Instituto de Ciências Matemáticas e de Computação
Computer Science and Mathematics Institute

ISSN - 0103-2569

A confidence-based active approach for semi-supervised hierarchical clustering

Bruno Magalhães Nogueira
Alípio Jorge
Solange Oliveira Rezende

RELATÓRIOS TÉCNICOS DO ICMC
ICMC TECHNICAL REPORTS

São Carlos
October/2011

A confidence-based active approach for semi-supervised hierarchical clustering

Bruno Magalhães Nogueira[1]
Alípio Jorge[2]
Solange Oliveira Rezende[1]

[1]University of Sao Paulo – USP
Institute of Mathematics and Computer Science – ICMC
Artificial Intelligence Laboratory – LABIC
P.O. Box 668
13560-970 - Sao Carlos – SP – Brasil

[2]Laboratory of Artificial Intelligence and Decision Support - LIAAD
INESC Porto L.A.
Department of Computer Science
Faculty of Sciences of the University of Porto
Porto, Portugal

brunomn@icmc.usp.br,amjorge@fc.up.pt,solange@icmc.usp.br

Abstract. *Semi-supervised approaches have proven to be effective in clustering tasks. They allow user input, thus improving the quality of the clustering obtained, while maintaining a controllable level of user intervention. Despite being an important class of algorithms, hierarchical clustering has been little explored in semi-supervised solutions. In this report, we address the problem of semi-supervised hierarchical clustering by using an active clustering solution with cluster-level constraints. This active learning approach is based on a new concept of merge confidence in an agglomerative clustering process. When there is lower confidence in a cluster merge the user can be queried and provide a cluster-level constraint. The proposed method was compared with a unsupervised algorithm (average-link) and a semi-supervised algorithm based on pairwise constraints. The results show that our algorithm tends to be better than the pairwise constrained algorithm and can achieve a significant improvement when compared to the unsupervised algorithm.*

Contents

1. Introduction

Semi-supervised clustering has been widely explored in the last years. Instead of finding groups guided only by an objective function, as in traditional unsupervised clustering algorithms, semi-supervised versions try to improve clustering results by employing external knowledge in the clustering process. The external knowledge is conveyed in form of constraints. These constraints can be directly derived from the original data (using partially labelled data) or provided by an user, trying to adapt the clustering results to his/her expectations (Dasgupta and Ng, 2010).

Constraints in semi-supervised clustering processes are related to a small part of the dataset, as the supervision of large amounts of data is expensive (Bilenko et al., 2004). So, it is very important to optimize the usage of external knowledge, obtaining the largest amount of useful information from the smallest number of constraints. In this sense, semi-supervised clustering algorithms must deal with two crucial aspects: how to add information to the clustering process (add information in a proper way) and to which cases (when) the user should provide information.

To assure efficacy in information addition, the characteristics of the semi-supervised algorithm are very important. Mainly, three aspects can be observed: (1) the type of the constraints that will be used (for example, pairwise constraints (Wagstaff and Cardie, 2000; Vu et al., 2010; Miyamoto and Terami, 2010), initial seeds (Basu et al., 2002) or feedback (Dasgupta and Ng, 2010)); (2) the level of the constraints (instance-level Wagstaff and Cardie (2000), cluster-level Huang and Mitchell (2008) or instance-cluster-level (Eick et al., 2004; Huang and Mitchell, 2006)); and (3) how the algorithm deals with these constraints (constraint-based (Wagstaff and Cardie, 2000), distance-based (Kumar et al., 2005) or hybrid (Bilenko et al., 2004)).

Active learning algorithms (Settles, 2009) can be used to choose proper cases to add information. In semi-supervised clustering, these algorithms can be used to detect instances or clusters to which the addition of constraints can better help the clustering process to obtain an optimal solution. Active learning algorithms have been successfully used to select pairs of instances to elicit pairwise contraints from the user (Huang and Lam, 2009; Vu et al., 2010). Also, algorithms that employ seeds use active-based solutions to choose better initial seeds (Basu et al., 2004).

In the literature few works deal with semi-supervised hierarchical clustering, despite the popularity and effectiveness of this category of clustering methods. More specifically, neither the apropriate addition of information nor the selection of good cases to add constraints are fairly explored in hierarchical clustering context.

Looking for a better use of external knowledge on the hierarchical clustering process, this work presents a new method called **HCAC** (Hierarchical Confidence-Based Active Clustering). The HCAC method improves a hierarchical clustering process by hierarchically adding cluster-level constraints, i. e., the user is asked to insert cluster-level constraints along the hierarchical clustering process when it seems more appropriate. This method approaches two aspects which are not explored in the literature. The first one is related to the kind of query that is presented to the user, which allows this user, at a given iteration, to determine the next pair of clusters to be merged among a pool of pre-selected pairs. The second aspect is related to the development of an active learning method to determine when it is appropriate to make a query, based on a new concept of confidence in a cluster merging decision.

This report is organized as follows. In the next section, we present some related work on hierarchical semi-supervised clustering and active learning algorithms. In Section 3, we present the proposed HCAC algorithm. Then, in Section 4, we present some experiments that evaluate the proposed algorithm. Finally, in Section 5, we present some conclusions and point some future works.

2. Related work

There are few works on semi-supervised hierarchical clustering. Among the main works, in Klein et al. (2002), instance-level pairwise constraints (must-link and cannot-link) are used in a semi-supervised clustering algorithm based on the complete-link algorithm (see Jain and Dubes (1988)). Constraint insertion has two phases: imposition and propagation. During the imposition, constraints are added to pairs of examples. The algorithm transforms the feature space into a similarity space by modifying the distance between elements. If two points x_i and x_j have a must-link constraint, then their distance is set to zero. Otherwise, if they have a cannot-link constraint, their distance is set to the maximum distance on the distance matrix plus one. In the propagation, the algorithm considers that if an example x_k is near an example x_i and x_i has a must-link or a cannot-link constraint with x_j, so x_k is also near or far from x_j. The new distance between x_k and x_j is calculated through a triangle inequality.

Kestler et al. (2006) use pairwise constraints at the first level of a hierarchical clustering algorithm in order to generate the initial clusters. The constraints are not propagated to posterior levels as the algorithm aims to generate stable dendrograms.

Using labeled examples, an algorithm also based on the complete-link algorithm is proposed in Daniels and Giraud-Carrier (2006). This algorithm learns a distance threshold x from which there are no more cluster merges. In order to learn this distance threshold, the algorithm uses a small set of labeled objects. This small set is clustered and several threshold values are tested. The value which presents the best evaluation measure is chosen for clustering the entire data set.

Labeled examples are used in Bade et al. (2007) in a post-processing step. The method uses the labeled instances to generate must-link and cannot-link constraints between pairs of objects. So, after an unsupervised clustering process, these constraints are used to determine whether to merge or split the resulting clusters.

In Böhm and Plant (2008) a semi-supervised density-based hierarchical algorithm is proposed. Labeled data are used to generate an initial hierarchy, which is later expanded. So, unlabeled data are assigned to the most consistent clusters, according to the cluster structure predefined.

Finally, in Davidson and Ravi (2009) the authors analysed the use of pairwise constraints and cluster-level constraints (minimum and maximum intra-cluster distances). The authors prove that the combination of these constraints is computationally viable in a hierarchical clustering, unlike flat clustering methods, where this combination is a NP-Complete problem.

Among these works, only in Klein et al. (2002) it is possible to find the usage of an active learning algorithm which inserts constraints in a hierarchical clustering process. In this active approach, the algorithm is allowed to perform m pairwise questions. So, the algorithm performs an entirely unsupervised complete-link clustering process in order to learn a distance α from which it is expected to need no more than m questions to cluster properly. The clustering restarts in an unsupervised way until it makes a merge of distance α. Then, the user is asked whether the roots of the next proposed merge belong together. According to the answer, the constraints are propagated, as explained before.

It is possible to see that semi-supervised hierarchical clustering is still little explored and there are almost no active learning approaches to these algorithms. In the next section, we present our active hierarchical clustering algorithm, which is based on a new concept of cluster merge confidence.

3. HCAC: a confidence-based active clustering method

The HCAC (Hierarchical Confidence-Based Active Clustering) is a new semi-supervised clustering method based on an agglomerative hierarchical clustering process. This algorithm uses cluster-level constraints which are provided by a human supervisor along the iterations of an agglomerative hierarchical clustering algorithm. In the next section, we will identify the kind of situation that motivated us to create this method and our approach to detect these situations. Then, we will explain our approach to deal with these situations by adding cluster-level constraints.

3.1. Confidence-based active clustering

In an unsupervised agglomerative hierarchical clustering procedure, the pair of elements[1] that present the shortest distance between them in a given step is selected to be merged. However, sometimes this approach may cluster objects that represent different concepts not fully represented by the distance function.

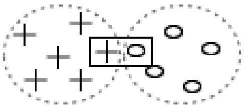

Figure 1. Cluster border problem

This often occurs near cluster borders, as presented in Figure 1. In this figure, it is clear that we have two underlying clusters (dashed circles), corresponding to two different concepts. In an unsupervised approach, despite representing different concepts, the pair of elements in

[1] In this work, the term elements may refer both to single examples or clusters.

the rectangle would be the first to be merged, as they are nearest. However, there are better options nearby to cluster with one of these two elements, since they are also close to other elements that belong to the same concept.

Motivated by this kind of situation, in this work we present the concept of **confidence** of a merge. The confidence of a merge is related to the distance between the elements from the proposed merge and other elements near them. If a pair of elements are close to each other but far from other elements, the confidence of merging these two elements is high since apparently there is no good alternative. However, if they are also close to other elements, it might be advisable to ask a human supervisor to check if there is a better merge.

Formally, a confidence value can be calculated as follows. Considering a distance matrix M and a distance function $dist(.,.)$ between elements in M, the natural merge is between the nearest pair of elements a and b, where $dist(a,b) = d_{a,b} = \arg\min_{x,y} dist(x,y),\ x \neq y$. The confidence C of this merge is calculated by the difference between $d_{a,b}$ and $d_{e,f}$, where $d_{e,f} = \arg\min_{x,y} dist(x,y), x \neq y, (x,y) \neq (a,b), x \in \{a,b\} \vee y \in \{a,b\}$.

Therefore, merges having low confidence values are taken as points where the algorithm is more likely to make incorrect decisions (misclusterings). To reduce misclusterings, the proposed algorithm HCAC detects low confidence merges and queries the human to check whether a better alternative merge exists.

In practical terms, low confidence merges are those where confidence below a predefined threshold. The higher the threshold value, the more user interaction is requested. In this work, we also propose a calibration procedure to estimate this threshold with respect to the amount of tolerated interaction. This is done through an unsupervised execution of the hierarchical clustering algorithm, in a spirit similar to Klein et al. (2002). At each step of this unsupervised execution, the confidence value is calculated. At the end of this procedure an adequate threshold value is selected according to the desired number of human interactions. This procedure is described on Algorithm 1.

Once the threshold is calibrated, we have a criterion for deciding when to make queries to the human. In the next section, we will explain

```
Input: $n$: number of elements in the dataset; $M$: distance
       matrix; $dist(.,.)$: distance function; $q$: desired number of
       human interactions
Output: $confT$: confidence threshold value
Initialize vector $C$ with $n-1$ positions;
for $k = 1 : n-1$ do
    $minDist_k = d_{i,j} = \arg\min_{x,y} dist(x,y), x \neq y$;
    $secMinDist_k = d_{r,s} = \arg\min_{x,y} dist(x,y), x \neq y, (x,y) \neq (i,j), x \in \{i,j\} \vee y \in \{i,j\}$;
    $C_k = secMinDist_k - minDist_k$;
end
Order vector $C$;
$confT = C[q]$;
```

Algorithm 1: Threshold calibration procedure

how the user can interact with the HCAC in order to guide the clustering process.

3.2. Cluster-level constraints

When a low confidence merge is spotted, the user is queried for additional information. The response comes in the form of a clustering constraint. In general, constraints can be stated at the instance level (to what degree two instances should be in the same cluster) or at the cluster level, where we consider whole subclusters instead of single instances. In our proposal, we use cluster-level constraints. Cluster-level constraints can obviously convey more information than instance-level ones. This can reduce the number of user interventions. Instance level queries, however, can be more easily resolved by the human.

In HCAC, a cluster-level query is posed to acquire a cluster-level constraint when a low confidence merge is detected. For that, a pool of pairs of clusters is presented to the user. Then, the user chooses the pair that corresponds to the best merge. The pool contains c nearest pairs of clusters, where c is given a priori. The generation of this pool is done as described in Algorithm 2. It starts by finding the best unsupervised merge (the two nearest clusters i, j). After that, the $c-1$ best unsupervised merges involving i or j are included. This assembling procedure has a linear-time cost in function of the number of elements on the distance matrix ($O(n)$, where n is the number of elements).

The higher the value of c, the more options the user has, and the

```
Input: $n$: number of elements in the dataset; $M$: distance matrix; $dist(.,.)$: distance function; $c$: size of the pool of clusters
Output: $P_k$: pool of pairs of clusters on the $k$-th iteration
Initialize vector $P$ with $c$ positions;
$P[1] = (i,j) | dist(i,j) = \arg\min_{x,y} dist(x,y), x \neq y$;
for $l = 2 : c$ do
    $P[l] = (r,s) | (r,s) \notin P, dist(r,s) = \arg\min_{x,y} dist(x,y), x \neq y, (x,y) \neq (i,j), x \in \{i,j\} \vee y \in \{i,j\}$;
end
```

Algorithm 2: Procedure for assembling the pool of cluster pairs

brighter are the chances of finding a good choice. However, a large number of cluster pairs may imply excessive human effort.

The adoption of the active confidence-based approach tries to optimize the user intervention. Moreover, the adoption of this kind of cluster-level constraints and this new kind of queries tends to generate clusters with high density and high purity degrees.

4. Experimental evaluation

To evaluate the HCAC method, we have used 22 datasets from the UCI repository[2]. All of these datasets have labeled instances which makes possible to objectively evaluate the clustering results. These datasets are approximately balanced. A brief description of these datasets can be observed in Table 1. The evaluation methodology applied on these datasets and the obtained results are presented in the following sections.

Table 1. Description of used datasets

Dataset	# Examples	# Classes	Dataset	# Examples	# Classes
Balance	625	3	Madelon	600	2
Breast Cancer Wisc.	683	2	Mammographic	830	2
Breast Tissue	106	6	Musk	476	2
Ecoli	336	8	Pima	768	2
Glass	214	6	Secom	1151	2
Haberman	306	2	Sonar	208	2
Image Segmentation	210	7	Soybean	266	15
Ionosphere	351	2	Spectf	267	2
Iris	150	3	Transfusion	748	2
Libras	360	15	Vehicle	846	4
Lung Cancer	27	3	Wine	178	3

[2] http://archive.ics.uci.edu/ml/datasets.html

4.1. Evaluation methodology

We have compared the HCAC with two standards: an unsupervised algorithm, which is used as a baseline; and a semi-supervised algorithm using must-link and cannot-link pairwise constraints Wagstaff and Cardie (2000). All three approaches use the average-link strategy (Jain and Dubes, 1988).

In the experiments, we simulated the human interaction in the semi-supervised algorithms by using the labels provided with the data sets. The idea is to automatically answer the queries using a sensible criteria that models the user behaviour. The automatic answer is entirely derived from the instance labels. In HCAC, for the cluster-level queries, the criteria for choosing the best cluster merge is entropy (Shannon, 2001). Among the pairs in the pool, the one with the lowest entropy value is selected for merging. For the algorithm using pairwise constraints, we randomly pick pairs of instances before the clustering process starts. As suggested in Davidson and Ravi (2009), if the elements belong to the same class, then a must-link constraint was added and the distance between this pair was set to zero. Otherwise, a cannot-link constraint was added and the distance was set to infinity.

We have tried different numbers of human interventions in the clustering process (number of pairwise queries or cluster-level queries). We have varied the number of desired interventions in 1%, 5%, 10%, 20% ... 100% of the number of merges in the agglomerative clustering process (which is equal to the number of instances in the dataset minus one). In the case of the HCAC algorithm, we have also tested three different number of pair of elements in the pool: 5, 10 and 20.

During the evaluation procedure, we have used 10-fold cross validation. For each dataset, in each experiment configuration, the algorithms were applied 10 times, always leaving one fold out of the dataset. Each resulting clustering was evaluated through the FScore measure (Larsen and Aone, 1999; Gil-García and Pons-Porrata, 2010) which is very adequate for hierarchical clustering. The FScore for each class K_i is the maximum value of FScore obtained at any cluster C_j of the hierarchy, which can be calculated according to Equation 1:

$$F(K_i, C_j) = \frac{2 * R(K_i, C_j) * P(K_i, C_j)}{R(K_i, C_j) + P(K_i, C_j)} \tag{1}$$

Table 2. Results of the statistical comparisons. The symbol $>>$ indicates that HCAC wins with statistical significance; $>$ indicates that HCAC wins with no statistical significance; $<$ indicates that HCAC loses with no statistical significance. Each symbol is followed by the number of victories and losses of HCAC.

	5 Pairs				10 Pairs				20 Pairs			
%	Pairwise		Average		Pairwise		Average		Pairwise		Average	
1	>>	12 - 6	<	4 - 6	>	13 - 5	<	5 - 6	>	13 - 5	>	7 - 4
5	>	12 - 10	<	7 - 11	<	8 - 14	<	7 - 10	>	13 - 9	>	12 - 7
10	>	12 - 9	>	11 - 8	>	12 - 10	>	11 - 8	>	12 - 10	>	14 - 6
20	-	11 - 11	<	8 - 12	<	10 - 12	<	7 - 12	>	14 - 8	>	13 - 7
30	>	12 - 10	<	10 - 12	-	11 - 11	>>	14 - 8	>	15 - 7	>>	18 - 4
40	<	9 - 13	>>	16 - 6	>	12 - 10	>>	16 - 6	>	12 - 10	>>	16 - 6
50	<	10 - 12	>>	14 - 8	-	11 - 11	>>	18 - 4	>	13 - 9	>>	18 - 4
60	>	12 - 10	>>	18 - 4	>	12 - 10	>>	19 - 3	>>	15 - 7	>>	20 - 2
70	-	11 - 11	>>	18 - 4	>	13 - 9	>>	21 - 1	>>	17 - 5	>>	22 - 0
80	>	12 - 10	>>	22 - 0	>>	14 - 8	>>	21 - 1	>>	17 - 5	>>	22 - 0
90	>	12 - 10	>>	21 - 1	>>	18 - 4	>>	22 - 0	>>	20 - 2	>>	21 - 1
100	>>	21 - 1	>>	22 - 0	>>	22 - 0	>>	22 - 0	>>	22 - 0	>>	22 - 0

where $R(K_i, C_j)$ is the recall value for the class K_i on the cluster C_j, which is defined as n_{ij} / size of C_j (n_{ij} is the number of elements in C_j that belongs to K_i) and $P(K_i, C_j)$ is the precision value, defined as n_{ij} / size of K_i. The FScore value for a clustering is calculated by the weighted average of the FScore for each class, as shown on Equation 2.

$$FScore = \sum_{i=1}^{c} \frac{n_i}{n} F(C_i) \tag{2}$$

The non-parametric Wilcoxon statistical test (Wilcoxon, 1945) was used to detect statistical significance in the differences of the algorithms performance considering an α of 0.05. The test was applied to compare the HCAC algorithm against one of the other algorithms.

4.2. Results

The results of the comparisons of the different algorithms and configurations can be observed in Table 2. From these results, we can see that the more pairs are presented to the user in the HCAC algorithm, the better the algorithm performs. This effect can also be observed in the results presented in Figure 2. This is an expected behaviour, since the more pairs the user is shown, the higher the probability of finding a good solution. In other words, HCAC is able to exploit extra information. However, increasing the number of pairs in the pool has a cognitive cost.

Comparing the HCAC method with the baseline unsupervised algorithm (Average), we can see that when the pool is smaller (5 and 10 pairs), there is only a clear advantage between 30% and 40% of user in-

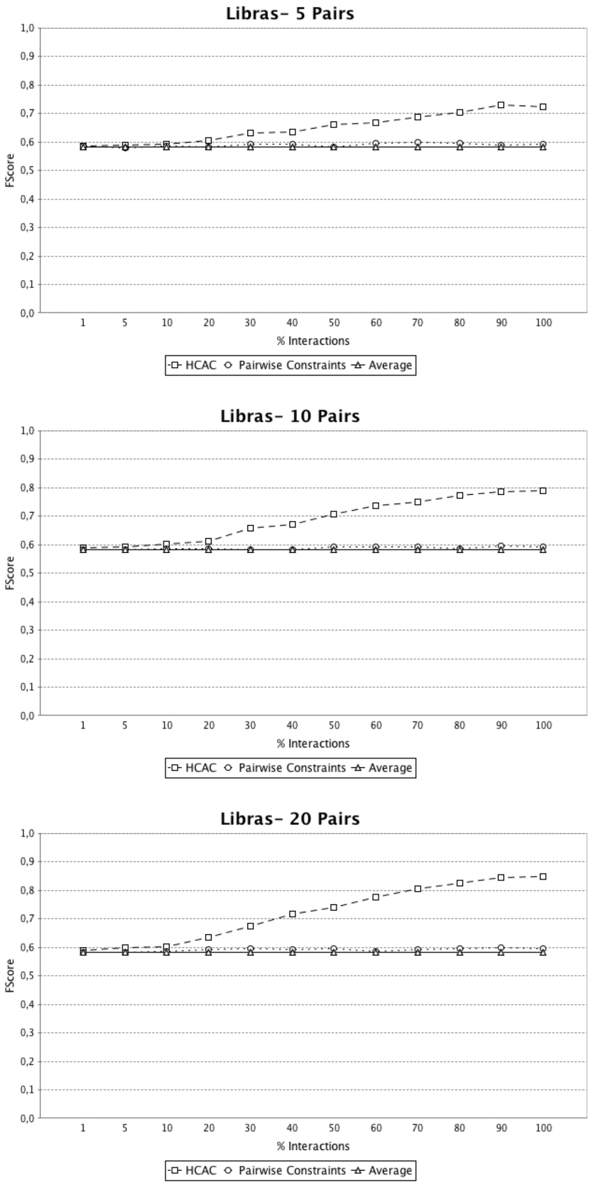

Figure 2. Results for the Libras dataset using 5, 10 and 20 pairs on the pool

terventions. With less interventions there are non significant wins and losses. With 20 pairs in the pool (and more effort per query), the HCAC algorithm has the tendency of always outperforming the unsupervised clustering approach, significantly with 30% or more interventions.

When using 5 and 10 pairs in the pool, the performance of the HCAC algorithm is very similar to the pairwise constrained approach, alternating winnings and losses. This indicates that, in a general way, the quality of information added to the clustering algorithm is very similar in both approaches. When using 20 pairs, the selected pools of pairs present very good results and the scenario is much more favourable to the HCAC algorithm, with the tendency of always winning.

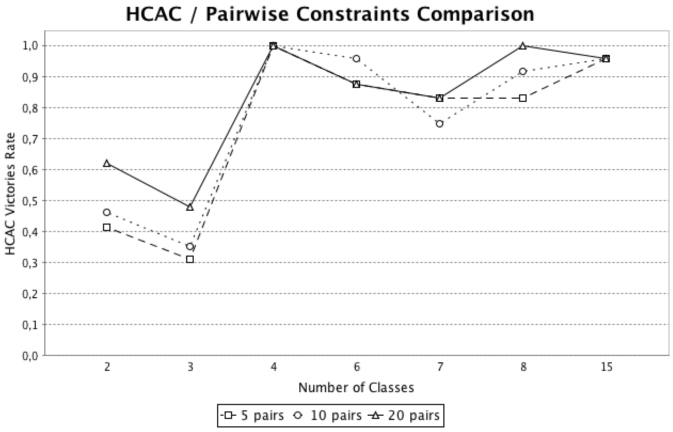

Figure 3. Comparison of HCAC and pairwise-constrained approach

Another interesting point is the relation between the number of natural groups (classes) in the dataset and the performance of the algorithm. In Figure 3, we present a comparison of HCAC and pairwise-constrained approaches according to the number of classes of the dataset. In the horizontal axis we have the number of classes. For each number of classes, we have calculated the victories rate of the HCAC algorithm over the algorithm that uses pairwise constraints. The HCAC victories rate is the proportion of the cases where HCAC presents higher FScore than pairwise with respect to the total number of comparisons considering all datasets with the same number of classes and all user intervention percentages. Three different victories rate lines were then plotted, one for each experiment configuration (5, 10 and 20 pairs). According to the results, the HCAC algorithm tends to have more advantage over pairwise (rate above 0.5) in datasets of more than 3 classes.

This tendency can be explained by the nature of the constraints. In general, the more clusters a dataset has, the more information will be needed to correctly delimit them. With the pairwise constraints, the user indicates whether two instances do or do not belong to the same cluster. On the other hand, our proposed cluster-level constraints indicate that two groups of instances must be merged. So, in the cluster-level constraint the number of instances influenced and the quantity of information added are higher. Moreover, our active learning approach tends to require the user intervention on points that can be regarded as cluster borders. The more clusters a dataset has, the more border regions are present and the higher are the chances of misclusterings.

5. Conclusions

In this work, we presented the HCAC, a new active semi-supervised hierarchical clustering method. This method uses cluster-level constraints where the user can indicate a pair of clusters to be merged. It also uses a new active learning process based on the concept of merge confidence. We have also devised a method for determining the adequate confidence threshold given a maximum amount of user effort allowed.

The empirical analysis shows that the proposed approach is very promising. When compared to the pairwise constrained approach, the HCAC method showed a slight advantage using pools of 5 and 10 pairs and a tendency to outperform when using 20 pairs. The method also has the advantage of pre-selecting a pool of clusters for the user in linear-time, reducing the number of pairs to be analysed by the user and presenting good results. Moreover, the algorithm presented a good performance when compared to the unsupervised algorithm, specially when using 20 pairs even with a small number of interactions. These results indicate that it is worthwhile to exploit the concept of confidence. Empirical results also indicate that HCAC is particularly useful with datasets of more than 3 classes, which is the case of many real life applications.

The application of the HCAC method has the limitation of requiring an adequate description of the groups when presenting the pairs of elements to the user. A poor description of the groups may lead the user to incorrect decisions.We are currently investigating how to adequately formulate the cluster-level queries so that the user can provide the constraints with minimal cognitive effort. One possibility is the use of bi-dimensional plots to represent the clusters overimposed by descriptor labels automatically extracted.

In future works, we intend to improve the performance of the HCAC method by exploiting constraint propagation, as in Klein et al. (2002). Furthermore, we intend to measure the performance of the algorithm when dealing with textual datasets and compare this performance with some active pairwise constrained approaches for this kind of dataset.

References

Bade, K., Hermkes, M., and Nürnberger, A. (2007). User oriented hierarchical information organization and retrieval. In *ECML '07: Proceedings* *of the 18th European conference on Machine Learning*, pages 518–526, Berlin, Heidelberg. Springer-Verlag. Cited in page 3.

Basu, S., Banerjee, A., and Mooney, R. J. (2002). Semi-supervised clustering by seeding. In *ICML '02: Proceedings of the Nineteenth International Conference on Machine Learning*, pages 27–34, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. Cited in page 1.

Basu, S., Banerjee, A., and Mooney, R. J. (2004). Active semi-supervision for pairwise constrained clustering. In *SDM '04: Proceedings of the 2004 SIAM International Conference on Data Mining*, pages 333–344, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics. Cited in page 1.

Bilenko, M., Basu, S., and Mooney, R. J. (2004). Integrating constraints and metric learning in semi-supervised clustering. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, pages 81–88, New York, NY, USA. ACM. Cited in page 1.

Böhm, C. and Plant, C. (2008). Hissclu: a hierarchical density-based method for semi-supervised clustering. In *EDBT '08: Proceedings of the 11th international conference on Extending database technology*, pages 440–451, New York, NY, USA. ACM. Cited in page 3.

Daniels, K. and Giraud-Carrier, C. (2006). Learning the threshold in hierarchical agglomerative clustering. In *ICMLA '06: Proceedings of the 5th International Conference on Machine Learning and Applications*, pages 270–278, Washington, DC, USA. IEEE Computer Society. Cited in page 3.

Dasgupta, S. and Ng, V. (2010). Which clustering do you want? inducing your ideal clustering with minimal feedback. *Journal of Artificial Intelligence Research*, 39:581–632. Cited in page 1.

Davidson, I. and Ravi, S. S. (2009). Using instance-level constraints in agglomerative hierarchical clustering: theoretical and empirical results. *Data Mining and Knowledge Discovery*, 18(2):257–282. Cited in pages 3 and 8.

Eick, C. F., Zeidat, N., and Zhao, Z. (2004). Supervised clustering: Algorithms and benefits. In *ICTAI '04: Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence*, pages 774–776, Washington, DC, USA. IEEE Computer Society. Cited in page 1.

Gil-García, R. and Pons-Porrata, A. (2010). Dynamic hierarchical algorithms for document clustering. *Pattern Recognition Letters*, 31(6):469 – 477. Cited in page 8.

Huang, R. and Lam, W. (2009). An active learning framework for semi-supervised document clustering with language modeling. *Data and Knowledge Engineering*, 68(1):49–67. Cited in page 1.

Huang, Y. and Mitchell, T. M. (2006). Text clustering with extended user feedback. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 413–420, New York, NY, USA. ACM. Cited in page 1.

Huang, Y. and Mitchell, T. M. (2008). Exploring hierarchical user feedback in email clustering. In *EMAIL '08: Proceedings of the Workshop on Enhanced Messaging - AAAI 2008*, pages 36–41. AAAI Press. Cited in page 1.

Jain, A. K. and Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA. Cited in pages 2 and 8.

Kestler, H. A., Kraus, J. M., Palm, G., and Schwenker, F. (2006). On the effects of constraints in semi-supervised hierarchical clustering. In *Artificial Neural Networks in Pattern Recognition*, pages 57–66. Springer-Verlag. Cited in page 3.

Klein, D., Kamvar, S. D., and Manning, C. D. (2002). From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *ICML '02: Proceedings of the Nineteenth International Conference on Machine Learning*, pages 307–314, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. Cited in pages 2, 3, 5, and 12.

Kumar, N., Kummamuru, K., and Paranjpe, D. (2005). Semi-supervised clustering with metric learning using relative comparisons. In *ICDM '05: Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 693–696, Washington, DC, USA. IEEE Computer Society. Cited in page 1.

Larsen, B. and Aone, C. (1999). Fast and effective text mining using linear-time document clustering. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 16–22, New York, NY, USA. ACM. Cited in page 8.

Miyamoto, S. and Terami, A. (2010). Semi-supervised agglomerative hierarchical clustering algorithms with pairwise constraints. In *FUZZ'10: 2010 IEEE International Conference on Fuzzy Systems (FUZZ)*, pages 1 –6. Cited in page 1.

Settles, B. (2009). Active learning literature survey. Computer Sciences

Technical Report 1648, University of Wisconsin–Madison. Cited in page 1.

Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5:3–55. Cited in page 8.

Vu, V.-V., Labroche, N., and Bouchon-Meunier, B. (2010). Boosting clustering by active constraint selection. In *ECAI '10: Proceeding of the 19th European Conference on Artificial Intelligence*, pages 297–302, Amsterdam, The Netherlands, The Netherlands. IOS Press. Cited in page 1.

Wagstaff, K. and Cardie, C. (2000). Clustering with instance-level constraints. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1103–1110, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. Cited in pages 1 and 8.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83. Cited in page 9.