Improving the Dynamic Hierarchical Compact Clustering Algorithm by Using Feature Selection

Reynaldo Gil-García and Aurora Pons-Porrata

Center for Pattern Recognition and Data Mining
Universidad de Oriente, Santiago de Cuba, Cuba
{gil,aurora}@cerpamid.co.cu

Abstract. Feature selection has improved the performance of text clustering. In this paper, a local feature selection technique is incorporated in the *dynamic hierarchical compact* clustering algorithm to speed up the computation of similarities. We also present a quality measure to evaluate hierarchical clustering that considers the cost of finding the optimal cluster from the root. The experimental results on several benchmark text collections show that the proposed method is faster than the original algorithm while achieving approximately the same clustering quality.

1 Introduction

Managing, accessing, searching and browsing large repositories of text documents require efficient organization of the information. In dynamic information environments, such as the World Wide Web or the stream of newspaper articles, it is usually desirable to apply adaptive methods for document organization such as clustering. Dynamic algorithms have the ability to update the clustering when data are added or removed from the collection. These algorithms allow us dynamically tracking the ever-changing large scale information being put or removed from the web everyday, without having to perform complete re-clustering.

Hierarchical clustering algorithms have an additional interest, because they provide data-views at different levels of abstraction, making them ideal for people to visualize and interactively explore large document collections. In the context of hierarchical document clustering, the high dimensionality of the data and the large size of collections are the major challenges facing researchers today.

In [1], a hierarchical clustering algorithm, namely *dynamic hierarchical compact* (*DHC*) was introduced. It is not only able to deal with dynamic data while achieving a similar clustering quality than static state-of-the-art hierarchical algorithms, but also has a linear computational complexity with respect to the number of dimensions. It uses a multi-layered clustering to update the hierarchy when new documents arrive (or are removed). The process in each layer involves two steps: the updating of similarity-based graphs and the obtaining of the connected components for these graphs. The graph updating requires to compute the similarities between clusters, which is the most time-consuming operation.

Feature selection for text clustering is the task of disregarding irrelevant terms, aiming to find the smallest subset of terms that reveals “natural” clusters from

text data. Different measures such as scatter separability, entropy and document or term frequency have been proposed [2,3]. Unsupervised feature selection can be categorized as global or local approaches. The global feature selection chooses the relevant features once according to a ranking criterion, and uses the same feature subset in the whole clustering process. On the contrary, in local feature selection a different subset of features is chosen for each cluster.

The main contribution of this paper is twofold. Firstly, we present a version of *DHC* algorithm for clustering of dynamic document collections, in which a local feature selection strategy is incorporated to reduce the cost of computing similarities. Secondly, we present a new quality measure to evaluate hierarchical clustering solutions that considers the cost of finding the optimal cluster from the root. The experimental results on several benchmark text collections show that the proposed method is faster than *DHC* algorithm while achieving approximately the same clustering quality.

2 Feature Selection in Dynamic Hierarchical Compact Algorithm

DHC is an agglomerative method based on graph. It uses a multi-layered clustering to produce the hierarchy. The process in each layer involves two steps: updating of similarity-based graphs and obtaining the connected components for these graphs. Each connected component represents a cluster.

DHC algorithm uses two graphs. The first one is the β -similarity graph, which is an undirected graph whose vertexes are the clusters and there is an edge between vertexes i and j , if the cluster j is β -similar to i . Two clusters are β -similar if their similarity is greater than or equal to β , where β is a user-defined parameter. Analogously, i is a β -isolated cluster if its similarity with all clusters is less than β . As inter-cluster similarity measure we use group-average. In the vector space model, the cosine similarity is the most commonly used measure to compare the documents. By using the cosine measure, we can take advantage of a number of properties involving the composite vector of a cluster (i.e., the sum of document vectors of the cluster) [4]. In particular, the group-average similarity between clusters i and j is equal to the fraction between the scalar product of the composite vectors of these clusters and the product of clusters' sizes.

The second graph relies on the maximum β -similarity relationship (denoted as *max-S* graph) and it is a subgraph of the first one. The vertices of this graph coincide with vertices in the β -similarity graph, and there is an edge between vertices i and j , if i is the most β -similar cluster to j or vice verse.

Given a cluster hierarchy previously built by the algorithm, each time a new document arrives (or is removed), the clusters at all levels of the hierarchy must be revised (see Figure 1). When a new document arrives (or is removed), a singleton is created (or deleted) and the β -similarity graph at the bottom level is updated. Then, the *max-S* graph is updated too, which produce (or remove) a vertex and can also produce new edges and remove others. These changes on the *max-S* graph lead to the updating of the connected components. When

Fig. 1. Dynamic hierarchical compact algorithm

clusters are created or removed from a level of the hierarchy, the β -similarity graph at the next level must be updated. This process is repeated until this graph is completely disconnected (all vertices are β -isolated). It is possible that the β -similarity graph became completely disconnected before the top level of the hierarchy is reached. In this case, the next levels of the hierarchy must be removed. Notice that the algorithm uses the same β value in all hierarchy levels.

Algorithm 1. DHC with Local Feature Selection

1. Arrival of a document to cluster (or to remove).
 2. Put the new document in a cluster on its own (or remove the single cluster to which the document belongs).
 3. $level = 0$ and update the β -similarity graph at the bottom level, G_0 .
 4. While G_{level} is not completely disconnected:
 - (a) Update the $max-S$ graph at $level$.
 - (b) Update the connected components for the $max-S$ graph. Let N be the set of the new clusters and R be the set of the removed clusters.
 - (c) For each new cluster c in N :
 - i. Calculate the composite vector $\vec{c}$ of the cluster c as the sum of the composite vectors of its subclusters.
 - ii. Select the most $f \cdot dt(\vec{c})$ relevant terms of $\vec{c}$ and remove the remaining terms from it.
 - (d) Update the β -similarity graph at the next level, $G_{level+1}$.
 - (e) $level = level + 1$
 5. If there exist levels greater than $level$ in the hierarchy, remove them.
-

The updating of the β -similarity graph in *DHC* is trivial. For each vertex to add, the similarities with the remaining vertices are calculated and the corresponding edges are added to the graph. On the contrary, for each vertex to remove, all its edges are removed too. Notice that *DHC* needs to compute the similarities between the new document and all existing documents at the bottom level. Also, for each level of the hierarchy the similarities between the new clusters created at the previous level and the existing clusters at the corresponding level must be calculated too. The composite vector of each cluster is used to

compute these similarities. The computation of the composite vectors and the similarities between them are the most time-consuming operations.

Our proposal focuses on improving the performance of the β -similarity graph updating by applying a simple local feature selection criterion. Since *DHC* is an agglomerative method, both the size of the clusters and the number of distinct terms (features) in its composite vectors increase as we go up in the hierarchy. Thus, a feature selection criterion is applied to choose the most relevant terms for each cluster. The relevance of a term t in the cluster c is calculated as $w(t, c) = \sum_{d \in c} tf(t, d)$, where $tf(t, d)$ is the number of occurrences of t in the document d . Notice that $w(t, c)$ coincides with the weight of term t in the composite vector $\vec{c}$. The local selection is performed by using a variable number of features according to $dt(\vec{c})$, which is the number of distinct terms in the composite vector of the cluster c . The number of selected features in the cluster c is $f \cdot dt(\vec{c})$, where $f \in [0, 1]$. Notice that each cluster is represented by a different subset of features. Since the dynamic nature of our algorithm, a global feature selection criterion can not be applied.

The steps are shown in Algorithm 1. A detailed description of steps 4(a) and 4(b) can be seen in [1,5].

3 F1-Travel Quality Measure

Evaluation of clustering is usually done by comparing system-generated clusters to a “gold standard” (i.e., the manually labeled topics). Several measures have been proposed to evaluate the quality of flat clustering. However, the evaluation of hierarchical approaches is still an open problem. The manually labeled topics of the most standard text collections have currently a flat structure. It is due to historical reasons and the difficulties for humans to build the topic hierarchy.

The most widely used measure is *overall F1* [6], which compares the system-generated clusters at all levels of the hierarchy with the manually labeled topics. It is calculated as: $Overall\ F1 = \frac{\sum_{i=1}^{|T|} |t_i| F1(t_i, \sigma(t_i))}{\sum_{i=1}^{|T|} |t_i|}$, where T is the set of manual topics and $\sigma(t_i)$ is the “best matching” cluster with the topic t_i , i.e., the cluster that maximizes $F1(t_i, c_j) = 2 |t_i \cap c_j| / (|t_i| + |c_j|)$.

The *extended F1-BCubed* measure to evaluate hierarchical clustering was recently proposed in [7]. Extended F1-BCubed is the F1-measure evaluated over extended BCubed precision and recall. It takes into account the multiplicity of document occurrences in clusters and topics.

Let us suppose that we have a collection of 8 documents and the manual topics are $\{1,2\}$, $\{3,4\}$, $\{5,6\}$ and $\{7,8\}$. Figure 2 shows three possible hierarchies obtained by clustering algorithms. As we can observe, extended F1-BCubed considers the hierarchy (a) better than (b) despite it does not contain any “perfect” cluster. It is due to that the number of document pairs that occur in the clusters of the hierarchy (a) is more similar to that in the topics than to that in the hierarchy (b). For example, the pair (1,2) occurs twice in (a) and four times in (b), while occurs only once in the topics. Both extended F1-BCubed and overall F1 measure can not distinguish the hierarchy (b) from (c).

Fig. 2. Example of hierarchy evaluation by using different measures

We see the hierarchy as providing paths for traveling between topics. Deep hierarchies may not be suitable for browsing, since a user may require a high number of navigation steps to finding the topics of her interest. However, if a hierarchy is too flat, a parent topic may contain too many subtopics and it would increase the time and difficulty for the user to locate her target. Therefore, we expect that the number of explored nodes to find the “best matching” clusters be as small as possible. In this sense, the hierarchy (c) is preferable.

In the context of Topic Detection and Tracking, Allam et al. [8] proposed the minimal cost measure, which is defined as a linear combination of detection cost and travel cost. The former cost is expressed in terms of missed and false alarm rates, while the latter is defined as the search cost to find the optimal cluster from the root. Following this idea, we propose the *F1-Travel* measure. It is very similar to overall F1-measure, but differs in that we add a travel cost to the measure. Thus, we defined *overall F1-Travel* as follows:

$$\text{Overall } F1\text{-Travel} = \frac{\sum_{i=1}^{|T|} |t_i| F1\text{-Travel}(t_i)}{\sum_{i=1}^{|T|} |t_i|}$$

To evaluate each topic t_i , we use a best-first search to find the optimal cluster (see Algorithm 2). That is, $F1\text{-Travel}(t_i) = F1(t_i, \sigma(t_i))(1 - \frac{v}{2n})$, where n is the number of documents in the collection, $\sigma(t_i)$ is the “best-matching” cluster with t_i found by the best-first search, and v denotes the number of visited clusters while looking $\sigma(t_i)$. Notice that $2n$ represents the worst case (i.e., all clusters of the largest hierarchy are visited). The greater number of visited clusters required to find the “best matching” clusters, the smaller F1-Travel is.

As we can observe in Figure 2, unlike extended F1-BCubed and overall F1, our measure considers the hierarchy (c) better than (b), because 20 clusters are visited to find all topics in (c), whereas 22 clusters are explored in (b).

Algorithm 2. F1-Travel for the topic t

-
1. Let Q be a null queue of nodes, sorted by decreasing $F1$
 2. $v = 1$, $BestF1 = 0$, and insert $root$ into Q
 3. while $Q \neq \emptyset$:
 - (a) Extract the first cluster c from Q
 - (b) if $F1(t, c) > BestF1$:
 - i. $BestF1 = F1(t, c)$
 - (c) $v = v + |c.childs|$ // $|c.childs|$ is the number of childs of c in the hierarchy
 - (d) for each child c' of c :
 - i. if $F1(t, c') > BestF1$:
 - A. Insert c' into Q
 4. $F1-Travel(t) = BestF1(1 - \frac{v}{2n})$
-

4 Experimental Results

The performance of the proposed version of dynamic hierarchical compact algorithm has been evaluated using six benchmark text collections, whose general characteristics are summarized in Table 1. They are heterogeneous in terms of document size, number of topics and document distribution. Human annotators identified the flat topics in each collection. In our experiments, the documents are represented using the traditional vector space model. Document terms represent the lemmas of the words appearing in the texts (stop words are disregarded) and they are statistically weighted using TF (term frequency in the document).

The experiments were focused on comparing the proposed version against the original *DHC* algorithm in terms of clustering quality and time efficiency. From the results reported in [5], we fix the parameter $\beta = 0.02$, which produces good hierarchies for all text collections.

To quantitatively compare the relative performance of both methods, we divided the overall F1 score obtained by the proposed method by the corresponding score obtained by *DHC*. We referred to this ratio as *relative overall F1* score. In the same way, we calculated *relative extended F1-BCubed* and *relative overall F1-Travel* scores. We also computed the speedup obtained by the proposed method, that is, the ratio between the execution times of *DHC* and our method.

Table 1. Description of document collections

Collection	Source	Docs	Terms	Topics
eln	TREC-4 (http://www.trec.nist.gov)	5829	83434	50
hitech	San Jose Mercury (http://glaros.dtc.umn.edu)	2301	13170	6
new3	San Jose Mercury (http://glaros.dtc.umn.edu)	9559	83487	44
oshcal	Ohsumed-233445 (http://glaros.dtc.umn.edu)	11162	11465	10
reu	Reuters-21578 (http://kdd.ics.uci.edu)	10369	35297	119
tdt	TDT2 (http://www.nist.gov/speech/test/tdt.html)	9824	55112	193

Fig. 3. Speedup and relative scores obtained by our method w.r.t. *DHC*

Figure 3 shows the speedups and the relative quality scores, when we vary the fraction of relevant terms (f) from 0.1 to 0.7 for each text collection. At a glance, we can observe in Figure 3(a) that speedups between 2 and 6 are achieved for all collections when $f \leq 0.5$. As expected, the greater reduction, the higher speedups are obtained.

From the Figures 3(c) and 3(d), we can see that the overall F1 and overall F1-Travel scores slightly decrease (about 5%) or even improve in all collections while reducing the number of features per cluster in the fraction between 0.2 and 0.5. The slight decrease in the relative overall F1-Travel score w.r.t. the relative overall F1 score demonstrates that the number of navigation steps to finding the “best matching” clusters is not significantly increased with the feature selection.

The extended F1-BCubed measure shows a different behavior (see Figure 3(b)). In this case, our method obtains better scores than those obtained by the original *DHC* algorithm for all feature reductions. The smaller the number of selected terms per cluster, the higher extended F1-BCubed scores are achieved, specially when few features are selected (about 10-20%). This can be explained by the fact that both the number of hierarchy levels and clusters’ size decrease, which causes that the number of document pairs that occur in the clusters

diminishes. This produces an effect of boosting in the calculation of the extended F1-BCubed values that actually hides the performance of the method.

To sum up, we can conclude that speedups greater than 2.5 can be achieved with less than 5% loss in clustering quality when we select between 20-45% of the original number of terms for each cluster. Thus, our method achieves both good clustering quality and efficiency improvement w.r.t. the original *DHC* algorithm.

5 Conclusions

In this paper, a version of *DHC* algorithm has been proposed. Since the dynamic nature of the method, a local feature selection criterion is applied to remove the irrelevant terms that may degrade the clustering accuracy, and to reduce the cost of computing similarities. We also present a new quality measure to evaluate hierarchical clustering solutions that considers the cost of navigating through the hierarchy to find the optimal clusters.

The experimental results on several benchmark text collections show that the proposed method significantly reduces the computation times of the original *DHC* algorithm while maintaining the clustering quality. Thus, we advocate its use for tasks that require dynamic clustering of large text collections, such as creation of document taxonomies and hierarchical topic detection.

As future work, we plan to combine the proposed method with the speedup version presented in [9], to further improve *DHC* algorithm.

References

1. Gil-García, R., Badía-Contelles, J.M., Pons-Porrata, A.: Dynamic Hierarchical Compact Clustering Algorithm. In: Sanfeliu, A., Cortés, M.L. (eds.) CIARP 2005. LNCS, vol. 3773, pp. 302–310. Springer, Heidelberg (2005)
2. Dy, J.G., Brodley, C.E.: Feature selection for unsupervised learning. *Journal of Machine Learning Research* 5, 845–889 (2004)
3. Ribeiro, M.N., Neto, M.J.R., Prudêncio, R.B.C.: Local feature selection in text clustering. In: Köppen, M., Kasabov, N., Coghill, G. (eds.) ICONIP 2008. LNCS, vol. 5507, pp. 45–52. Springer, Heidelberg (2008)
4. Zhao, Y., Karypis, G.: Evaluation of hierarchical clustering algorithms for document datasets. In: 11th CIKM, pp. 515–524. ACM Press, New York (2002)
5. Gil-García, R., Pons-Porrata, A.: Dynamic hierarchical algorithms for document clustering. *Pattern Recognition Letters* 31(6), 469–477 (2010)
6. Larsen, B., Aone, C.: Fast and Effective Text Mining Using Linear-time Document Clustering. In: KDD 1999, pp. 16–22. ACM Press, New York (1999)
7. Amigó, E., Gonzalo, J., Artiles, J., Verdejo, F.: A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Inform. Retrieval* 12, 461–486 (2009)
8. Allan, J., Feng, A., Bolivar, A.: Flexible intrinsic evaluation of hierarchical clustering for tdt. In: 12th CIKM, pp. 263–270. ACM Press, New York (2003)
9. Gil-García, R., Pons-Porrata, A.: A speed-up hierarchical compact clustering algorithm for dynamic document collections. In: Bayro-Corrochano, E., Eklundh, J.-O. (eds.) CIARP 2009. LNCS, vol. 5856, pp. 379–386. Springer, Heidelberg (2009)