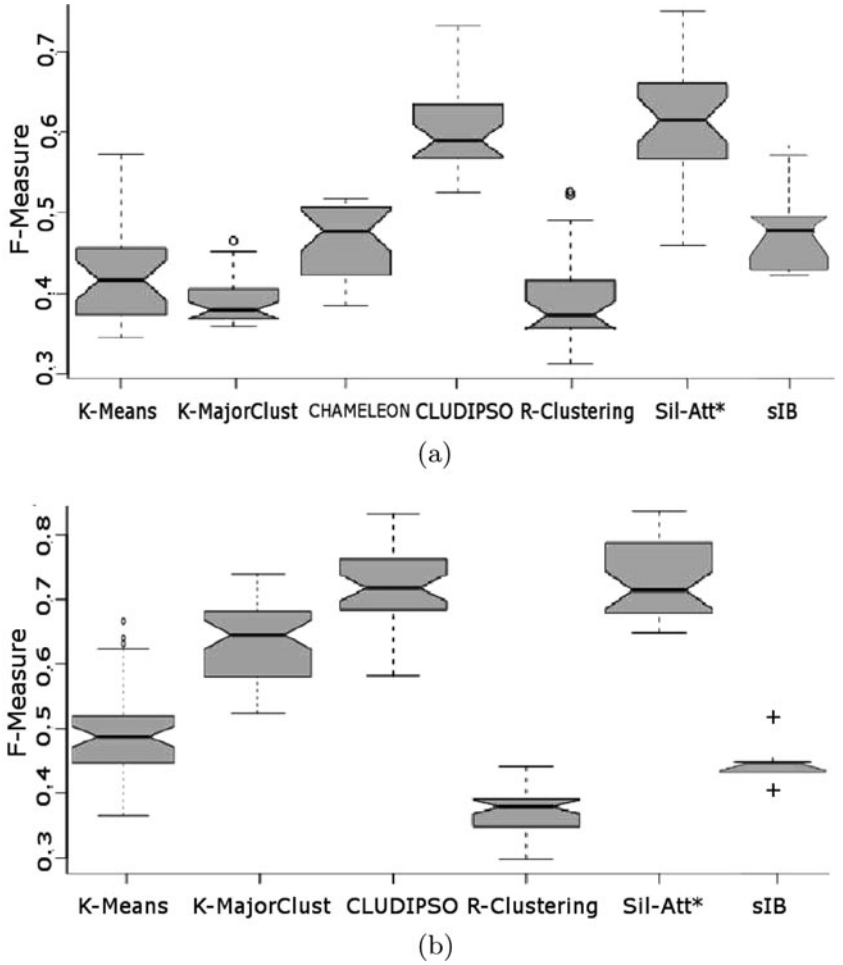


Fig. 9. Sil–Att as independent clustering method. (a) CICLing-2002-A. (b) SEPLN-CICLing.

and K-means outperform $Sil\text{–}Att(NULL)^{\star}$, but not significant differences can be observed in these cases, as can also be seen in the boxplots presented in Figures 8–10.

5 Related work

The use of silhouette coefficient information beyond its role as ICVM has several antecedents in clustering problems. In Errecalde *et al.* (2008) for instance, the evaluation of the GS coefficient and other ICVMs on the 'gold standard' of different short-text corpora is proposed as a method to estimate the hardness of those corpora. The GS coefficient has also been used as an explicit *objective function* that the clustering algorithms attempt to optimize. This idea has recently been used in short-text clustering by using discrete and continuous Particle Swarm Optimization algorithms as function optimizers (Cagnina *et al.* 2008; Ingaramo *et al.* 2009; Cagnina *et al.* 2014). In these works, CLUDIPSO obtained the best results on different short-text corpora when the GS coefficient was used as objective function.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

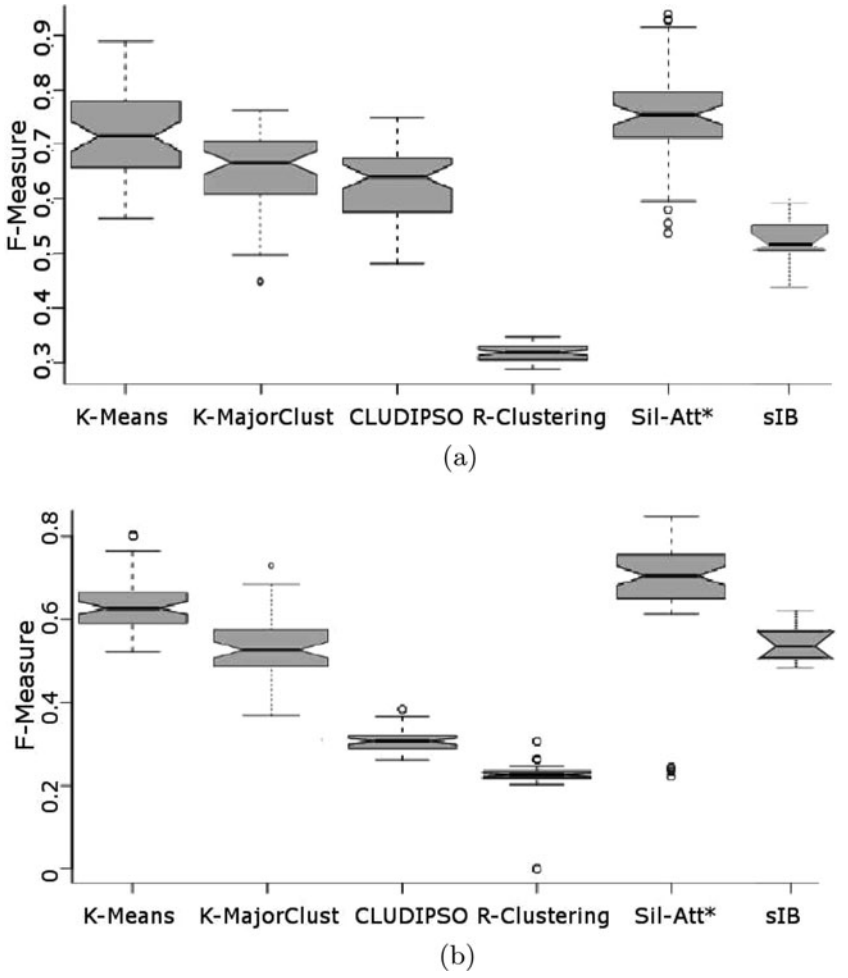


Fig. 10. Sil–Att as independent clustering method. (a) R4. (b) R6.

This coefficient has also been recognized as a good measure to determine the correct number of clusters in arbitrary data sets (Rousseeuw 1987; Tan *et al.* 2005; Choi *et al.* 2011).

In Bonato dos Santos *et al.* (2011), an estimation method to automatically determine a threshold for a similarity measure is proposed. It relies on a clustering phase and on the choice of a similarity threshold based on the silhouette coefficient. This work also includes some experimental studies that show that the silhouette coefficient is highly correlated with the (external) FM. The idea of closeness can be an alternative choice to the similarity measure. In Zhou, Cheng and Yu (2009), a closeness measure is used to perform the clustering of data in networks. The algorithm learns the weights of the edges connecting nodes in the network, during a random walks process and uses them in the clustering process. For the same problem, the algorithm proposed in Qi, Aggarwal and Huang (2012) considers that text content is attached to the edges meanwhile in Yang *et al.* (2009) the content is attached to the nodes of the network. The approach in Qi *et al.* (2012) implements

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

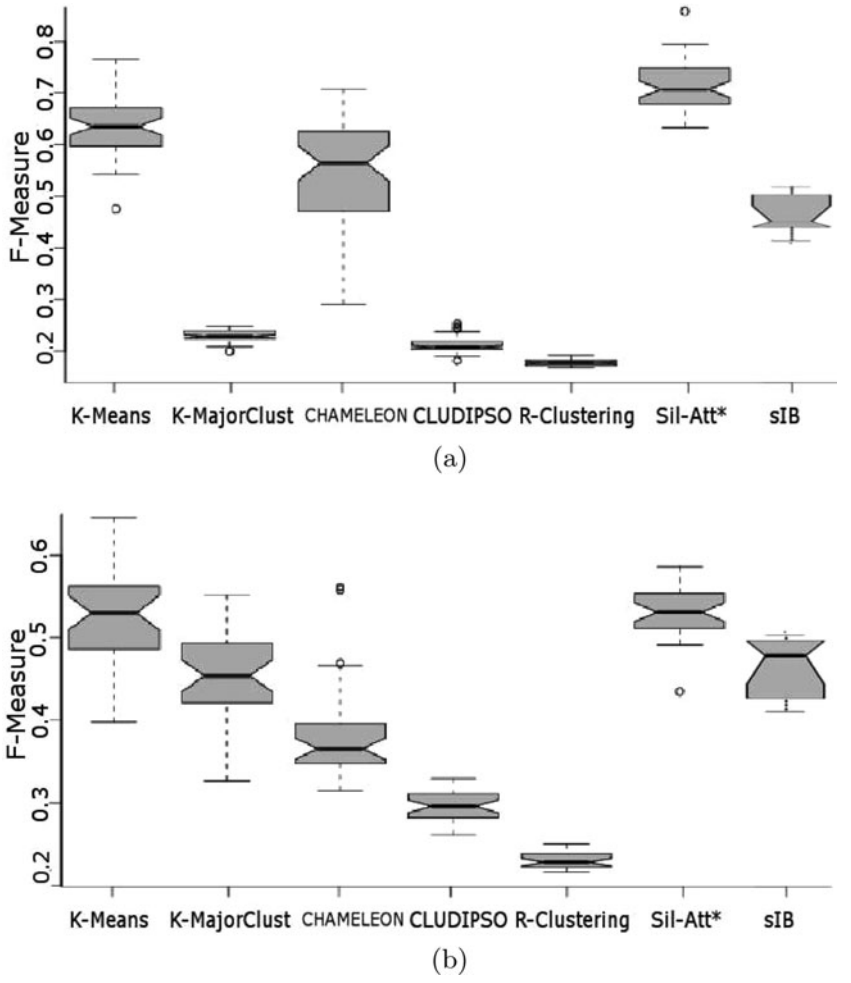


Fig. 11. Sil–Att as independent clustering method. (a) R8B. (b) JRC6.

a matrix-factorization methodology for modeling the content and the structure of a network to be clustered then, by a K-means algorithm. The work proposed in Yang *et al.* (2009) uses a two-stage optimization algorithm which combines conditional and discriminative content models for performing the clustering. In opposition to the idea of similarity, in Zhang, Wang and Si (2011) the authors proposed to use a *universum* of documents not belonging to any class, in order to guide the clustering process. The use of that universum as background information avoids mistakes in the selection of the adequate cluster. In Aranganayagi and Thangavel (2007), a novel algorithm is proposed to cluster categorical data. Objects are grouped into clusters taking into account minimum dissimilarity values. Then, these objects are relocated with a merging process by using the silhouette coefficient.

The idea of *attraction* can have different interpretations but in this work, it corresponds to the concept described in Stein and Meyer zu Eissen (2002); Stein and Busch (2005), where it is used as a key component of an efficient density-based

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

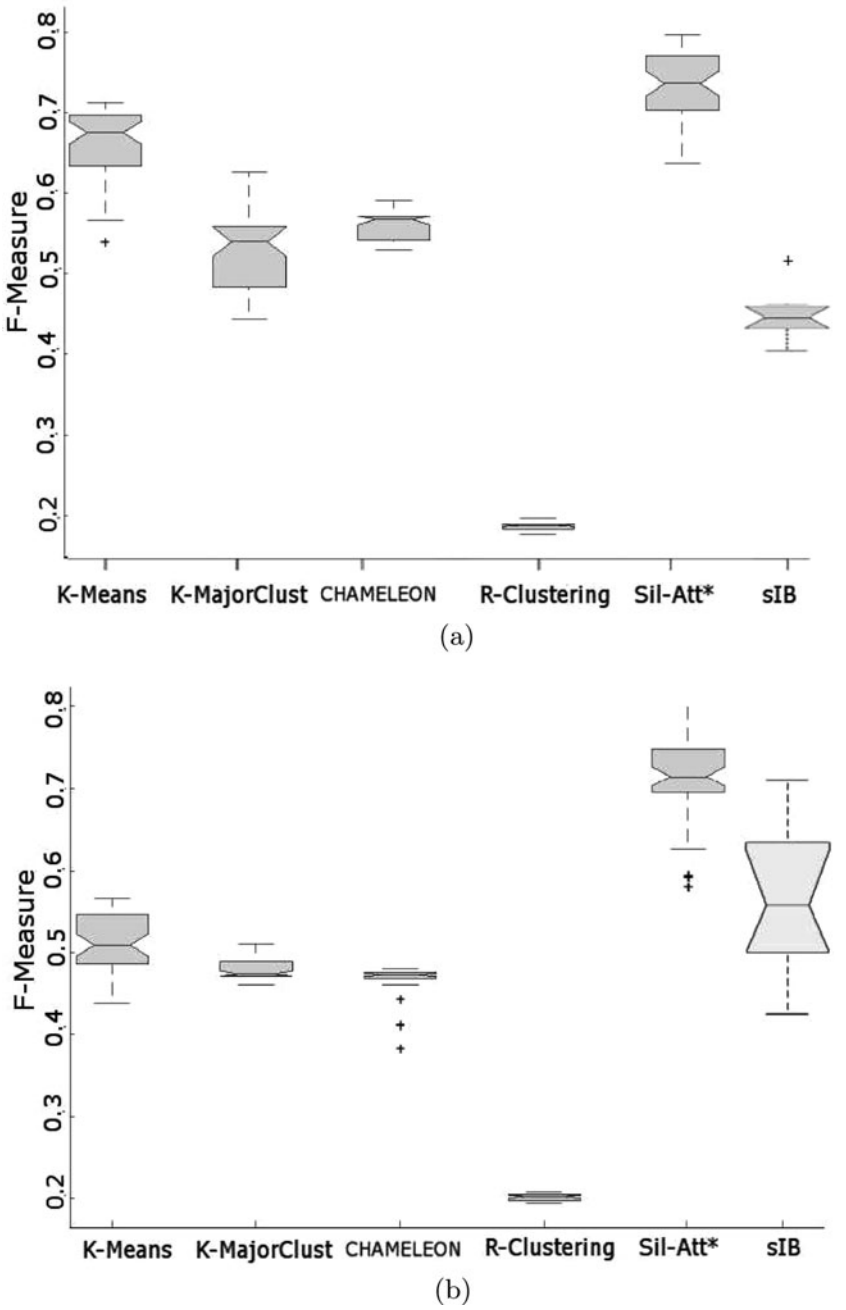


Fig. 12. Sil–Att as independent clustering method. (a) R8-Test. (b) JRC-Full.

clustering algorithm (MajorClust). MajorClust derives its density information from the attraction a cluster C exerts on some object q. This attraction is computed as the sum of the similarity values among q and all the objects in C. MajorClust implements density propagation according to the principle 'maximum attraction wins'.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

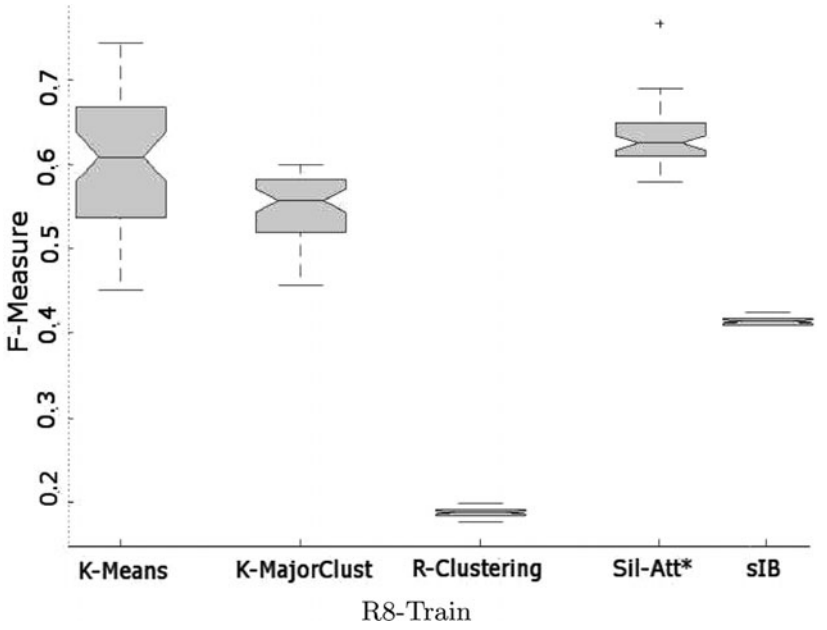


Fig. 13. Sil–Att as independent clustering method.

Table 14. *Tukey's tests: (Sil–Att(NULL)⋆ vs. CLUDIPSO) and (Sil–Att(NULL)⋆ vs. K-means). Results with Micro4News, EasyAbstracts, SEPLN-CICLing, CICLing-2002-A, R4 and JRC6*

	CLUDIPSO	*K*-means
Corpora	*p* value	*p* value
Micro4News	0.6	...
EasyAbstracts	0.9	...
SEPLN-CICLing	0.9	...
CICLing-2002-A	0.9	...
R4	...	0.07
JRC6	...	0.9

Another work which combines the idea of attraction and density is proposed in Jiang, Pei and Zhang (2003). However, it differs from the previous approach (and the one used in *Sil–Att*) in the way the attraction is computed. Here, the attraction between two objects is based on the density that both objects have and the idea is that objects with high density attract some other objects with lower density.

In Tu and Chen (2009), D-Stream, a new density-based framework for clustering stream data is proposed. This framework aims at finding clusters of arbitrary shapes. A key component in D-Stream is an attraction-based mechanism which allows to accurately generate cluster boundaries. The algorithm maps each input data into a grid, computes the density of each grid and clusters the grids by using a density-based algorithm. Then, a new concept on the attraction between grids is used in order to improve the quality of density-based clustering.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Sil–Att initially sorts the documents to be clustered and then iteratively adds these documents into the most appropriate group. This idea was also used in two bio-inspired hierarchical clustering algorithms for clustering arbitrary objects (Azzag *et al.* 2003; Ingaramo, Leguizamón and Errecalde 2005). These works did not consider collections of documents in the experimental work and used a different approach to obtain the initial ordering of the objects to be clustered. Furthermore, the decision on which group an element should be added to, was not made following the attraction idea and only took into account a single ant (object) that played the role of representative of the group.

The first approach in which silhouette coefficient information and the idea of attraction were simultaneously integrated in the same clustering algorithm, is *AntSA-CLU* (Ingaramo *et al.* 2010b). *AntSA-CLU* is a hierarchical AntTree-based algorithm whose main ideas were taken from Azzag *et al.* (2003) but it also incorporated silhouette coefficient information and the idea of attraction as key components in its functioning. AntSA-CLU heavily depended on the initial data partition generated by the CLUDIPSO algorithm. This idea was later generalized and served as antecedent for the approach of using *Sil–Att* as a general improvement method. In Ingaramo *et al.* (2010a), a simplified and more general version of AntSA-CLU is presented, named *Partitional AntSA⋆ (PAntSA⋆)*. PAntSA⋆ is the *partitional* version of the *hierarchical* AntSA-CLU algorithm where, furthermore, it is not assumed as input the results of any particular clustering algorithm. In that way, PAntSA⋆ takes the clusterings generated by arbitrary clustering algorithms and attempts to improve them by using techniques based on silhouette coefficient information and the idea of attraction.

Finally, ITSA⋆ (ITerative PAntSA⋆), the iterative version of *PAntSA⋆*, is a bio-inspired method which recently obtained interesting results in short-text clustering problems (Errecalde *et al.* 2010). ITSA⋆ was also used to cluster documents whose representations were enriched with semantic information (concepts) obtained from knowledge-based disambiguation methods (Ingaramo *et al.* 2011). The experimental results offered strong evidence that ITSA⋆ is a robust method which can handle different document representations, obtaining competitive results when semantic information was added to these representations. ITSA⋆ is the direct predecessor of the proposal presented in this work, but *Sil–Att* does not consider any bio-inspired aspects in its functioning in order to keep the proposal as simple and clear as possible. In that way, the present work can be considered as an extension of the ideas proposed in Errecalde *et al.* (2010), presenting a simpler and more efficient algorithm, considering more general corpora and providing a more exhaustive analysis on the statistical significance of the results and the independence of *Sil–Att* on the quality of the initial clusterings.

6 Conclusions

Sil–Att is a three-step clustering algorithm based on silhouette coefficient and attraction where: (i) given an initial clustering, documents of each group are sorted in decreasing order accordingly to the silhouette coefficient and the document with

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

the highest silhouette coefficient is selected as representative of the group; (ii) the remaining documents of each sorted group are merged in a single list; (iii) each document of the list is assigned to the group that exerts the highest attraction. The obtained result can eventually be used as an input clustering for a further iteration of the algorithm.

The combination of both silhouette and attraction allows to obtain a general technique that can be used as a boosting method, which improves results of other clustering algorithms, or as an independent clustering algorithm. Experiments were carried out on fourteen corpora with different levels of complexity with respect to the size, length of documents and vocabulary overlapping showing that *Sil–Att* is a very robust algorithm that obtains very good performance no matter the very different characteristics of text corpora. This is corroborated by the exhaustive analysis we did on the statistical significance of the results and the independence of the quality of the initial clusters is executed initially with.

In the present work, we focused on problems and algorithms that assume that the right number of clusters (k) is provided beforehand. However, as mentioned in Sections 3 and 5, the silhouette coefficient could also be used for determining the optimal number of groups (Rousseeuw 1987; Tan *et al.* 2005; Choi *et al.* 2011). In that way, we might use this measure in a pre-processing stage that estimates the correct number of clusters and then, apply the *Sil–Att* algorithm in the same way as we did in the present article. This would open a new research line where *Sil–Att* could also be compared against other algorithms that automatically determine the number of clusters in the result. These ideas could be applied to solve web applications as clustering of web pages, web images, wiki entities and web users.

Although we have shown that the *silhouette + attraction* combination is an appealing and effective idea for document clustering, this is only the beginning of different research lines that will be addressed in future works. For instance, we used the 'standard' silhouette coefficient but other techniques that only consider one of the components that capture cohesion/centrality and separation/discrimination (components $a(i)$ and $b(i)$ respectively in the $s(i)$ formula) could also be used. We plan to propose an *adjustable* silhouette coefficient that allows to attach different weights to these components and to carry out a detailed analysis of the performance of these approaches against the 'standard' silhouette coefficient used in the present work.

The attraction measure is another aspect in which different improvements could be achieved. In the present article, we just tested the attraction measure introduced in (1) but other more effective and efficient approaches could be obtained. For instance, an improvement in efficiency could be obtained if a *centroid*-based attraction is implemented. This idea would require that each time a document is added to a group, the corresponding centroid of the group is updated. On the other hand, more elaborated and effective attraction measures could also be used like, for instance, the same silhouette coefficient that was used to order the documents in the *Sil–Att* algorithm. Although this approach would have a higher computation cost, it might also be decreased by using a *centroid*-based approach when computing the $b(i)$ component of the silhouette coefficient.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

It is also interesting to notice that the silhouette coefficient is a measure that might be useful to *boost* the remainder algorithms used in the present work. For instance, an interesting option would be use it in the traditional K-means method to select the prototypes that represent each cluster. In that case, this information could be used only as an 'initial seed' to select the initial pivots of K-means or in each iteration step of the algorithm where the centroids need to be computed. An exhaustive comparison of those variants of K-means against the different versions of *Sil–Att* are beyond the scope of this article but it would give new evidence of the relevance of the proposed *silhouette+attraction* combination. This proposal will be addressed in our future works.

An interesting approach could be obtained from the combination of K-means and *Sil–Att* algorithms. Executing iteratively some steps of K-means then some steps of *Sil–Att* we could study the improvements obtained with both algorithms.

Finally, algorithms were evaluated by using the well-known FM; in future works other evaluation metrics will be considered.

References

Alexandrov, M., Gelbukh, A., and Rosso, P. 2005. An approach to clustering abstracts. In A. Montoyo, R. Muñoz, and E. Métais (eds.), *Natural Language Processing and Information Systems*, vol. 3513, pp. 275–85. Lecture Notes in Computer Science. Germany: Springer-Verlag Berlin Heidelberg, .

Aranganayagi, S., and Thangavel, K. 2007. Clustering categorical data using silhouette coefficient as a relocating measure. In *Proceedings of the International Conference on Computational Intelligence and Multimedia Applications (ICCIMA-2007)*, vol. 02, California, USA: IEEE Computer Society, pp. 13–7.

Azzag, H., Monmarche, N., Slimane, M., Venturini, G., and Guinot, C. 2003. AntTree: a new model for clustering with artificial ants. In *Proceedings of the IEEE Conference on Evolutionary Computation (CEC-2003)*, vol. 04, New Jersey, USA: IEEE Press, pp. 2642–7.

Banerjee, S., Ramanathan, K., and Gupta, A. 2007. Clustering short texts using Wikipedia. In *Proceedings of the 30th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR-2007)*, New York, USA: ACM, pp. 787–8.

Barnette, J. J., and McLean, J. E. 1998. The Tukey honestly significant difference procedure and its control of the type I error-rate. In *Proceedings of 27 Annual Meeting of the Mid-South Educational Research Association*, John Petry. Louisiana, USA, pp. 2–15.

Berry, M. W. (ed.) 2003. *Survey of Text Mining: Clustering, Classification, and Retrieval*. New York, USA: Springer Science+Business Media, Inc.

Bezdek, J. C., and Pal, N. R. 1995. Cluster validation with generalized Dunn's indices. In *Proceedings of the Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, California, USA: IEEE Computer Society Press, pp. 190–3.

Bonato dos Santos, J., Heuser, C., Pereira Moreira, V., and Krug Wives, L. 2011. Automatic threshold estimation for data matching applications. *Information Sciences* **181**(13): 2685–99.

Brun, M., Sima, C., Hua, J., Lowey, J., Carroll, B., Suh, E., and Dougherty, E. R. 2007. Model-based evaluation of clustering validation measures. *Pattern Recognition* **40**(3): 807–24.

Cagnina, L., Errecalde, M., Ingaramo, D., and Rosso, P. 2008. A discrete particle swarm optimizer for clustering short-text corpora. In *Proceedings of the International Conference on Bioinspired Optimization Methods and their Applications (BIOMA08)*, Ljubljana, Slovenia: Jozef Stefan Institute, pp. 93–103.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Cagnina, L., Errecalde, M., Ingaramo, D., and Rosso, P. 2014. An efficient particle swarm optimization approach to cluster short texts. *Information Sciences* **265**: 36–49.

Carullo, M., Binaghi, E., and Gallo, I. 2009. An online document clustering technique for short web contents. *Pattern Recognition Letters* **30**(10): 870–6.

Charikar, M., Chekuri, C., Feder, T., and Motwani, R. 2004. Incremental clustering and dynamic information retrieval. *SIAM Journal on Computing* **33**(6): 1417–40.

Choi, M. J., Tan, V. Y. F., Anandkumar, A., and Willsky, A. S. 2011. Learning latent tree graphical models. *Journal of Machine Learning Research* **12**: 1771–812.

Cutting, D. R., Karger, D. R., Pedersen, J. O., and Tukey, J. W. 1992. Scatter/gather: a cluster-based approach to browsing large document collections. In *Proceedings of the 15th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR 2002)*, Interface Design and Display, New York, USA: ACM, pp. 318–29.

Davies, D. L., and Bouldin, D. W. 1979. A cluster separation measure. *IEEE Transactions on Analysis and Machine Intelligence* **1**(2): 224–7.

Dunn, J. C. 1974. Well separated clusters and optimal fuzzy-partitions. *Journal of Cybernetics* **4**(1): 95–104.

Errecalde, M., Ingaramo, D., and Rosso, P. 2008. Proximity estimation and hardness of short-text corpora. In *19th International Workshop on Database and Expert Systems Application*, California, USA: IEEE Computer Society, pp. 15–9.

Errecalde, M., Ingaramo, D., and Rosso, P. 2010. ITSA*: an effective iterative method for short-text clustering tasks. In *Proceedings of the 23rd International Conference on Industrial Engineering and other Applications of Applied Intelligent Systems, IEA/AIE 2010*, vol. 6096, Lecture Notes in Computer Science, Germany: Springer-Verlag Berlin, Heidelberg. , pp. 550–9.

Fisher, R. 1925. *Statistical Methods for Research Workers*. Edinburgh, Scotland: Oliver & Boyd.

Hasan, M. A., Chaoji, V., Salem, S., and Zaki, M. J. 2009. Robust partitional clustering by outlier and density insensitive seeding. *Pattern Recognition Letters* **30**(11): 994–1002.

He, H., Chen, B., Xu, W., and Guo, J. 2007. Short text feature extraction and clustering for web topic mining. In *Proceedings of the 3rd International Conference on Semantics, Knowledge and Grid*, California, USA: IEEE Computer Society, pp. 382–5.

Hearst, M. A. 2006. Clustering versus faceted categories for information exploration. *Communications of the ACM* **49**(4): 59–61.

Hu, X., Sun, N., Zhang, C., and Chua, T. 2009. Exploiting internal and external semantics for the clustering of short texts using world knowledge. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, New York, USA: ACM, pp. 919–28.

Ingaramo, D., Errecalde, M., Cagnina, L., and Rosso, P. 2009. Particle Swarm Optimization for clustering short-text corpora. In *Computational Intelligence and Bioengineering*, pp. 3–19. Amsterdam, The Netherlands: IOS Press.

Ingaramo, D., Errecalde, M., and Rosso, P. 2010a. A general bio-inspired method to improve the short-text clustering task. In *Proceedings of 11th International Conference on Intelligent Text Processing and Computational Linguistics, CICLing 2010*, vol. 6008, pp. 661–72. Lecture Notes in Computer Science. Germany: Springer-Verlag Berlin Heidelberg. .

Ingaramo, D., Errecalde, M., and Rosso, P. 2010b. A new AntTree-based algorithm for clustering short-text corpora. *Journal of Computer Science & Technology* **10**(1): 1–7.

Ingaramo, D., Leguizamón, G., and Errecalde, M. 2005. Adaptive clustering with artificial ants. *Journal of Computer Science & Technology* **5**(4): 264–71.

Ingaramo, D., Pinto, D., Rosso, P., and Errecalde, M. 2008. Evaluation of internal validity measures in short-text corpora. In *Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics, CICLing 2008*, vol. 4919, pp. 555–67. Lecture Notes in Computer Science. Germany: Springer-Verlag Berlin Heidelberg.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Ingaramo, D., Rosas, M. V., Errecalde, M., and Rosso, P. 2011. Clustering iterativo de textos cortos con representaciones basadas en concepto. *Procesamiento del Lenguaje Natural* **46**: 19–26.

Jiang, D., Pei, J., and Zhang, A. 2003. DHC: a density-based hierarchical clustering method for time series gene expression data. In *Proceedings 3rd IEEE Symposium on BioInformatics and BioEngineering (BIBE 2003)*, California, USA: IEEE Computer Society, pp. 393–400.

Jing, L. 2005. Survey of text clustering. *Technical report*. Department of Mathematics. The University of Hong Kong, Hong Kong, China.

Karthikeyan, T., Peter, S. J., and Chidambaranathan, S. 2011. Hybrid algorithm for noise-free high density clusters with self-detection of best number of clusters. *International Journal of Hybrid Information Technology* **4**(2): 39–54.

Karypis, G., Han, E., and Kumar, V. 1999. Chameleon: hierarchical clustering using dynamic modeling. *IEEE Computer* **32**(8): 68–75.

Kaufman, L., and Rousseeuw, P. J. 1990. *Finding Groups in Data: An Introduction to Cluster Analysis*, New York, USA: John Wiley and Sons.

Kruskal, W. H., and Wallis, W. A. 1952. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association* **47**(260): 583–621.

Kyriakopoulou, A. 2008. Text classification aided by clustering: a literature review. In*Tools in Artificial Intelligence*, pp. 233–52. Rijeka, Croatia: INTECH.

Larocca Neto, J., Santos, A. D., Kaestner, C. A. A., and Freitas, A. A. 2000. Document clustering and text summarization. In N. Mackin (ed.), *Proceedings of the 4th International Conference Practical Applications of Knowledge Discovery and Data Mining (PADD-2000)*, London, England: The Practical Application Company, pp. 41–55.

Levene, H. 1960. Robust tests for equality of variances. In *Contributions to Probability and Statistics*, pp. 278–92. California, USA: Stanford University Press.

Liu, X., and Croft, W. B. 2004. Cluster-based retrieval using language models. In M. Sanderson, K. Jrvelin, J. Allan, and P. Bruza (eds.), *Proceedings of the 27th annual International ACM Conference on Research and Development in Information Retrieval (SIGIR 2004)*, New York, USA: ACM, pp. 186–93.

MacQueen, J. B. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, California, USA: University of California Press, pp. 281–97.

Makagonov, P., Alexandrov, M., and Gelbukh, A. 2004. Clustering abstracts instead of full texts. In *Text, Speech and Dialogue, 10th International Conference (TSD 2004)*, vol. 3206, pp. 129–35. Lecture Notes in Artificial Intelligence, Germany: Springer-Verlag Berlin Heidelberg.

Manning, C. D., Raghavan, P., and Schutze, H. 2008. *Introduction to Information Retrieval*. Cambridge, England: Cambridge University Press.

Ng, A. Y., Jordan, M. I., and Weiss, Y. 2001. On spectral clustering: analysis and an algorithm. In *The Proceedings of the 2001 Neural Information Processing Systems (NIPS) Conference*. vol. 14, pp. 849–56. Advances in Neural Information Processing Systems. Massachusetts, USA: MIT Press.

Ng, R. T., and Han, J. 1994. Efficient and effective clustering methods for spatial data mining. In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB '94)*, California, USA: Morgan Kaufmann Publishers Inc., pp. 144–55.

Paterlini, A. A., Nascimento, M. A., and Traina Jr., C. 2011. Using pivots to speed-up k-medoids clustering. *Journal of Information and Data Management* **2**(2): 221–36.

Pinto, D., Benedí, J. M., and Rosso, P. 2007. Clustering narrow-domain short texts by using the Kullback-Leibler distance. In *Computational Linguistics and Intelligent Text Processing, 8th International Conference (CICLing 2007)*, vol. 4394, pp. 611–22. Lecture Notes in Computer Science, Germany: Springer-Verlag Berlin Heidelberg.

Pinto, D., and Rosso, P. 2007. On the relative hardness of clustering corpora. In *Text, Speech and Dialogue, 10th International Conference (TSD 2007)*, vol. 4629,

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

pp. 155–61. Lecture Notes in Computer Science, Germany: Springer-Verlag Berlin Heidelberg. .

Pons-Porrata, A., Berlanga-Llavori, R., and Ruiz-Shulcloper, J. 2007. Topic discovery based on text mining techniques. *Information Processing and Management* **43**(3): 752–68.

Popova, S., and Khodyrev, I. 2011. Local theme detection and annotation with keywords for narrow and wide domain short text collections. In *Proceedings of the 5th International Conference on Advances in Semantic Processing (SEMAPRO 2011)*. Delaware, USA: International Academy, Reseach, and Industry Association (IARIA), pp. 49–55.

Qi, G.-J., Aggarwal, C., and Huang, T. 2012. Community detection with edge content in social media networks. In *IEEE 28th International Conference on Data Engineering (ICDE 2012)*, California, USA: IEEE Computer Society - Conference Publishing Services (CPS), pp. 534–45.

Rousseeuw, P. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics* **20**(1): 53–65.

Selim, S. Z., and Alsultan, K. 1991. A simulated annealing algorithm for the clustering problem. *Pattern Recognition* **24**(10): 1003–8.

Shannon, C. E. 1948. A mathematical theory of communication. *The Bell system technical journal* **27**: 379–423.

Slonim, N., Friedman, N., and Tishby, N. 2002. Unsupervised document classification using sequential information maximization. In *Proceedings of the 25th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR 2002)*, New York, USA: ACM, pp. 129–36.

Spearman, C. 1904. The proof and measurement of association between two things. *The American Journal of Psychology* **15**(1): 72–101.

Stein, B., and Busch, M. 2005. Density-based cluster algorithms in low-dimensional and high-dimensional applications. In *2nd International Workshop on Text-Based Information Retrieval (TIR 2005)*, Fachberichte Informatik, Mainz, Germany: University of Koblenz-Landau, pp. 45–56.

Stein, B., and Meyer zu Eissen, S. 2002. Document categorization with MajorClust. In*Proceedings Workshop on Issues in the Theory of Security (WITS 2002)*, Barcelona, Spain: Technical University of Barcelona, pp. 91–6.

Stein, B., and Meyer zu Eissen, S. 2004. Topic identification: framework and application. In *Proceedings of the International Conference on Knowledge Management (I-KNOW 2004)*, Graz, Austria: Know-Center, pp. 353–60.

Stein, B., Meyer zu Eissen, S., and Wißbrock, F. 2003. On cluster validity and the information need of users. In *3rd International Conference on Artificial Intelligence and Applications (AIA 2003)*, Benalmádena, Spain: ACTA Press, pp. 216–21.

Steinbach, M., Karypis, G., and Kumar, V. 2000. A comparison of document clustering techniques. In *Working Notes - Workshop on Text Mining. The 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Massachusetts, USA, pp. 109–11.

Steinberger, R., Pouliquen, B., Widiger, A., Ignat, C., Erjavec, T., Tufis, D., and Varga, D. 2006. The JRC-Acquis: a multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, Paris, France: European Language Resources Association (ELRA), pp. 2142–7.

Takeda, T., and Takasu, A. 2007. Updatenews: a news clustering and summarization system using efficient text processing. In *Proceedings of the 7th ACM/IEEE CS Joint Conference on Digital Libraries*, New York, USA: ACM, pp. 438–9.

Tan, P. N., Steinbach, M., and Kumar, V. 2005. *Introduction to Data Mining*. Massachusetts, USA: Addison-Wesley Longman Publishing Co..

Tu, L., and Chen, Y. 2009. Stream data clustering based on grid density and attraction. *ACM Transactions on Knowledge Discovery from Data* **3**: 1–27.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Tukey, J. W. 1953. The problem of multiple comparisons. In *The Collected Works of John W. Tukey VIII. Multiple Comparisons: 1948–1983*, pp. 1–300. New York, USA: Chapman and Hall.

Tukey, J. W. 1977. *Exploratory Data Analysis*. Massachusetts, USA: Pearson Education, Inc.

van Rijsbergen, C. J. 1979. *Information Retrieval*, 2 edn. London, England: Butterworths.

Xu, W., Liu, X., and Gong, Y. 2003. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR 2003)*. New York, USA: ACM, pp. 267–73.

Yang, T., Jin, R., Chi, Y., and Zhu, S. 2009. Combining link and content for community detection: a discriminative approach. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09. New York, USA: ACM, pp. 927–36.

Zha, H., He, X., Ding, C., Simon, H., and Gu, M. 2001. Spectral relaxation for k-means clustering. In *The Proceedings of the 2001 Neural Information Processing Systems (NIPS) Conference*, vol. 14, pp. 1057–64. Advances in Neural Information Processing Systems. Massachusetts, USA: MIT Press.

Zhang, D., Wang, J., and Si, L. 2011. Document clustering with universum. In *Proceedings of the 34th International ACM Conference on Research and Development in Information Retrieval (SIGIR 2011)*, New York, USA: ACM, pp. 873–82.

Zhou, Y., Cheng, H., and Yu, J. X. 2009. Graph clustering based on structural/attribute similarities. In *Proceedings of the VLDB Endowment*, vol. 2. Very Large Data Base Endowment Inc (VLDB Endowment), Lyon, France, pp. 718–29.

Zhao, Y., and Karypis, G. 2004. Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning* **55**: 311–31.

Zhao, Y., Karypis, G., and Fayyad, U. 2005. Hierarchical clustering algorithms for document datasets. *Data Mining and Knowledge Discovery* **10**: 141–68.

Appendix A: Computational complexity and execution time

The computational complexity of the algorithms used in the experimentation section is summarized in Table 15. Note that all values are expressed using the conventional mathematical $\mathcal{O}$ notation and n refers to the amount of documents of the corpus to be processed by the algorithms.

Table 16 shows the execution time of each algorithm with two collections, one of the smallest with lowest Relative Hardness measure: SEPLN-CICLing, and the largest one: R8-Train corpus. The intention is that the reader could have an estimation of the time needed by the algorithms. For obtaining these values, a

Table 15. *Computational complexities of algorithms considering collections with n documents*

Algorithm	Complexity
Sil–Att$(NULL)^{\star}$	$\mathcal{O}(n^2)$
K-means	$\mathcal{O}(n)$
K-MajorClust	$\mathcal{O}(n^2)$
Chameleon	$\mathcal{O}(n^2)$
CLUDIPSO	$\mathcal{O}(n^2)$
sIB	$\mathcal{O}(n^2)$

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press

Table 16. *Averaged execution times of algorithms*

Algorithm	CICLing-2002-F (48 documents)	R8-Train (5485 documents)
Rand-Clust	0.18	5.02
Sil–Att(NULL)*	0.76	5148.01
K-means	0.63	227.10
K-MajorClust	1.10	50327.33
Chameleon	12.00	NA
CLUDIPSO	2.56	NA
sIB	12.91	85221.12

NA: Not Available result.

personal computer with processor Intel(R) Core (TM)2 Quad, CPU 2.83 GHz and 3 GB RAM was used. We have carried out the same experiments described in Section 4.2 and we have obtained the averaged time of execution of a single run. The times are expressed in seconds.

https://doi.org/10.1017/S1351324915000273 Published online by Cambridge University Press