Clustering With Constraints Using Graph Based Approach

Haytham Elghazel[1], Khalid Benabdeslem[1] and Alain Dussauchoy[1]
[1] LIESP (ex. PRISMa) Laboratory, Claude Bernard University of Lyon I, France
{elghazel,kbenabde,dussauchoy}@bat710.univ-lyon1.fr

1 Introduction and Motivation

Clustering can be considered as the most important *unsupervised learning* problem which deals with finding a *structure* in a collection of unlabeled data. To this end, it conducts a process of organizing objects into groups whose members are similar in some way and dissimilar to those of other groups [1]. While this process yields in an entirely unsupervised manner, additional background information (namely constraints) are available in some domains and must be considered in the clustering solutions. These latter vary from the user and the domain but we are usually interested to the use of background information in the form of instance-level *must-link* and *cannot-link* constraints. A *must-link constraint* enforces that two instances must be placed in the same cluster while a *cannot-link constraint* enforces that two instances must not be placed in the same cluster.

Setting these constraints requires some modifications in the clustering algorithms which is not always feasible. Many authors investigated the use of constraints in clustering problem. In [2], the authors have proposed a modified version of COBWEB clustering algorithm that uses background information about pairs of instances to constrain their cluster placement. Equally, a recent work [3] has looked at extending the ubiquitous k-Means algorithm to incorporate the same types of *instance-level hard constraints* (*must-link* and *cannot-link*).

Recently, we have proposed a new clustering approach [4] based on the concept of *b-coloring* of a graph [5]. It exhibits more important clustering features and enables to build a fine partition of the data set (*numeric* or *symbolic*) in clusters when the number of clusters is not specified beforehand.

A *graph b-coloring* is the assignment of colors (clusters) to the vertices of the graph such that

(i) no two adjacent vertices have the same color (*proper coloring*),

(ii) for each color there exists at least one *dominating vertex* which is adjacent to all the other colors. This specific vertex reflects the properties of the class and also guarantees that the class has a distinct separation from all other classes of the partitioning.

In this paper, we are interested in ways to integrate background information into the *b-coloring based clustering* algorithm. The proposed algorithm which we will refer to as COP-*b-coloring* (for *constraint portioning b-coloring*) is evaluated against benchmark data sets and the results of this study indicate the effectiveness of the *instance-level hard constraints* to offer real benefits (accuracy and runtime) for clustering problem.

2 *COP-b-coloring* Algorithm: A new approach

This section is devoted to discuss our *b-coloring* based clustering algorithm. In the sequel, we define two forms of instance-level hard constraints we are using. We then show our investigation to incorporate this kind of constraints into a *b-coloring clustering algorithm*.

2.1 The Constraints

Let $\boldsymbol{X}=\{x_1,...,x_n\}$ denotes the given set of instances which must be partitioned such that the number of clusters is not given beforehand. In the context of clustering algorithms, instance-level constraints are a useful way to express a priori knowledge that constrains a placement of instances into clusters. In general, constraints may be derived from partially labeled data or from background knowledge about the domain of real data set. We consider the clustering problem of the data set $\boldsymbol{X}$ under the following types of constraints.

- *Must-Link* constraints denoted by ML(x_i,x_j) indicates that two instances x_i and x_j must be in the same cluster.
- *Cannot-Link* constraints denoted by CL(x_i,x_j) indicates that two instances x_i and x_j must not be in the same cluster.
- *Transitively derived* Instance-Level constraints from:
  - ML(x_i,x_j) and ML(x_j,x_k) imply ML(x_i,x_k),
  - ML(x_i,x_j), ML(x_k,x_l) and CL(x_i,x_k) imply both CL(x_i,x_l) and CL(x_j,x_k).

2.2 The Proposed Algorithm

In the remainder of this section, we describe the constrained *b-coloring* clustering approach called COP-*b-coloring* (for *constraint portioning b-coloring*). The algorithm takes in a *data set* $\boldsymbol{X}=\{x_1,...,x_n\}$, a *pairwise dissimilarity table* $D=\{d_{j,j'} | x_j,x_{j'} \in \boldsymbol{X}\}$, a full set of *must-link constraints* (both directly and transitively) denoted by $\boldsymbol{Con_=}$, and a full set of *cannot-link constraints* (both directly and transitively) denoted by $\boldsymbol{Con_{\neq}}$. It returns a partition $\boldsymbol{P}=\{C_1,C_2,..,C_k\}$ of the data set $\boldsymbol{X}$ that satisfies all specified constraints and the clustering quality (the number of clusters in not given beforehand).

Based on D, the data set $\boldsymbol{X} = \{x_1,...,x_n\}$ to be clustered can be conceived as an undirected complete edge-weighted graph $G = (\boldsymbol{V}, \boldsymbol{E})$, where $\boldsymbol{V} = \{v_1,...,v_n\}$ is the vertex set and $\boldsymbol{E} = \boldsymbol{V} \times \boldsymbol{V}$ is the edge set. Vertices in G correspond to instances (vertex v_i for instances x_i), edges represent neighborhood relationships, and edge-weights reflect dissimilarity between pairs of linked vertices. A common informal definition states that "a cluster is a set of entities which are *similar*, and entities from different clusters are not *similar*". Hence, the edges between two vertices within one cluster should be small weighted (denoting weak dissimilarity), and those between vertices from two clusters should be large weighted (high dissimilarity). The clustering problem is hence formulated as a graph *b-coloring* problem. The *b-coloring* of such a *complete graph* is not interesting for the clustering problem. Indeed, the *trivial partition* is returned where each cluster (*color*) is assumed to contain one and only one instance (*vertex*). Consequently and in order to incorporate the *instance-level constraints* into the clustering problem, our clustering approach requires to construct a *non-complete graph* that will be presented to the *b-coloring* algorithm in [4]. For that, the following definition is introduced:

Definition 1 A *composite vertex* $\boldsymbol{v'}$ is a subset of instances such that all pairs among these instances appear together in $\boldsymbol{Con_=}$. As an illustration, the two *must-link constraints* ML(x_i,x_j) and ML(x_j,x_k) transitively imply ML(x_i,x_k). Thus, both must-link constraints can be viewed as a single one namely ML(x_i,x_j,x_k). A composite vertex $\boldsymbol{v'}$ is hence identified as a subset $\{x_i,x_j,x_k\}$.

The construction of the *non-complete graph* $G'=(\boldsymbol{V'},\boldsymbol{E'})$ is now formulated using the following instructions:

- Transform the full set of *must-link constraints* $\boldsymbol{Con_=}$ on a *composite vertex* set $\boldsymbol{V_1'} =\{v'_1,...,v'_m\}$. The composite vertices in $\boldsymbol{V_1'}$ are pairwise disjointed. In the other hand, the remaining r instances ($\boldsymbol{X}-\cup_{i=1..m} v'_i$) which are not involved in any *must-link constraint* are affected to new r composite vertices $\boldsymbol{V_2'} =\{v'_{m+1},...,v'_r\}$. Finally $\boldsymbol{V_1'}$ and $\boldsymbol{V_2'}$ are combined into $\boldsymbol{V'}=\{v'_1,...,v'_r\}$ where $r<n$.
- Using the full set of *cannot-link constraints*, for any two composite vertices v'_i and v'_j in $\boldsymbol{V'}$, the edge (v'_i, v'_j) is in $\boldsymbol{E'}$ if there is at least one instance x_i in v'_i and another x_j in v'_j appear together in a *cannot-link constraint* (*i.e.* $\exists\, x_i \in v'_i\ \exists\, x_j \in v'_j$ ***such that*** CL(x_i,x_j)).
- Using a dissimilarity threshold value θ chosen among the dissimilarity table D, for any two composite vertices v'_i and v'_j in $\boldsymbol{V'}$, the edge $(v'_i, v'_j) \in \boldsymbol{E'}$ if there $\exists\, x_i \in v'_i\ \exists\, x_j \in v'_j$ ***such that*** $D(x_i,x_j)=d_{ij} >\theta$.

The data set $\boldsymbol{X}$, the full set of *must-link constraints* $\boldsymbol{Con_=}$, and the full set of *cannot-link constraints* $\boldsymbol{Con_{\neq}}$ are now summarized within an *undirected non-complete graph* $G'=(\boldsymbol{V'},\boldsymbol{E'})$. The main idea consists to apply the *b-coloring based clustering algorithm* [4] on G'. It consists of two steps: 1) initializing the colors of vertices with maximum number of colors, and 2) removing colors without any dominating vertex using a *greedy procedure*. Indeed, the goal is to give an assignment of colors (clusters) to the vertices (*i.e.* a composite vertices) of G' so that no two adjacent vertices have the same color, and that for each color class, at least one *dominating vertex* is adjacent to at least one vertex of every other color sets. The color of each composite vertex is then assigned to its members.

Proposition 1 The partition returned by the *b-coloring based clustering algorithm* satisfies all specified constraints (both *must-link* and *cannot-link*).

Proof Each composite vertex consists of instances such that all pairs among these instances appear together in $\boldsymbol{Con_=}$. The b-coloring algorithm affects a color for each composite vertex which is then assigned to its members. Consequently, the vertices appear in a *must-link constraint* will be placed in the same cluster (color). Thus, the given partition satisfies all *must-link constraints*. Each *cannot-link constraint* (among $\boldsymbol{Con_{\neq}}$) between two instances (x_i,x_j) is transformed as an edge between their composite vertices in G'. According to the property of the b-coloring of G', the colors of two adjacent vertices are different. Thus, x_i and x_j can never appear in the same cluster. Therefore, the given partition satisfies all *cannot-link constraints*.

Proposition2 The incorporation of the instances-level constraints decreases the runtime of the clustering algorithm.

Proof The *b-coloring based clustering algorithm* in [4] generates the *b-coloring* of any graph G (associated with a threshold value θ) in $O(n^2\Delta)$ where n is the number of vertices (instances). In our case, the incorporation of the all instance-level constraints transforms the initial graph G (with n vertices) into a graph G' (with r vertices) where $r<n$. Therefore, the incorporation of the *instances-level constraints* allow to decrease the complexity of the clustering algorithm to $O(r^2\Delta)$.

The clustering algorithm is iterative and performs multiple runs, each of them increasing the value of the dissimilarity threshold θ. Once all threshold values passed, the algorithm provides the optimal partitioning (corresponding to one threshold value θ_o) which maximizes *Dunn's generalized index* ($Dunn_G$) [6]. $Dunn_G$ is designed to offer a compromise between the *intercluster separation* and the *intracluster cohesion*. So, it is the more appropriated to partition data set in *compact* and *well-separated* clusters.

As an illustration, let suppose the data set related to the weighted dissimilarity matrix D in Table 1 and the following constraints: ML(C,E) and CL(A,C) which imply CL(A,E). Fig. 1 shows the *non-complete graph* with dissimilarity threshold θ=0.1 for Table 1 where the thick edge denotes the *cannot-link constraint*. Fig. 2

illustrates the *b-coloring algorithm* performed on the graph in Fig. 1. The vertices with the same color (shape) are grouped into the same cluster. Therefore, {**A**}, {**C**,**E**}, {**B**,**F**} and {**D**} are the clusters. In this example all vertices are dominating.

x_i	A	B	C	D	E	F
A	0					
B	0.20	0				
C	0.10	0.20	0			
D	0.20	0.20	0.25	0		
E	0.10	0.20	0.10	0.05	0	
F	0.40	0.075	0.15	0.15	0.15	0

Table 1. A dissimilarity Matrix

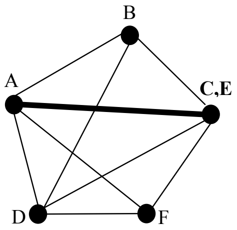

Fig. 1. A *non-complete edge-weighted graph* with *θ =0.1* for data in Table 1.

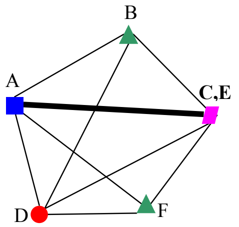

Fig. 2. A *b-coloring* of the graph in Fig. 1.

3 Experimental Results

In this section, we illustrate our algorithm's performance on several standard UCI data sets [7].

3.1 Evaluation metric

The used UCI data set includes *class information* (label) for each data instance. Since the objective was to perform an unsupervised classification that correctly identifies the underlying *classes* in the given data, we use the predefined *class labels* for the evaluation step. Consequently, as used in [2,3], our evaluation will be based on a label matching scheme called *Rand index* which concerns the *clustering accuracy*.

The returned partition (the *b-coloring* one) will be considered as a relation on the instances: for each pair of instances, they have either the same label or different ones. For a data set with n instances, there are $n(n-1)/2$ unique pairs of instances (x_i,x_j), and thus there are $n(n-1)/2$ pairwise decisions reflected in our returned partition. The *Rand index* [8] is defined as:

$$Rand(P,P') = Accuracy = \frac{a+b}{n\times(n-1)/2} \tag{1}$$

where

- P is the correct partition which is produced using the predefined class label.
- P' is the returned partition through the COP-*b-coloring* algorithm.
- a and b are the correct decisions. a is the number of decisions where x_i and x_j are in the same cluster for the partitions P and P'. b is the number of decisions where x_i and x_j are placed in different cluster for P and P'.

In order to examine the effectiveness of the COP-*b-coloring* algorithm, our experimental methodology follows the same principle used in [2,3]. For each data set, the main idea is to generate a number of artificial constraints and compute the *accuracy improvement* as more constraints are included into the COP-*b-coloring* algorithm. The constraints generation is given as follows: for each constraint, we randomly select two instances from the data set and check their labels. If they have the same label, we generate a must-link constraint. Otherwise, we generate a cannot-link constraint.

For an interesting assess of the learning improvements gained with the *COP-b-coloring algorithm*, we try to examine its ability to generalize the constraint information to the unconstrained instances. Thus, we propose to compute, aside the overall accuracy, the one on a *held-out test set* (a subset of data set composed of instances that are not directly or transitively affected by the constraints). The Both evaluation metrics (*overall accuracy* and *held-out test set accuracy)* are determined as the average results given from 100 trials conducted on each used data set.

3.2 Results

We report here our experiments using three relevant benchmark data sets chosen from UCI database [7]. We suppose that the number of cluster k is not given beforehand. Thus, the clustering algorithm was required to select the best value of k using the *Dunn*$_G$ *index*.

3.2.1. soybean data set contains 47 instances with 35 features and 4 output classes. The Figure 3 provides the clustering accuracy with a varying number of directly added constraints from 5 to 100. It gives also the number of cluster identified at each step. Without any constraints, the *graph b-coloring clustering algorithm* achieves an accuracy of 84%. The overall accuracy reaches 100% after 30 random constraints which attain 5.4% in isolation. Likewise, held-out accuracy improves and achieving 100% with 30 constraints. Then, we deduce that incorporating 30 constraints achieves a 16% increase in accuracy. Moreover, our algorithm produces best results than COP-COBWEB and COP-KMEANS which achieve a held accuracy respectively of 96% and 98% for 100 random constraints[1].

3.2.2. tic-tac-toe data set contains 100 instances, each described by 9 categorical attributes. The instances were also classified into two classes. COP-b-coloring starts at 48% accuracy with no constraints, reaching an overall accuracy of 95% and a held-out of 82% with 500 random constraints (56% for COP-KMEANS and 49% for COP-COBWEB) and yielding an improvement of 34% over the baseline (*c.f.* Fig. 4). The set of 500 random constraints achieves 70% accuracy before any clustering occurs.

3.2.3. cleve data set is very interesting due to its real and mixture appearance. It consists of 303 instances of heart disease (generated at the Cleveland Clinic on 1988) with 13 features. There are 5 numeric and 8 categorical attributes. The instances were also classified into two classes each class is either healthy (buff) or with heart-disease (sick). In the absence of constraints, the *graph b-coloring clustering algorithm* achieves an accuracy of 50%. After incorporating 500 constraints the overall accuracy reaches 89%. Here, 500 random constraints achieve 54% accuracy before any clustering occurs. Held-out accuracy climbs to 66% yielding an improvement of 16% over the baseline (*c.f.* Fig. 5).

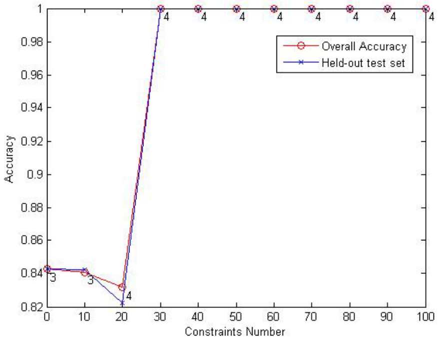

Fig. 3. *COP-b-coloring* results on soybean.

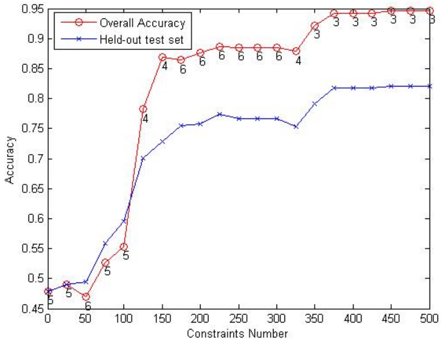

Fig. 4. *COP-b-coloring* results on tic-tac-toe.

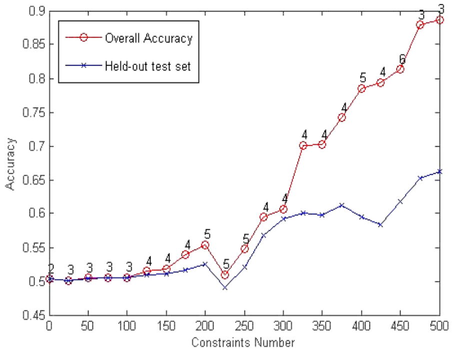

Fig. 5. *COP-b-coloring* results on cleve.

References

[1] Jain, A.K., M.N. Murty, and P.J. Flynn: Data Clustering: A Review. In: *ACM Computing Surveys*, Vol. 31, (1999), pp. 264-323.

[2] Wagsta, K., C. Cardie: Clustering with instance-level constraints. In Proceedings of the 17th International Conference on Machine Learning, (2000). pp. 1103-1110.

[3] Wagsta, K. et *al.*: Constrained K-means Clustering with Background Knowledge. In Proceedings of the 18th International Conference on Machine Learning, (2001). pp. 577-584.

[4] Elghazel, H. et *al.*: A new clustering approach for symbolic data and its validation: Application to the healthcare data. In F.Esposito et al.(Eds), editor, ISMIS2006 (Springer Verlag LNAI 4208) ,(2006), pp. 473–482.

[5] Irving, W. and D. F. Manlove: The b-chromatic number of a graph. *Discrete Applied Mathematics,* Vol. 91, (1999), pp. 127-141.

[6] Kalyani, M. and M. Sushmita: Clustering and its validation in a symbolic framework. *Pattern Recognition Letters*, 24(14), (2003), pp. 2367-2376.

[7] Blake, C.L. and C.J. Merz: *UCI repository of machine learning databases*. University of California, Irvine, Dept. of Information and Computer Sciences. Available from http://www.ics.uci.edu/ ~mlearn/MLRepository.html, (1998).

[8] Rand, W. M.: Objective criteria for the evaluation of clustering methods. In: *Journal of the American Statistical Association*, Vol. 66, (1971), pp. 846-850.

[1] We note that the results of COP-COBWEB and COP-KMEANS algorithms are given from [3] and not be reproduced in this paper.