Data Science and Engineering (2018) 3:359–378
https://doi.org/10.1007/s41019-018-0080-6

Evolutionary Active Constrained Clustering for Obstructive Sleep Apnea Analysis

Son T. Mai[1] · Sihem Amer-Yahia[3] · Sébastien Bailly[1] · Jean-Louis Pépin[1] · Ahlame Douzal Chouakria[1] · Ky T. Nguyen[1] · Anh-Duong Nguyen[2]

Received: 22 August 2018 / Accepted: 20 October 2018 / Published online: 7 November 2018

Abstract
We introduce a novel interactive framework to handle both instance-level and temporal smoothness constraints for clustering large longitudinal data and for tracking the cluster evolutions over time. It consists of a constrained clustering algorithm, called *CVQE+*, which optimizes the clustering quality, constraint violation and the historical cost between consecutive data snapshots. At the center of our framework is a simple yet effective active learning technique, named *Border*, for iteratively selecting the most informative pairs of objects to query users about, and updating the clustering with new constraints. Those constraints are then propagated inside each data snapshot and between snapshots via two schemes, called *constraint inheritance* and *constraint propagation*, to further enhance the results. Moreover, a historical constraint is enforced between consecutive snapshots to ensure the consistency of results among them. Experiments show better or comparable clustering results than state-of-the-art techniques as well as high scalability for large datasets. Finally, we apply our algorithm for clustering phenotypes in patients with Obstructive Sleep Apnea as well as for tracking how these clusters evolve over time.

Keywords Semi-supervised clustering · Active learning · Interactive clustering · Incremental clustering · Temporal clustering · Obstructive Sleep Apnea

1 Introduction

In semi-supervised clustering, domain knowledge is typically encoded in the form of instance-level *must-link* and *cannot-link* constraints [11] for aiding the clustering process, thus enhancing the quality of results. Such constraints specify that two objects must be placed or must not be placed in the same clusters, respectively. Constraints have been successfully applied to improve clustering quality in real-world applications, e.g., identifying people from surveillance cameras [11] and aiding robot navigation [10]. However, current research on constrained clustering still faces several major issues described below.

Most existing approaches assume that we have a set of constraints beforehand, and an algorithm will use this set to produce clusters [3, 10]. Davidson et al. [8] show that the clustering quality varies significantly using different equi-size sets of constraints. Moreover, annotating constraints requires human intervention, an expensive and time consuming task that should be minimized as much as possible given the same expected clustering quality. Therefore, how to choose a *good* and *compact* set of constraints rather than

✉ Son T. Mai
mtson@univ-grenoble-alpes.fr

Sihem Amer-Yahia
sihem.amer-yahia@univ-grenoble-alpes.fr

Sébastien Bailly
sbailly@chu-grenoble.fr

Jean-Louis Pépin
JPepin@chu-grenoble.fr

Ahlame Douzal Chouakria
ahlame.douzal@univ-grenoble-alpes.fr

Ky T. Nguyen
trung-ky.nguyen@univ-grenoble-alpes.fr

Anh-Duong Nguyen
anh-duong.nguyen@irisa.fr

1 University of Grenoble Alpes, Grenoble, France

2 University of Rennes 1, Rennes, France

3 University of Grenoble Alpes, CNRS, Grenoble, France

randomly selecting them from the data has been the focus of many research efforts, e.g., [2, 32, 43].

Many approaches employ different *active learning* schemes to select the most meaningful pairs of objects and then query experts for constraint annotation [2, 32]. By allowing the algorithms to choose constraints themselves, we can avoid insignificant ones, and expect to have high quality and compact constraint sets compared to the randomized scheme. These constraints are then used as input for constrained clustering algorithms to operate. However, if users are not satisfied with the results, they must provide another constraint set and then start the clustering again, which is obviously expensive.

Other algorithms follow a *feedback* schema which does not require a full set of constraints in the beginning [7]. They iteratively produce clusters with their available constraints, show results to users, and get feedback in the form of new constraints. By iteratively refining clusters according to user feedback, the acquired results fit users' expectations better [7]. Constraints are also easier to select with an underlying cluster structure as a guideline, thus reducing the overall number of constraints and human annotation effort for the same quality level. However, exploring the whole data space for finding meaningful constraints is also a non-trivial task for users.

To reduce human effort, several methods incorporate *active learning* into the feedback process, e.g., [17, 18, 32, 43]. At each iteration, the algorithm automatically chooses pairs of objects and queries users for their feedback in terms of *must-link* and *cannot-link* constraints instead of leaving the whole clustering results for users to examine. Though these techniques are proven to be very useful in real-world tasks such as document clustering [17], they suffer from very high runtime since they have to repeatedly perform clustering and explore all $O(n^2)$ pairs of objects to generate queries to users each time.

In this work, we develop an efficient framework to cope with the above problems following the iterative active learning approach as in [17, 43]. However, instead of examining all pairs of objects, our technique, called *Border*, selects a small set of objects around cluster borders and queries users about the most uncertain pairs of objects. We also introduce a constraint inheritance approach based on the notion of μ-nearest neighbors for inferring additional constraints, thus further boosting performance. Finally, we revisit our approach in the context of evolutionary clustering [6]. Evolutionary clustering aims to produce high-quality clusters while ensuring that the clustering does not change dramatically between consecutive timestamps. This scheme is very useful in many application scenarios. For example, doctors want to better describe trajectories of chronic diseases over time by identifying progressive aggregation of complications (comorbidities) [19]. It is a crucial issue to identify and ideally to anticipate the occurrence of multimorbidity (i.e., the association of more than 2 chronic diseases). They may expect that existing groups do not change so much over time if there are minor changes in the data. However, the clustering process should be able to reflect the changes if there are significant differences in the new data. Therefore, we propose to formulate a temporal (longitudinal) smoothness constraint into our framework and add a time-fading factor to our constraint propagation.

1.1 Contributions

Our contributions are summarized as follows:

- We introduce a new algorithm CVQE+ that extends CVQE [10] with weighted must-link and cannot-link constraints and a new object assignment scheme.
- We propose a new algorithm, Border, that relies on active clustering and constraint inheritance to choose a small number of objects to solicit user feedback for. Beside the active selection scheme for pairs of objects, Border employs a constraint inheritance method for inferring more constraints, thus further enhancing the performance.
- We present an evolutionary clustering framework which incorporates instance-level and temporal smoothness constraints for temporal data. To the best of our knowledge, our algorithm is the first framework that combines active learning, instance-level and temporal smoothness constraints.
- Experiments are conducted for six real datasets to demonstrated the performance of our algorithms over state-of-the-art ones.

1.2 Extensions

In this paper, we extend the work in [27] for investigating clinical clusters in a use case of patients with Obstructive Sleep Apnea (OSA) [22]. OSA is a common sleep disorder and a major medical concern corresponding to repetitive upper airway collapse during sleep [22, 44]. OSA is known to be related to many serious health problems including cardiovascular and metabolic morbidity and mortality [22, 25, 34]. Both symptoms and comorbidities vary significantly at diagnosis but data regarding longitudinal clustering are scarce [21]. Thus, identifying subgroups of patients that share the similar disease characteristics such as comorbidities, indices of OSA severity, and symptoms is an important step for personalizing OSA treatments [1]. Recent approaches employ automatic clustering methods to detect specific phenotypes, e.g., Hierarchical clustering [1] and Latent Class Analysis (LCA) [44]. However, these techniques do not exploit domain knowledge to improve the

quality of results during the clustering process. Moreover, all existing techniques are based on static data such as the diagnosis visit data [1] or questionnaires [44], while patient responses are not static and change over the life span and particularly symptoms are hugely modified by available treatments, thus causing the changes in clusters. In other words, current available analysis were not designed to capture the evolvements of patients' phenotypes over time, allowing to characterize response to treatments and occurrence of comorbidities. To the best of our knowledge, our algorithm Border is the first approach which can deal with these problems while analyzing OSA patients. To summary, we extend Border as follows:

- We extend the data model by assigning for each data object a patient identification (or process ID) as its owner. These IDs will be used for finding overlapped patient clusters at different time frames.
- We revise the historical cost of the temporal smoothness scheme by considering all overlapped clusters from previous snapshots.
- We introduce a *temporal evolution graph* to capture the evolvements of patient clusters at different visit times.

1.3 Outline

The rest of the paper is organized as follows. We formulate the problem in Sect. 2. Our framework is described in Sect. 3. Experiments are presented in Sect. 4. Section 5 discusses related works. In Sect. 6, we present an application of our algorithm to track the evolution of groups of patients with sleep disorder symptoms. Section 8 concludes the paper.

2 Problem Formulation

Let $D = \{(d, t, id(d))\}$ be a set of $|D|$ vectors (objects) $d \in \mathbb{R}^p$ observed at time t associated with a process ID $pid(d)$ that generates d. Note that two objects u and v in D may be generated by the same process, i.e., $pid(u) = pid(v)$.

Let $S = \{(S_s, D_s, ts_s, te_s)\}$ be a set of preselected $|S|$ data snapshots. Each S_s starts at time ts_s, ends at time te_s and contains a set of objects $D_s = \{(d, t, pid(d)) \in D \mid ts_s \leq t < te_s \wedge \forall u, v \in D_s : pid(u) \neq pid(v)\}$. In other words, all objects in D_s must be generated by different processes during the time frame $[ts_s, te_s)$. Two snapshots S_s and S_{s+1} may overlap but must satisfy the time order, i.e., $ts_s \leq ts_{s+1}$ and $te_s \leq te_{s+1}$.

For each snapshot S_s, let $ML_s = \{(x, y, w_{xy}) | (x, y) \in D_s^2\}$ and $CL_s = \{(x, y, w_{xy}) | (x, y) \in D_s^2\}$ be the set of *must-link* and *cannot-link* constraints of S_s with a degree of belief of $w_{xy} \in [0, 1]$. ML_s and CL_s can be empty.

2.1 Algorithms

In this paper, we focus on the problem of grouping objects in all snapshots into clusters in an active interactive feedback scheme as described in Sect. 1. To summary, our goals are (1) reduce the number of constraints thus reducing the constraint annotation costs (2) make the algorithm scale well with large datasets and (3) smooth the gap between clustering results of two consecutive snapshots, i.e., ensure temporal smoothness. The technical details of our methods will be described in Sect. 3.

2.2 Applications

We apply our algorithms for investigating the clinical groups of patients with Obstructive Sleep Apnea (OSAS) based on their medical visit records over time. The detailed study will be presented in Sect. 6.

3 Our Proposed Framework

Figure 1 illustrates our framework which relies on two algorithms, Border and CVQE+. Our framework starts with a small (or empty) set of constraints in each snapshot. Then, it iteratively produces clustering results and receives refined constraints from users in the next iterations. This process is akin to feedback-driven algorithms for enhancing clustering quality and reducing human annotation effort [7]. However, instead of passively waiting for user feedback as in [7], our algorithm, Border, *actively* examines the current cluster structure, selects β pairs of objects whose clustering labels are the least certain, and asks users for their feedback in terms of instance-level constraints. Examining all possible pairs of objects to select queries is time consuming due the quadratic number of candidates. To ensure scalability, Border limits its selection to a small set of most promising objects. When there are new constraints, instead of reclustering from scratch as in [17, 43], our algorithm, CVQE+, incrementally updates the cluster structures for saving computation times. We also aim to ensure a smooth transition between consecutive clusterings [6]. We additionally introduce two novel concepts: (1) the *constraint inheritance* scheme for automatically inferring more constraints inside each snapshot and (2) the *constraint propagation* scheme for propagating constraints between different snapshots. These schemes help significantly reduce the number of constraints that users must enter into the systems for acquiring a desired level of clustering quality by automatically adding more constrained based on the annotated ones. To the best of our knowledge, Border is the first framework that combines active learning, instance-level and temporal smoothness constraints.

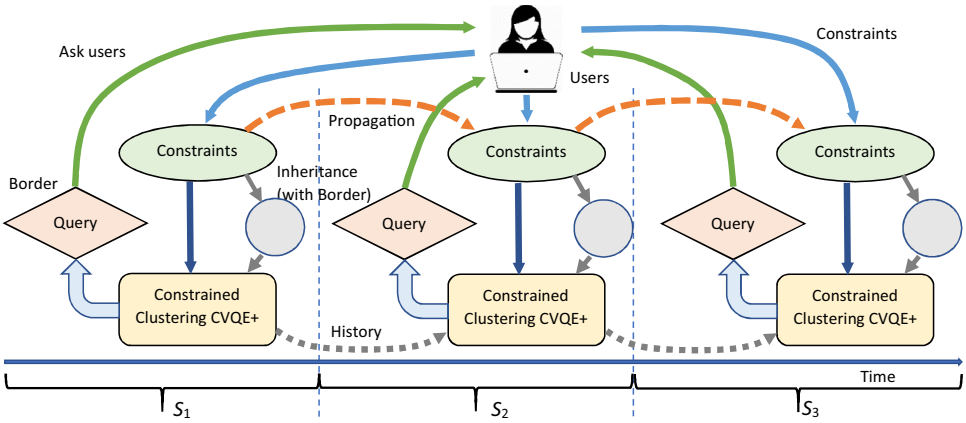

Fig. 1 Our active temporal clustering framework

3.1 Constrained Clustering Algorithm

For each snapshot S_s, we use constrained *k*means for grouping objects. Generally, any existing techniques such as MPCK-means [3], CVQE [10] or LCVQE [36] can be used. Here we introduce CVQE+, an extension of CVQE [10] to cope with weighted constraints, to do the task.

3.1.1 The New Algorithm CVQE+

Let $C = \{C_i\}$ be a set of clusters. The cost of C_i is defined as its vector quantization cost VQE_i and the constraint violation costs ML_i and CL_i (where $ML_i \subseteq ML$ and $CL_i \subseteq CL$ are the sets of *must-link* and *cannot-link* constraints related to C_i) as follows. Note that our ML_i cost is symmetric compared to [10].

$$Cost_{C_i} = Cost_{VQE_i} + Cost_{ML_i} + Cost_{CL_i} \tag{1}$$

$$Cost_{VQE_i} = \sum_{x \in C_i} (c_i - x)^2$$

$$Cost_{ML_i} = \sum_{(a,b) \in ML_i \wedge vl(a,b)} w_{ij}(c_i - c_{\pi(a,b,i)})^2$$

$$Cost_{CL_i} = \sum_{(a,b) \in CL_i \wedge vl(a,b)} w_{ij}(c_i - c_{\varphi(i)})^2$$

where $vl(a, b)$ is true for (a, b) that violates *must-link* or *cannot-link* constraints, c_i is the center of cluster C_i, $\pi(a, b, i)$ returns the center of clusters of a or b (not including cluster C_i), and $\varphi(i)$ returns the nearest cluster center of C_i. Note that $Cost_{ML_i}$ is symmetric compared to [10].

Taking the derivative of $Cost_{C_i}$, the new center of C_i is updated as:

$$\frac{dCost_{C_i}}{d_{c_i}} = \frac{dCost_{VQE_i}}{d_{c_i}} + \frac{dCost_{ML_i}}{d_{c_i}} + \frac{dCost_{CL_i}}{d_{c_i}}$$

$$c_i = \frac{\sum_{x \in C_i} x + \sum_{(a,b) \in ML_i \wedge vl(a,b)} w_{ij} c_{\pi(a,b,i)} + \sum_{(a,b) \in CL_i \wedge vl(a,b)} w_{ij} c_{\varphi(i)}}{|C_i| + \sum_{(a,b) \in ML_i \wedge vl(a,b)} w_{ij} + \sum_{(a,b) \in CL_i \wedge vl(a,b)} w_{ij}} \tag{2}$$

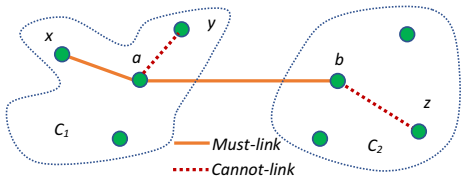

Fig. 2 Assigning a pair of constrained object (a, b) to clusters in CVQE+. All constraints starting and ending at a and b will be considered

For each constraint (a, b), CVQE+ assigns objects to clusters by examining all k^2 cluster combinations for a and b like CVQE. The major difference is that when we calculate the violation cost, we consider all constraints starting and ending at a and b instead of only the constraint (a, b) as in CVQE [9] or LCVQE [9], which is very sensitive to the cost change when some constraints share the same objects (changing these objects affects all their constraints) as illustrated in Fig. 2. The assigning cost for (a, b) will include the violation costs for (a, x), (a, y) and (b, z) as well. Thus, this scheme is expected to improve the clustering quality of CVQE+ compared to CVQE and LCVQE.

3.1.2 Complexity Analysis

Let n be the number of objects, m be the number of constraints, k be the number of clusters. CVQE+ has time complexity $O(rkn + rk^2m^2)$ which is higher than $O(rkn + rk^2m)$ of CVQE due to the fact that all related

constraints must be examined while assigning a constraint, where r is the number of iterations of the algorithm. Since k and m are constants, CVQE+ is thus has linear time complexity to the number of objects n. It also requires $O(n)$ space for storing objects and constraints.

3.2 Active Constraint Selection

We introduce an active learning method called Border for selecting pairs of objects and query users for constraint types. The general idea is examining objects lying around borders of clusters since they are the most uncertain ones and choosing a block of β pairs of objects to query users until the query budget δ is reached. Here, β and δ are predefined constants.

3.2.1 Active Learning with Border

To avoid examining all pairs of objects, Border chooses a subset of $m = min(O(\sqrt{n}), M)$ objects located at the boundary of the clusters as the main targets since they are the most uncertain ones, where M is a predefined constant (default as 100). This bound limits the number of pairs as a constant, thus reducing the number of pairs needed for examining in the subsequence steps. For each object a in cluster C_i, the border score of a is defined as:

$$bor(a) = \frac{(a - c_i)^2}{(a - c_{\varphi(i)})^2(1 + ml(a))(1 + cl(a))} \tag{3}$$

where $ml(a)$ and $cl(a)$ are the sums of weights of must and cannot-link constraints of a. Here, we favor objects that have fewer constraints for increasing constraint diversity. This also fits well with our constraint inheritance scheme. Moreover, by considering the distance to the second nearest cluster center $c_{\varphi(i)}$, we focus more on objects that are close to the boundaries of two clusters rather than ones that are far away from other clusters, which may not bring much benefit to clarify the groups. For each cluster C_i, we select $m|C_i|/n$ top objects based on their border score distribution in C_i. This can be done by building a histogram with $O(\sqrt{|C_i|})$ bins (a well-known rule of thumb for the optimal histogram bin) [4]. Then, objects are taken sequentially from the outermost bins. This scheme ensures that all clusters are considered based on their current sizes. Bigger clusters contribute more objects than smaller ones since their changes will more likely affect the final clustering result. Moreover, by using histogram bins, we give equal changes to objects within a bin since these objects might have the same importances for clarifying the clusters.

For each selected object a, we estimate the uncertainty of a w.r.t. the current clustering result as:

$$sco(a) = ent(\mu nn(a)) + \frac{vl(ml(a)) + vl(cl(a))}{ml(a) + cl(a) + 1} \tag{4}$$

where $ent(\mu nn(a))$ is the entropy of class labels of μ nearest neighbors of a and $vl(ml(a))$ and $vl(cl(a))$ are the sums of violated must-link and cannot-link constraints of a. A high $score(a)$ means that a is in high uncertain areas with different mixed class labels and a high number of constraint violations. And thus, it should be focused on.

We divide $m^2 = O(n)$ pairs of selected objects into two sets: the set of inside cluster pairs X and between cluster pairs Y, i.e., for all $(x, y) \in X : label(x) = label(y)$ and for all $(x, y) \in Y : label(x) \neq label(y)$. For a pair $(x, y) \in X$, it is sorted by $val(x, y) = \frac{(x-y)^2(1+sco(x))(1+sco(y))}{(1+ml(x)+cl(x))(1+ml(y)+cl(y))}$. For $(x, y) \in Y$, $val(x, y) = \frac{(x-y)^2(1+ml(x)+cl(x))(1+ml(y)+cl(y))}{(1+sco(x))(1+sco(y))}$. The larger val is, the more likely x and y belong to different clusters and vice versa. Moreover, in Y, we tend to select pairs with more related constraints to strengthen the current clusters, while we try to separate clusters in X by considering pairs with fewer related constraints. We choose top $\beta/2$ non-overlapped largest val pairs of X and top $\beta/2$ non-overlapped smallest pairs of Y in order to maximize the changes in clustering results (inside and between clusters). To be concrete, if a pair (a, b) was chosen, all pairs starting and ending with a or b will not be considered for enhancing the constraint diversity, which can help to bring up better performance. If all pairs are excluded, we select the remainder randomly.

We show β pairs to users to ask for the constraint type and add their feedback to the constraints set and update clusters until the total number of queries exceeds a predefined budget δ as illustrated in Fig. 1.

3.2.2 Constraint Inheritance in Border

For further reducing the number of queries to users, the general idea is to infer new constraints automatically based on annotated ones. Our inheritance scheme is based on the concept of μ nearest neighbors below.

Let h be the distance between an object p and its μ nearest neighbors. The influence of p on its neighbor x is formulated by a triangular kernel function $\phi_h(p, x)$ centered at p as in Fig. 3. Given a constraint (p, q, w_{pq}), for all $a \in \mu nn(p)$ and $b \in \mu nn(q)$, we add (a, b, w_{ab}) to the constraints set, where w_{ab} is defined as:

$$w_{ab} = w_{pq}\phi_h(p, a)\phi_h(q, b) \tag{5}$$

The general intuition is that the label of an object a tends to be consistent with its closest neighbors (which is commonly used in data classification such as the nearest neighbor classification [15]). This scheme is expected to increase the clustering quality, especially when combined with the active learning approach of the algorithm Border described above.

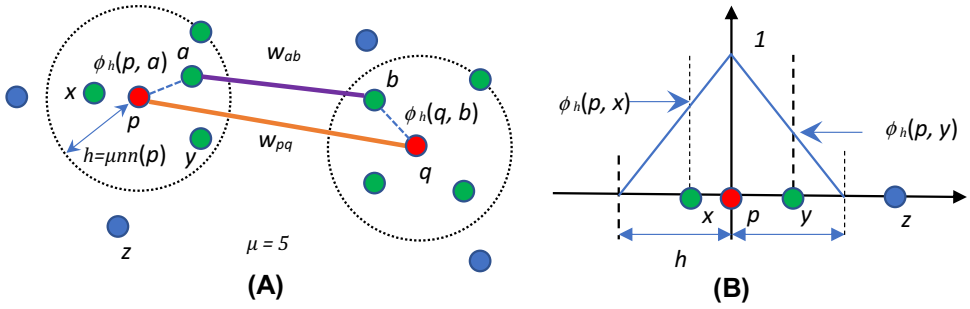

Fig. 3 **a** Constraint inheritance from (p, q) to (a, b). **b** The effect of the object b on its neighbors

During the inheritance scheme, if a pair of objects (a, b) is inherited from two constraints (c, d) and (e, f) with inherited weights w_1 and w_2, respectively, its weight and type are determined as follows:

$$w_{ab} = \begin{cases} max(w_1, w_2) & \text{if } type(c,d) = type(e,f) \\ |w_1 - w_2| & \text{otherwise} \end{cases} \quad (6)$$

where $type(c, d)$ is the constraint type of (c, d) (either *must-link* or *cannot-link*). And $type(a, b)$ is determined by $type(c, d)$ if $w_1 > w_2$ and vice versa. The general idea here is that if (a, b) is influenced by two constraints with different kinds, it will follow the one with the highest influence. Note that if (a, b) belongs to the main constraint set, we exclude it from the constraint inheritance scheme since it is annotated by users and thus it is confident.

3.2.3 Updating Clusters

At each iteration, instead of performing clustering again for producing the clustering result with the new set of constraints, we propose to update it incrementally for saving runtime. To do so, we only need to take the old cluster centers and update them following Eq. 1 with the updated constraints set. The intuition behind this is that new constraints are more likely to change clusters locally. Hence, starting from the current state might make the algorithm to converge faster, thus saving runtimes. In Sect. 4, we show that this updating scheme acquires the same quality but converges much faster than reclustering from scratch.

3.2.4 Complexity Analysis

Similarly to CVQE+, Border has $O(n)$ time and space complexity at each iteration and thus has $O(\delta n/\beta)$ time complexity overall, where δ is the budget limitation and β is the number of selected objects at each iteration described above.

3.3 Temporal Smoothness Constraints

The general idea of temporal smoothness [6] is that clusters not only have high quality in each snapshot but also do not change much between sequential time frames. It is useful in many applications where the transition between different snapshots is smoothed for consistency.

3.3.1 Temporal Smoothness

To ensure the smoothness, we re-define the cost of cluster C_i of snapshot S_s in Eq. 1 by enforcing a historical cost from its previous snapshot as follows:

$$TCost_{VQE_i} = (1 - \alpha) Cost_{VQE_i} + \alpha Hist(C_i, S_{s-1}) \quad (7)$$

where $Hist(C_i, S_{s-1})$ is the historical cost of cluster C_i between two snapshots S_s and S_{s-1} and α is a regulation factor to balance the current clustering quality and the historical cost. This cost keeps the new clusters do not deviate too much from ones from previous snapshot while performing clustering. We define the historical cost as follows:

$$Hist(C_i, S_{s-1}) = \frac{n}{k} \sum_{C_j \in S_{s-1}} \omega(C_i, C_j)(c_i - c_j)^2 \quad (8)$$

where $\omega(C_i, C_j) = \frac{|C_i \cap C_j|}{|C_i|}$ is the percentage of objects of C_j in C_i, c_i is the center of cluster C_i, n and k are the number of objects and number of clusters, respectively. The general idea is that if a cluster C_i consists of objects from several other clusters in S_{i-1}, its center will be moved toward these clusters according to their membership contributions in C_i and the distances between their centers to the center of C_i. Obviously if two clustering results are too different, indicated by high historical cost, the penalty will be higher thus forcing the algorithm to lower down the overall cost by creating clusters closer to those of the previous snapshot. The major difference between this scheme and the nearest centers

proposed in our previous works [27] is that each object inside a snapshot is assigned a unique Id. Two objects in two different snapshots may have the same Id as described in Sect. 2. These Ids can be used to check the overlap between two clusters in two snapshots. Thus, this scheme provides a better way to capture the smoothness between two clustering results since the nearest cluster C_j of C_i may actually contain only a few or no objects in C_i. Hence, moving C_i toward C_j as in [27] makes no sense anymore.

Taking the derivation of Eq. (7) as in Eq. (1), we can update the cluster centers as follows:

$$c_i = \frac{(1-\alpha)A + \alpha E}{(1-\alpha)B + \alpha F} \tag{9}$$

where A and B are, respectively, the numerator and the denominator given in Eq. 2 for updating clusters,

$$E = \frac{n}{k} \sum_{C_j \in S_{s-1}} \omega(C_i, C_j) c_j \tag{10}$$

and

$$F = \frac{n}{k} \sum_{C_j \in S_{s-1}} \omega(C_i, C_j) \tag{11}$$

3.3.2 Constraint Propagation

Whenever we have a new constraint (x, y, w_{xy}) in snapshot S_s, we propagate it to snapshots $S_{s'}$ where $s' > s$ if $pid(x), pid(y) \in S'$. The intuition is that if x and y are linked (either by must or cannot-link) in S_s, they are more likely to be linked in $S_{s'}$. Thus we add the constraint (x, y, w'_{xy}) to $S_{s'}$ where:

$$w'_{xy} = w_{xy} \frac{te_s - ts_{s'}}{te_{s'} - ts_s} \tag{12}$$

where $(te_s - ts_{s'})/(te_{s'} - ts_s)$ is a time-fading factor. This scheme helps to increase the clustering quality by putting more constraints into the clustering algorithm like the inheritance scheme. Since propagated constraints are not considered as user annotated ones, we treat it as non-confident constraints in our model and will not build offspring for them like those in the main constraint set described in the inheritance scheme above.

3.4 Temporal Evolutionary Graph

We propose to build a graph $G = (V, E)$, called the *temporal evolutionary graph*, to keep track of the relationships among clusters. In G, each node v is a cluster and each edge (u, v) represents the similarity between two clusters u and v that belong to two consecutive snapshots.

Given two clusters $C_i \in S_s$ and $C_j \in S_{s+1}$, we define the similarity between C_i and C_j as follows:

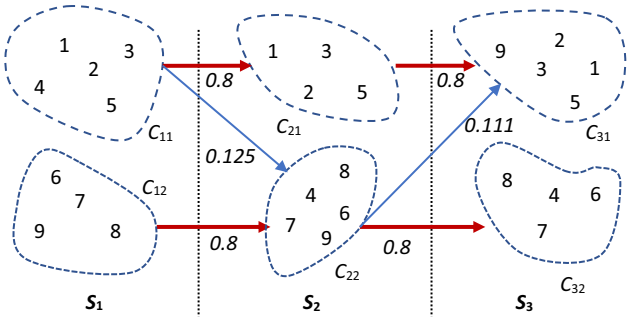

Fig. 4 The temporal evolutionary graph. The weight of each edge shows the overlap between two clusters. The red edge indicates two closest clusters between two snapshots

$$sim(C_i, C_j) = \frac{|C_i \cap C_j|}{|C_i \cap C_j|} \tag{13}$$

where $sim(C_i, C_j) = 1$ if C_i and C_j are identical and 0 if C_i and C_j are completely different. If $sim(C_i, C_j) > 0$, we put an edge (C_i, C_j) that connects C_i and C_j into G, indicating that they are related to each other.

Given a cluster $C_i \in S_s$, a cluster $C_j \in S_{i+1}$ is the closest one of C_i, denoted as $\xi(C_i, S_{s+1})$ if $C_j = max_{C_j \in S_{s+1}} sim(C_i, C_j)$, i.e., C_j is the one with highest cluster similarity to C_i.

Figure 4 illustrates a temporal evolution graph G with three snapshots. We have $sim(C_{11}, C_{21}) = 4/5 = 0.8$. There is no edge between C_{12} and C_{21} since they share nothing. And, C_{22} is the closest cluster of C_{12} (indicated by red edge). By following edges of G, we can keep track of the way clusters evolve over time. For example, from snapshot $S1$ to S_2, object 4 has changed its membership to an other different cluster. This scheme will be useful for us to study how cohorts (groups) of patients change over time and identify factors that cause these changes in our application scenario described in Sect. 6.

4 Experiments

Experiments are conducted on a workstation with 4.0Ghz CPU and 32GB RAM using Java. We use 6 datasets Iris, Ecoli, Seeds, Libras, Optdigits, and Wdbc acquired from the UCI archives.[1] The numbers of clusters k are acquired from the ground truths. Constraint queries are also simulated from the ground truths by adding a *must-link* if two objects have the same labels or a *cannot-link* if they have different labels. We use Normalized Mutual Information (NMI) [33]

[1] http://archive.ics.uci.edu/ml/.

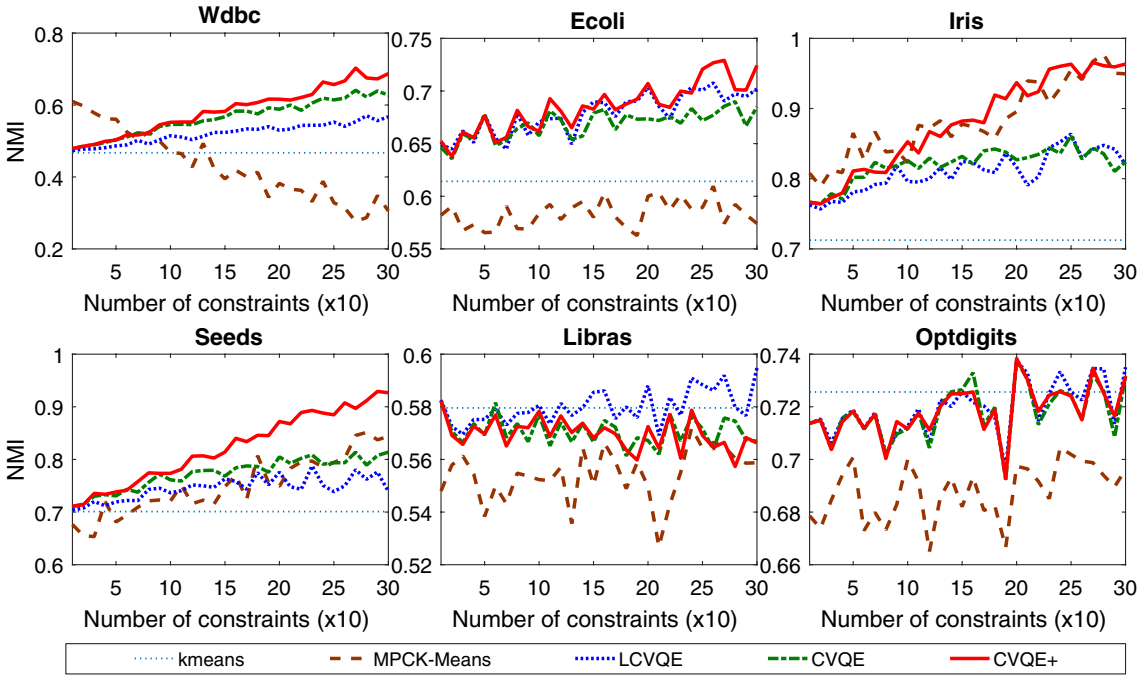

Fig. 5 Performance of CVQE+ compared to others

for assessing the clustering quality. NMI score is in [0,1] where 1 means a perfect clustering result compared to the ground truth and vice versa. All results are averaged over 10 runs.

4.1 Constrained Clustering

We study the performance of our constrained clustering method CVQE+ in comparisons with *k*means and state-of-the-art constrained clustering techniques such as MPCK-means [3], CVQE [10] and LCVQE [36].

4.1.1 Performance of CVQE+

Figure 5 shows comparisons among CVQE+ and existing techniques including *k*means, MPCK-means [3], CVQE [10] and LCVQE [36] over different sets of randomly selected constraints. CVQE+ consistently outperforms or acquires comparable results to CVQE and others for most datasets (except the Libras dataset where it is outperformed by LCVQE), especially when the number of constraints is large. This can be explained by the way CVQE+ assigns objects to clusters. By considering all related constraints while assigning cluster labels for objects, it can better optimize the overall cost function, thus leading to better clustering quality. Compared to its predecessor algorithm CVQE or LCVQE, it deals well with constraint overlap (constraints that share the same objects), which increases with the number of constraints. Note that when the constraint set is empty, CVQE+ produces clustering in the similar way with *k*-means. Thus, the clustering quality does not start from 0.

4.1.2 Noise Robustness

For studying the effect of noisy constraints on CVQE+, we randomly choose some constraints and change them from *must-link* to *cannot-link* and vice versa. Figure 6 shows the clustering quality of different algorithms w.r.t. the percentages of noisy constraints from 2 to 8% for real datasets. As we can see, for all algorithms, when the number of noisy constraints increases, the clustering quality decreases accordingly. However, CVQE+ tends to be more affected by noise than its related techniques CVQE and LCVQE. Though its point assignment scheme helps to increase the clustering quality as discussed above, it makes CVQE+ more sensitive to noise since one noisy constraint will affect the assignment cost for all of its related constraints. Nevertheless, in our experimented data, CVQE+ still acquires better (or equivalent) clustering results than CVQE and LCVQE under the same noisy conditions in most cases as seen in Fig. 6. However, developing a more effective algorithm to cope with noisy constraints is still an interesting target to pursue.

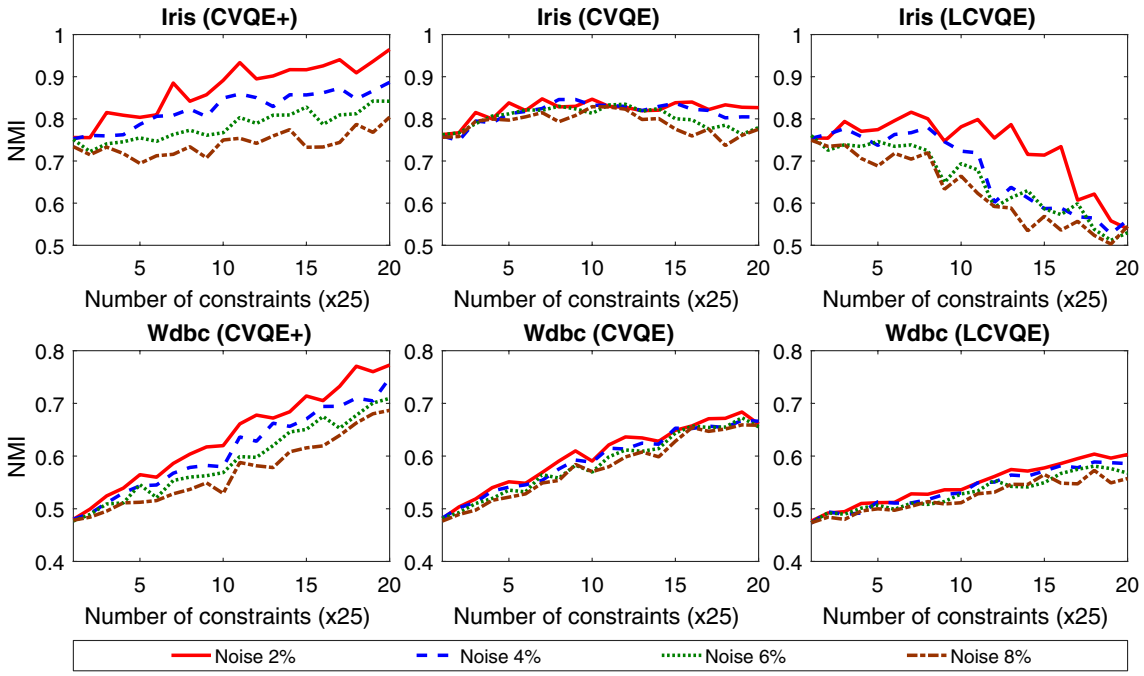

Fig. 6 Effect of noisy constraints on CVQE+ (left), CVQE (middle) and LCVQE (right) for the Iris and Wdbc datasets

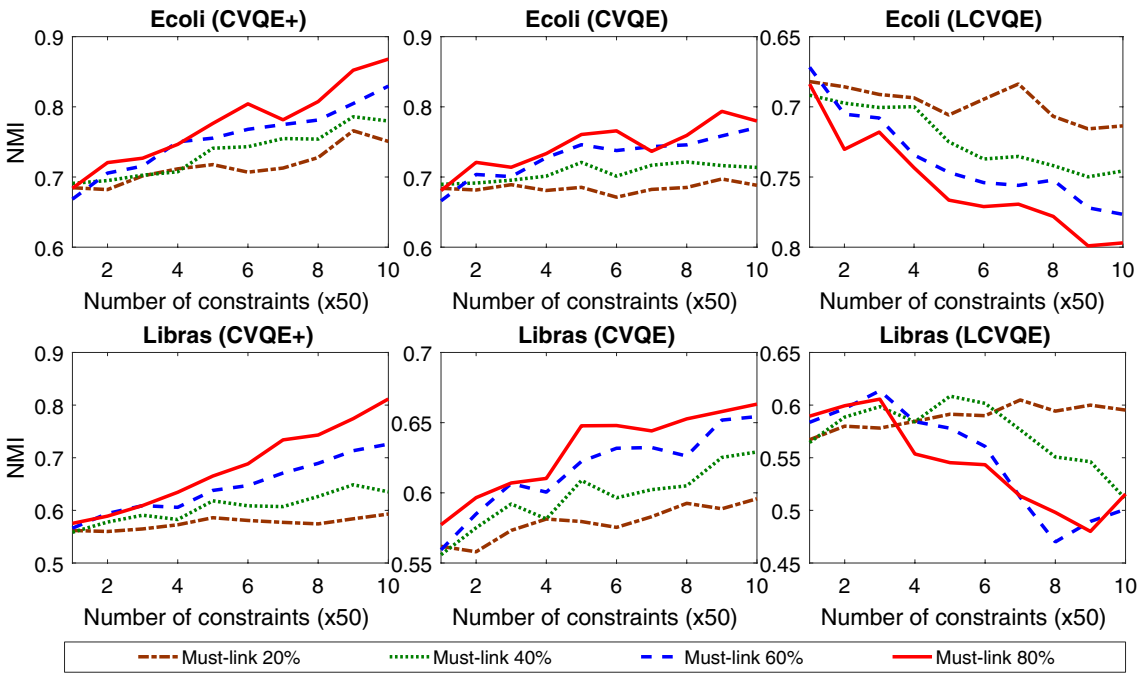

Fig. 7 Effect of constraint types on CVQE+ (left), CVQE (middle) and LCVQE (right) for the Ecoli and Libras datasets

4.1.3 Effect of Constraint Types

Figure 7 shows the performance of CVQE+ and its related techniques CVQE and LCVQE when the number of *must-link* constraints increases from 20 to 80% of the constraint sets. The clustering quality of CVQE+ and CVQE increases with the number of *must-link* constraints, while

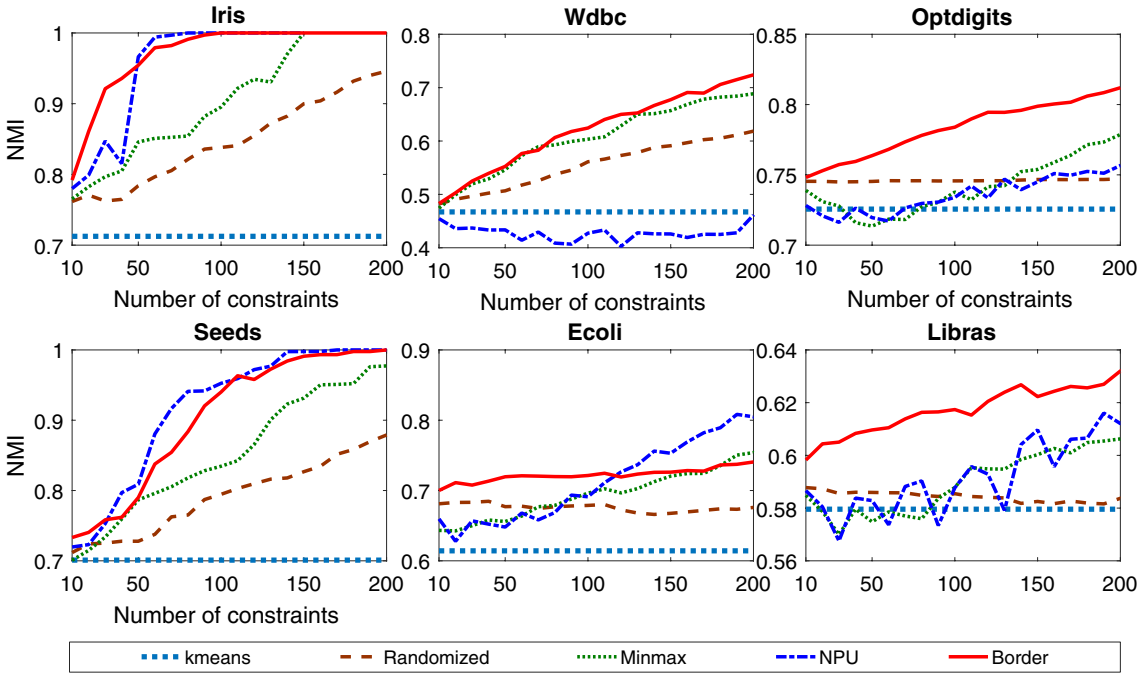

Fig. 8 Comparison among different active learning techniques

that of LCVQE decreases. This can be explained by the ways they calculate the constraint violation costs for the *must-link* and especially the *cannot-link* constraints. LCVQE treats violated *cannot-link* constraints more properly than CVQE and CVQE+. Thus, it deals well with higher number of those constraints.

4.2 Active Constraint Selection

We study the performance of Border in comparison with other state-of-the-art active learning techniques. Unless otherwise stated, the budget limitation δ is set to 200, the query size $\beta = 10$ and the neighborhood size $\mu = 4$.

4.2.1 Active Constraint Selection

Figure 8 shows comparisons between Border, NPU [43], Huang [43] (a modified version of [17] for working with non-document data), min–max [32], Explorer-consolidate [2], and a randomized method (Huang and Consolidate are removed from Fig. 8 for readability). Border acquires better results than others on Libras, Wdbc and Optdigits, comparable results on Iris and Ecoli. For the Seeds dataset, it is outperformed by NPU. The difference is because Border tends to strengthen existing clusters by fortifying both the cluster borders and inter connectivity for groups of objects rather than connecting a single object to existing components like NPU and Huang. Moreover, since it iteratively studies the clustering results for selecting constraints, it has better performance than non-iterative methods like Consolidate and min–max.

4.2.2 Runtime Comparison

For studying the runtime of Border on large-scale datasets, we create five synthetics datasets of sizes 2000–10,000 consisting of 5 Gaussian clusters and measure the time for acquiring 100 constraints. The results are shown in Fig. 9. Border is orders of magnitude faster than other methods in selecting pairs to query. For 1000 objects, it takes Border 0.1 s while NPU and min–max need 439.4 and 3.0 s, which is 4394 and 30 times slower than Border. For 10,000 objects, Border, NPU and min–max consumes

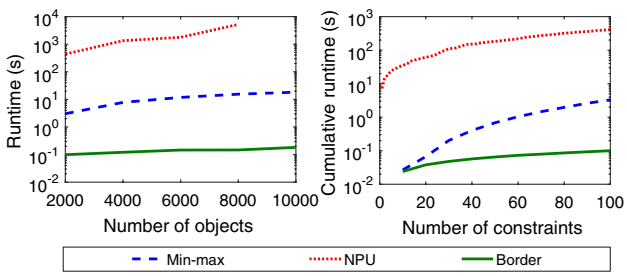

Fig. 9 Runtimes of different techniques

0.18, 5216.3 and 18.2 s, respectively. It is due to the fact that Border does not evaluate all pairs of objects at each iteration. Thus, it does much less works than others and faster. Besides, NPU and min–max are implemented in MATLAB which is slower than Border in Java. Nevertheless, the higher the number of objects and constraints, the higher the runtime differences. For 10,000 objects, Border is around 28,979.4 and 101.1 times faster than NPU and min–max, respectively. Hence, its runtime performance makes Border an effective technique to cope with very large datasets.

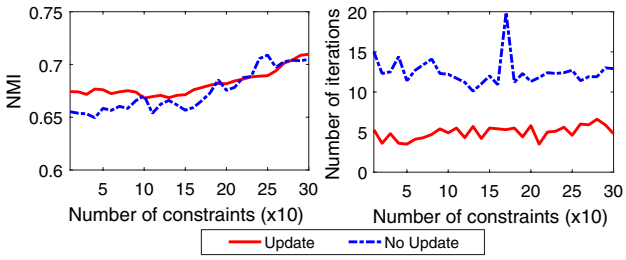

Fig. 10 Update versus fully reclustering for the Ecoli dataset

4.2.3 Cluster Update

Figure 10 shows the NMI and the number of iterations of our algorithm for the Ecoli dataset. The NMI scores are comparable, while it takes fewer iterations for our algorithm to converge in its update mode.

4.2.4 Effect of the Block Size β

Figure 11 shows the performance of Border when the query block size β varies from 10 to 30. As we can see, the smaller the value of β is, the better the performance of Border since the cluster structure is assessed more frequently, thus leading to better constraints to be selected at each iteration.

4.2.5 Effect of the Constraint Inheritance Scheme

Figure 12 shows the effect of the parameter μ on our algorithm Border via the inheritance scheme. Typically, its performance will increase with μ until it reaches the peak and then decreases as shown for the dataset Iris. This can be explained by the neighborhood influence scheme of Border. When μ is large enough, the number of wrong constraints will be increased, thus lower down the performance

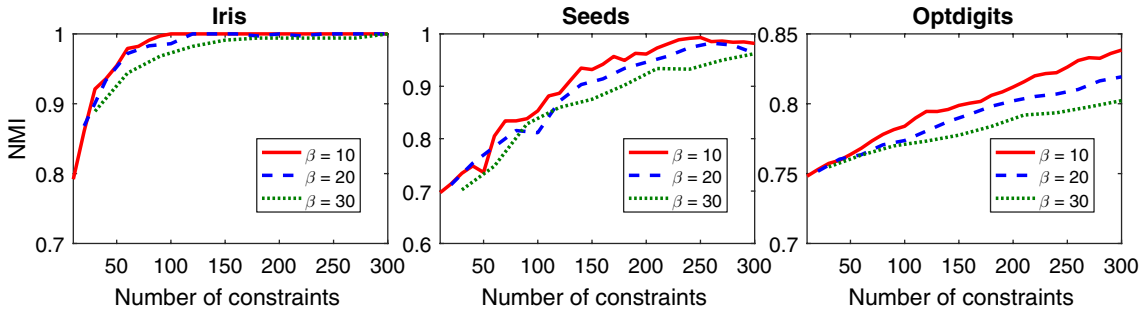

Fig. 11 The effect of the query block size β on the performance of Border

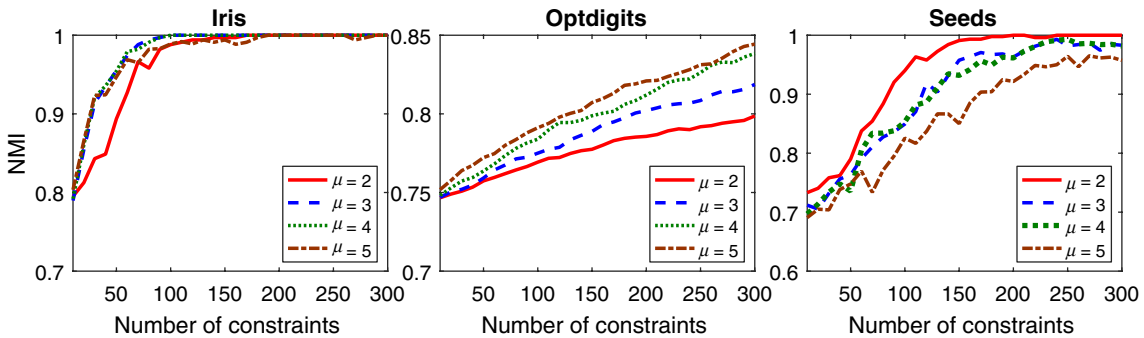

Fig. 12 The effect of the neighborhood size μ on the performance of Border

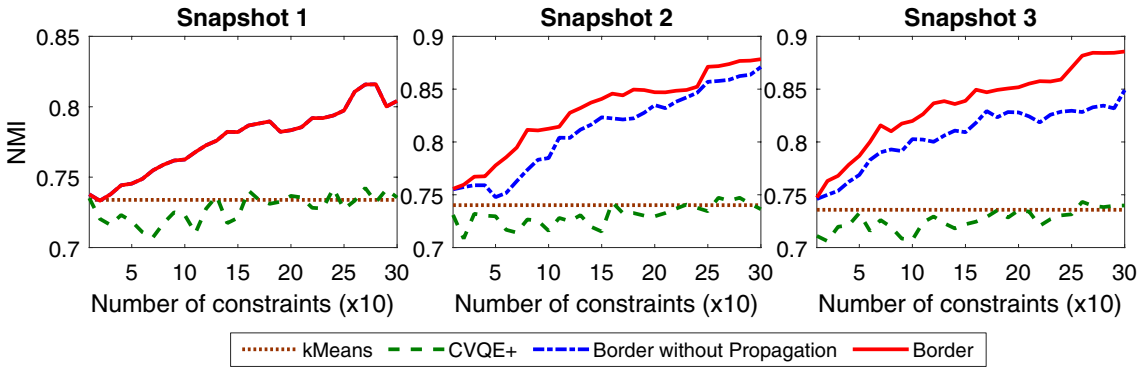

Fig. 13 Temporal clustering on the dataset Optdigits

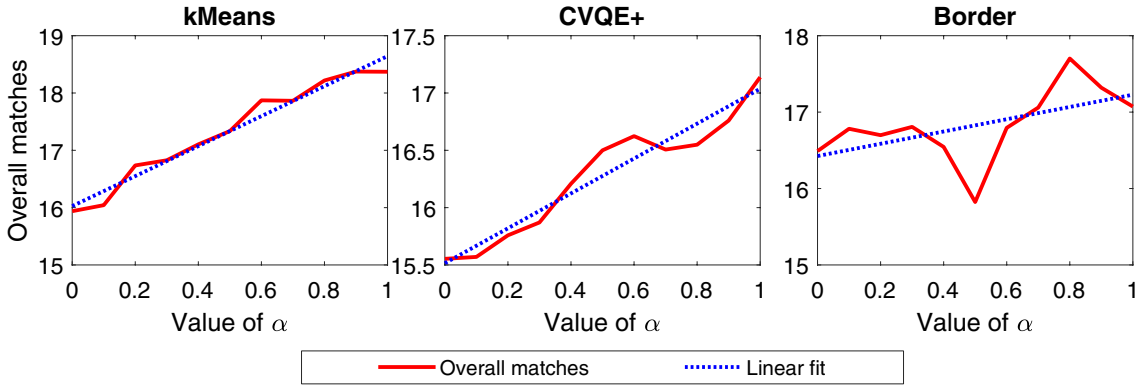

Fig. 14 The effect of α on the smoothness of results on the dataset Optdigits

of Border. However, the peak value of μ is actually dataset dependence and thus is very hard to predict. Taking the dataset Optdigits as an example, the performance of Border still increases when $\mu = 5$. However, with $\mu = 3$, Border starts perform worse on the dataset Seeds. Unfortunately, the value of μ is highly data dependent and is hard to select. In our experiments, we observe that the value of μ around 2–4 is overall good for most datasets. Thus, we choose $\mu = 4$ as a default value.

4.3 Temporal Clustering

We extends *k*means and constrained clustering CVQE+ algorithms to cope with the temporal smoothness constraint as described in Sect. 3.3.

4.3.1 Clustering Quality

We divide the datasets into different snapshots and measuring the clustering quality using the ground truths of the full datasets.

Figure 13 shows the active temporal clustering results for three snapshots of the Optdigits dataset (with $\alpha = 0.2$). As we see, our active learning scheme can help to boost clustering quality inside each snapshot compared to *k*means or CVQE+. With the constraint propagation scheme (Border), the clustering results are further boosted. For example, in Snapshot 2 and 3, Border performs much better than Border without the constraint propagation scheme. Since we only consider forward propagation, the clustering result in Snapshot 3 will be more affected than Snapshot 2 and Snapshot 1. For example, in Snapshot 3 the difference between Border and Border without Propagation is much higher than in Snapshot 2. In case of interest, we can easily extend the algorithm for a backward propagation scheme.

4.3.2 Temporal Smoothness

Given the cluster evolution graph G acquired after the temporal clustering process as described in Sect. 3.4. We define the similarity of two snapshots S_s and S_{s+1} as follows:

$$sim(S_s, S_{s+1}) = \sum_{C_i \in S_s} sim(C_i, \xi(C_i, S_{i+1})) \tag{14}$$

where $\xi(C_i, S_{s+1}) = max_{C_j \in S_{s+1}} sim(C_i, C_j)$ is the closest cluster $C_j \in S_{i+1}$ of C_i. By summing up all the similarities between two consecutive snapshots, we can measure how smooth the whole clustering results is. Obviously, higher sum means the more consistent the clustering results across snapshots are.

Figure 14 shows the overall smoothness of the clustering results of *k*means, CVQE+, and Border wrt. different values of α from 0 to 1 on the dataset Optdigits with three snapshots. The linear best fit lines (dotted) indicate that when the value of α increases, the overall smoothness among snapshots increases, i.e., the more consistent the clustering results between two consecutive snapshots. However, the trend is much clearer for *k*means compared to CVQE+ and especially Border. The reason is that in CVQE+ and Border we must trade-off the historical costs and constraint violation costs instead of only the historical costs as in *k*means. Thus, the historical aspect is more likely to be violated in CVQE+ and Border compared to *k*means. For Border, constraints are selected around the border of clusters, which are more likely to be violated than the randomly selected ones of CVQE+. Thus, the historical part of Border is more affected. As a result, the clustering results of Border are less smooth than those of *k*means and CVQE+.

5 Related Work

5.1 Constraint Clustering

There are many proposed constrained clustering algorithms such as MPC-*k*means [3], CVQE [10] and LCVQE [36]. These techniques optimize an objective function consisting of the clustering quality and the constraint violation cost like our algorithm CVQE+. CVQE+ is an extension of CVQE [10], where we extend the cost model to deal with weighted constraints, make the must-link violation cost symmetric and change the way each constraint is assigned to clusters by considering all of its related constraints. This makes cluster assignment more stable, thus enhancing the clustering quality. Interested readers are referred to [9] for a comprehensive survey on constrained clustering methods.

5.2 Active Learning

Active learning [37] are widely used in many different fields such as data clustering and pattern recognition [26, 28–31, 37, 38, 40, 42, 45].

For constrained clustering, most existing techniques employ *active learning* for acquiring a desired constraints set before or during clustering. In [2], the authors introduce the Explorer-Consolidating algorithm to select constraints by exploiting the connected components of must-link ones. Min–max [32] extends the consolidation phase of [2] by querying most uncertain objects rather than randomly selecting them. These techniques produce constraints sets before clustering. Thus, they cannot exploit the cluster labels for further enhancing performance. Huang et al. [17] introduce a framework that iteratively generates constraints and updates clustering results until a query budget is reached. However, it is limited to a probabilistic document clustering algorithm. NPU [43] also uses connected-components of must-link constraints as a guideline for finding most uncertain objects. Constraints are then collected by querying these objects again existing connected components like the Consolidate phase of [2]. Though more effective than pre-selection ones, these techniques typically have a quadratic runtime which makes them infeasible to cope with large datasets like Border. Moreover, Border relies on border objects around clusters to build constraints rather than must-link graphs [2, 43]. The inheritance approach is closely related to the constraint propagation in the multi-view clustering algorithm [13, 14] for transferring constraints among different views. The major difference is that we use the μ-nearest neighbors rather than the ϵ-neighborhoods which is limited to Gaussian clusters and can lead to an excessive number of constraints.

5.3 Temporal Clustering

Temporal smoothness has been introduced in the evolution framework [6] for making clustering results stable w.r.t. the time. We significantly extend this framework by incorporating instance-level constraints, active query selections and constraint propagation for further improving clustering quality while minimizing constraint annotation effort.

6 Application

Obstructive Sleep Apnea (OSA) is a major sleep disorder causing by the repetitive collapses of upper airway during sleep. OSA is associated with many health problems such as cardiovascular and metabolic diseases [22] including diabetes [35], coronary heart diseases [16], cancer [5] with finally an increased risk of mortality [5]. It is also known a heterogeneous disease with different symptoms and comorbidities for patients exhibiting the same level of OSA severity [21]. Thus, recent studies aim at better allocate patients into well-defined subgroups (i.e., phenotypes) based on clinical information such as symptoms, comorbidities, and demographics using clustering methods [20, 39, 41]. This can help to improve the clinical management and to define personalized treatments at time of diagnosis. For example, in [44], a Latent Class Analysis (LCA) is used to identify groups of patients in the Icelandic Sleep Apnea

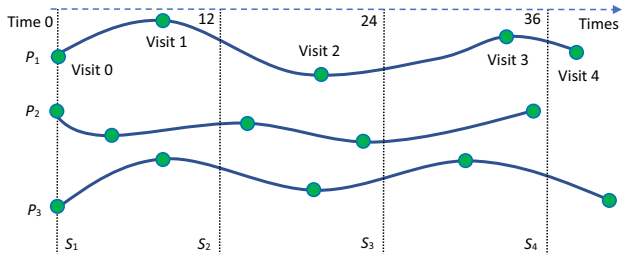

Fig. 15 The relative visit times for three patients p_1, p_2, and p_3 for snapshots S_1 to S_4 based on these visit times

Cohort including 822 patients with moderate-to-severe OSA. A similar strategy is employed for 922 patients recruited from the Sleep Apnea Global Interdisciplinary Consortium (SAGIC) [21]. Hierarchical clustering is employed in [1] for analyzing diagnosis data of 18,263 OSA patients. In [23], a relationship graph is built upon 198 patients collected from the Sleep Centre at the University of Foggia from 2012 to 2014 and patients are grouped using community detection algorithms. All of these approaches, however, do not incorporate domain knowledge into the clustering processes to improve the clustering quality. Moreover, they are only able to process static data, e.g., questionnaires [44] or a diagnosis visit data [1]. However, patients' responses to treatments and associated compilations are not static and change during clinical courses. Tracking these changes will provide more insights into disease progression and prognosis [1]. However, all existing techniques are not specifically designed to capture the evolution of patient cohorts over time.

In this section, we apply the algorithm Border to group patients into clinical meaningful clusters as well as to track how these clusters evolve over time. To the best of our knowledge, it is the first attempt that incorporates domain knowledge and tracks the cohort evolution for analyzing OSA syndromes.

6.1 OSFP Data

Our OSFP dataset is acquired from the French national registry of sleep apnea (OSFP).[2] It consists of longitudinal clinical information of many patients suffering from OSA collected from private practices, general hospitals, and university hospitals in France. At each medical visit, data are recorded such as demographic characteristics, comorbidities, and OSA symptoms as well as some environmental risk factors such as smoke, alcohol, and sedentary.

In our study, all patients that are over 18 years old and has medium to severe OSA symptoms [characterized by Apnea/Hypopnea Index (AHI) > 15 events/h or Oxygen Desaturation Index (ODI) > 15 events/h] are included. Moreover, patients with incomplete or aberrant data are excluded. At the end, we have 22,568 patients for analyze.

6.2 Snapshots

In our OSFP data, each patient data consist of information at different hospital visits as demonstrated in Fig. 15. For example, patients p_1, p_2, and p_3 have visited 5 times each. However, the visit times of patients vary significantly. For example, all p_1 visits are from 2009 to 2012, while all p_2 visits are from 2013 to 2016. Thus, it will not be reasonable if we create snapshots by using the exact visit times since patients with the same OSA symptoms and severities may be referred at different times and different follow-up points. Hence, our medical expert suggests to use the relative visit times. Concretely, we treat the first visit of a patient as time 0 (time of diagnosis) and calculate the next visits by the time difference in months to the first one. By this way, we can capture the disease evolvements. Following the relative visit times, we create different snapshots after a specific time frame of visits. For example, in Fig. 15, we use 4 different snapshots after 0, 12, 24, and 36 months. Note that if a patient has several visits at a specific snapshot, we use the last visit to present the patient status at that snapshot.

We use age, gender, Body Mass Index (BMI), environmental risk factors, comorbidities, and OSAS symptoms to group patients due to the heterogeneity of the OSA [1, 21, 44]. We divide our data into different snapshots with a time difference of 12 months, which is long enough to capture the disease changes. We set the number of clusters to 6 as suggested by [1].

6.3 Clusters at Snapshot 1 (Time 0)

Figure 16 shows the cluster centers of snapshot 1 (at time 0). Overall we have 6 clusters mainly discriminated by age, comorbidities, and symptoms as follows:

– *Group A* Youngest, very few comorbidities, and highest OSA severity (Cluster 6 in Fig. 16 with 4535 patients): this cluster is the middle group in terms of BMI (average BMI = 32.03) and the youngest group (average age = 49.99). Patients in this group consume less alcohol than those from other groups but smoke the most. They also among groups with lowest numbers of comorbidities. However, they suffer from the highest numbers of OSAS symptoms. For example, 25.7% people in this group has high blood pressure while 83.5% and 96.9% of them experience morning headache and fatigue, respectively. They also have the highest functional scales with median scales of Epworth, Pichot and Depression as 12.09, 16.1, and 4.92, respectively.

[2] http://www.osfp.fr.

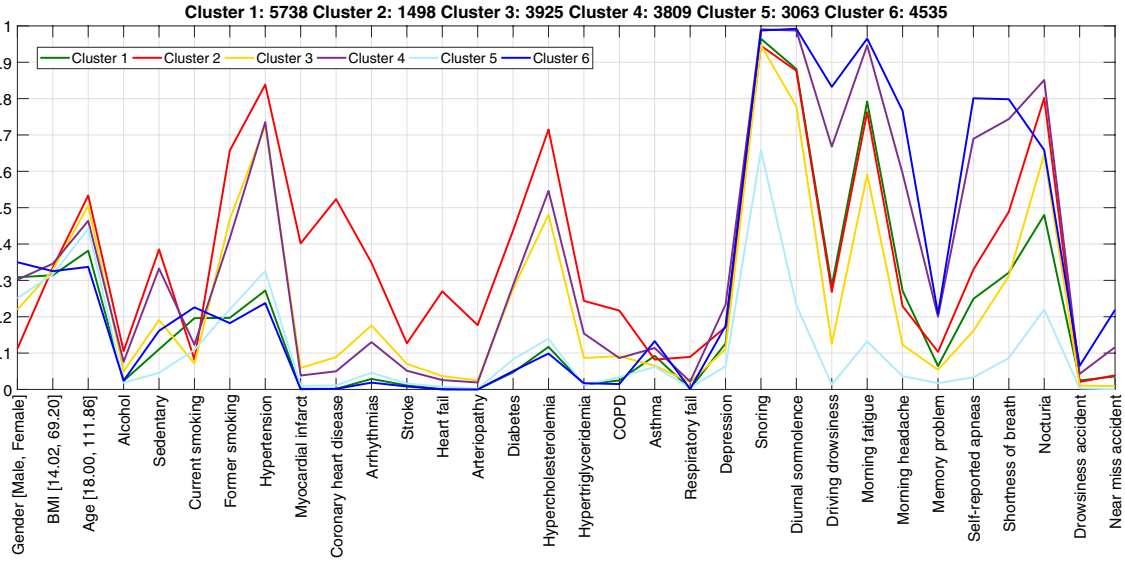

Fig. 16 The centers of clusters at time 0. The y-coord shows the percentage of patients in a group with a specific symptom (from Snoring to Near miss accident) or comorbidity (from hypertension to depression), except for Gender (percentage of Female), BMI and age (normalized into [0,1]) (best view in colors)

- *Group B* Oldest, poorest life style, highest comorbidities, and medium OSA severity (Cluster 2 in Fig. 16 with 1498 patients): this cluster consists of oldest people (median age = 68.08) who consume more alcohol than other groups and has the poorest life style. They have an average BMI = 32.5 and the highest number of comorbidities, e.g., 83.85% with high blood pressure. However, they show average OSA symptoms compared to other clusters.
- *Group C* Average age, best life style, few comorbidities, and lowest OSA severity (Cluster 5 in Fig. 16 with 3063 patients): this group has an average age of 59.48 and is among the lowest BMI groups (average BMI = 31.50). They have a particular good life style and consume less alcohols than others. They have few comorbidities. However, they suffer from the lowest numbers of OSA symptoms. Their functional scales are the lowest with medians of Epworth, Pichot, and Depression as 7.61, 7.16 and 2.35, respectively.
- *Group D* Average age, poor life style, high comorbidities, and high OSA severity (Cluster 4 in Fig. 16 with 3809 patients): people in this group have median age of 61.54, poor life styles, medium comorbidities, and suffer from high OSA severity. They also belong to the most obese group with average BMI of 33.10. Moreover, their medians of AHI (44.7 events/h) and ODI (36.03 events/h) are the highest among groups.
- *Group E* Second oldest group, few comorbidities, and medium OSA severity (Cluster 3 in Fig. 16 with 3925 patients): the medians age and BMI of this group are 65.47 and 32.10, respectively. The numbers of comorbidities are high. And people have medium OSA severity.
- *Group F* Second youngest group, very few comorbidities, and medium OSA severity (Cluster 1 in Fig. 16 with 5738 patients): the averaged AHI and ODI of this groups are 40.98 and 30.68 events/h, respectively, and are the lowest of all groups. The average age is 53.82. This group is the least obese one with averaged BMI of 31.35. It has few comorbidities and medium OSA symptoms.

6.4 Clusters at Other Snapshots

Figure 17 shows the cluster centers for Snapshot 2 (top left) to Snapshot 5 (bottom right). The acquired clusters match quite well with those found in Snapshot 1. Let us denote C_{ij} as the cluster j at snapshot i. We can see that *Group A*, for an example, appears in other snapshots such as C_{26}, C_{33}, C_{43}, and C_{52}. The similar observations can be seen for other groups B to F as well. Here the temporal smoothness constraint described in Sect. 3.3 keeps clusters in two consecutive snapshots from deviating too much from each other. Thus, it makes the clustering results in all snapshots consistent.

6.5 Evolution Graph

To keep track of the relationship between groups of patients over time, we build the evolutionary graph G as shown in Fig. 18 (left). If two clusters in two consecutive snapshots share some patients, we put a weighted edge between them indicating how many patients they share.

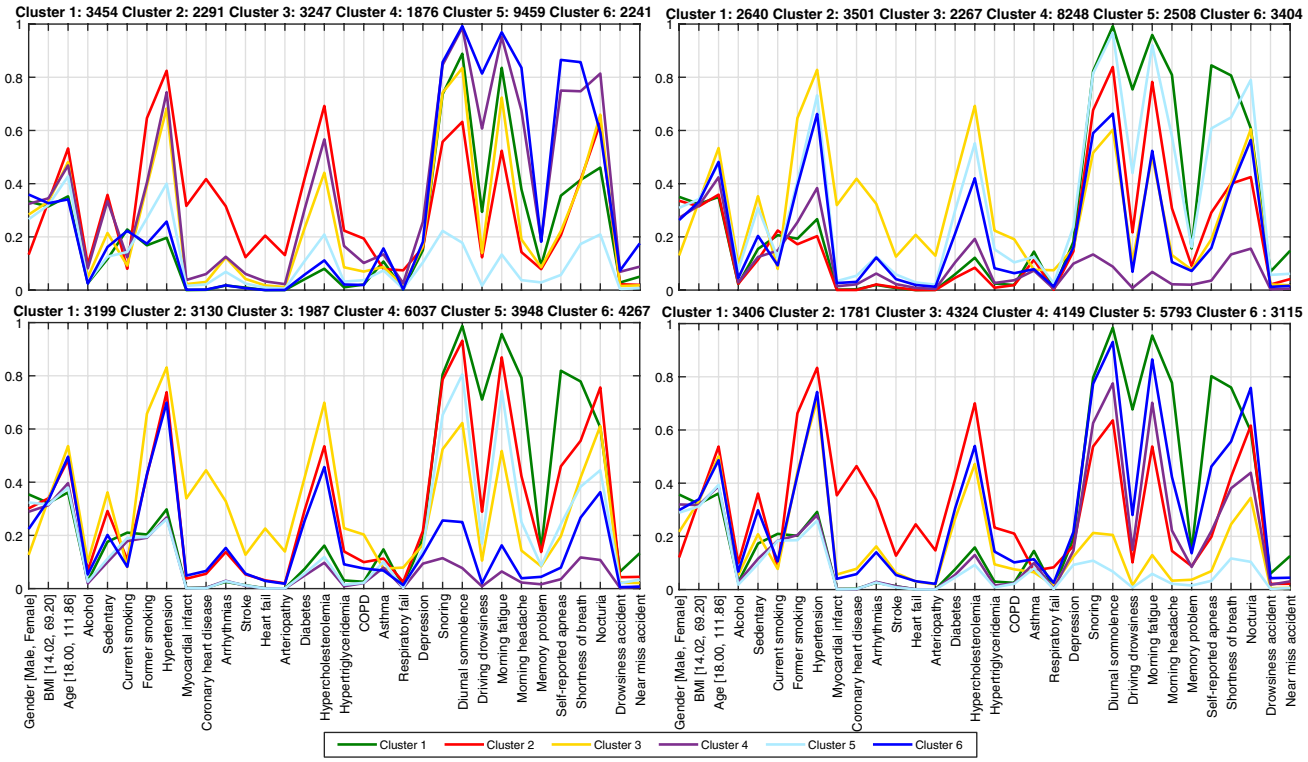

Fig. 17 The centers of clusters of Snapshot 2–5 (top left to bottom right). The *y*-coord shows the percentage of patients in a group with a specific symptom or comorbidity, except for gender (percentage of female), BMI and age (normalized into [0,1]) (best view in colors)

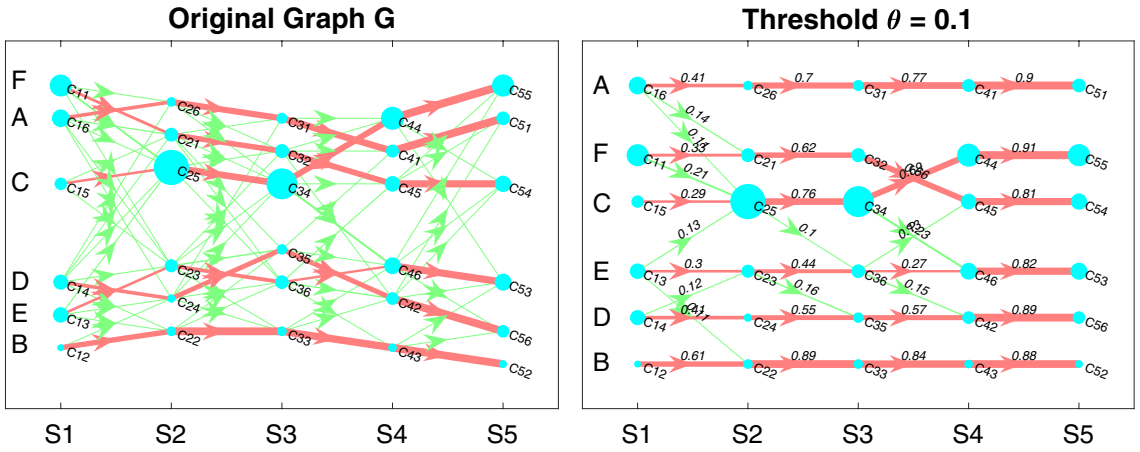

Fig. 18 The evolution graph G for Snapshot 1–5. The size of each node denotes the number of patients inside this node. An edge represents the relationship between two clusters with the weight as the edge's thickness. The red edge indicates a pair of two closest clusters. The left figure shows the original graph G and the right one shows the reduced graph G with the threshold $\theta = 0.1$ (best view in colors)

However, since a borderline patient p may be assigned to two different clusters during the clustering process, some edges with small weights may be presented in G but without any significant meaning. Thus, we propose to use a predefined threshold θ in [0,1] to filter out those insignificant ones if their weights are below the threshold. The acquired graph is shown in Fig. 18 (right). This simplified graph shows major transitions among clusters and snapshots. Thus, it captures many useful information on evolvement of the whole patient trajectories over time such as: (1) how a particular group of patients evolve over time? (2) analyzing sets of patients with particular group

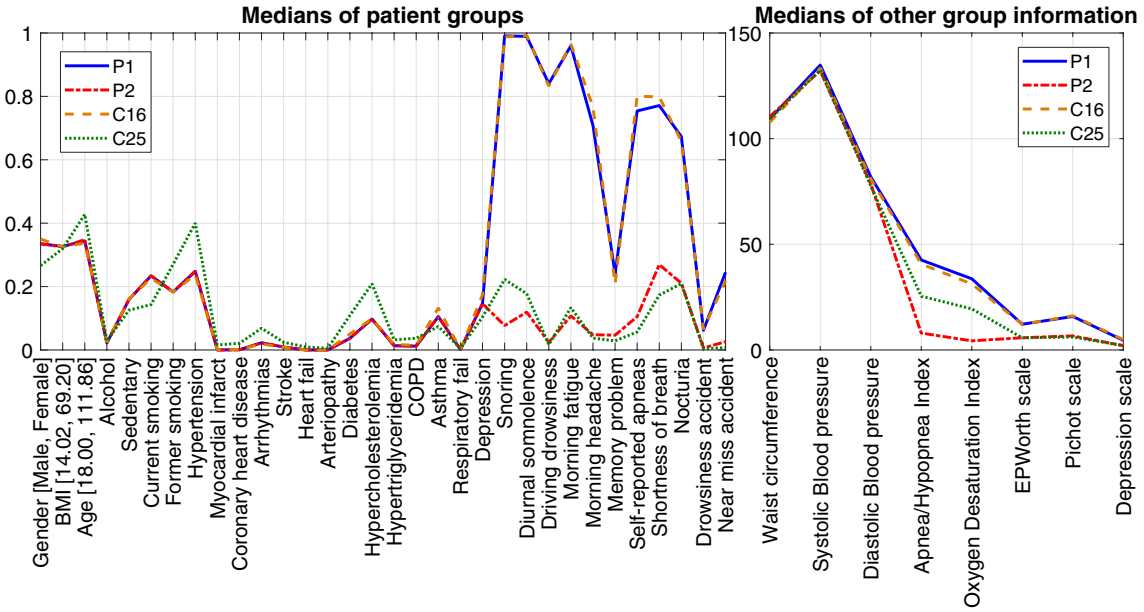

Fig. 19 The set P of patients that changes from the group A (most severe OSA symptoms) in Snapshot 1 to the group C (least severe OSA symptoms) in Snapshot 2. Here P_i means the set of patients P in Snapshot S_i and C_{ij} means cluster C_j in Snapshot S_i

changes; or (3) analyzing sets of patients with a particular disease evolvement path over time.

6.6 Tracking the Group Changes

The graph G can be used to track the group changes of a set of patients. For example, we can see that there is a set P of 1415 patients that changes from the group A (Cluster 6 in Fig. 16 with 4535 patients) in Snapshot 1 into the group C (Cluster 5 in Fig. 17 with 9459 patients) in Snapshot 2. The question is how it happens?

Figure 19 shows the medians of the set of patients P in S_1 and S_2 (denoted as P_1 and P_2) together with the medians of groups A in S_1 (C_{16}) and C in S_2 (C_{25}). In S_1, P has slightly lower OSA severity compared to the whole group A. However, after 12 months, all OSA symptoms have reduced significantly. For example, only 7.70% patients in P snore compared to 99.08% one year before. The average AHI and ODI also drop from 42.56 and 34.48 to 8.62 and 5.02, respectively. The Epworth, Pichot, and Depression scales also decrease significantly. As a consequence, P is moved from the most OSA severity group to the lowest OSA severity group. Now, what causes these changes? To answer that, we examine all treatments applied for those patients. It turns out that 66.64% and 90.88% of patients in P were treated by lifestyle interventions (lifestyle) and Continuous positive airway pressure (CPAP), respectively, compared to the average values of the whole group A (63.99% and 87.71%). This implies that P responses better to lifestyle and CPAP than others in A. Overall, P is an interesting group and should be taken out for further medical analyses. On the other hand, all of patients in $C_{16} \backslash C_{25}$ do not change group, indicating that they are not response well to treatments. Thus, they will be another important group to further study.

6.7 How a Specific Group of Patients Evolve Over Time?

Following the most closest groups across snapshots (the red edges in Fig. 18), we can track the evolution of a specific group over time. Let take the group B (oldest, highest comorbidities, and medium OSA severe) shown in Fig. 20 as an example. From S_1 to S_2, the average number of comorbidities and OSA symptoms per patients decrease considerably. The main reason is that 594 patients with high comorbidities in the group E (C_3) of S_1 but lower comorbidities and symptoms than B has moved to C_2 of S_2 as shown in Fig. 18. From S_2 to S_5, there are not many patients moving in and out of the group while the average numbers of comorbidities and symptoms, AHI, ODI, and Functional scales change very slightly. This implies that patients in this group B do not response well to current treatments (mainly by lifestyle and CPAP). Their diseases even increase in most cases (perhaps due to the old age consequences). Thus, alternative treatments should be considered.

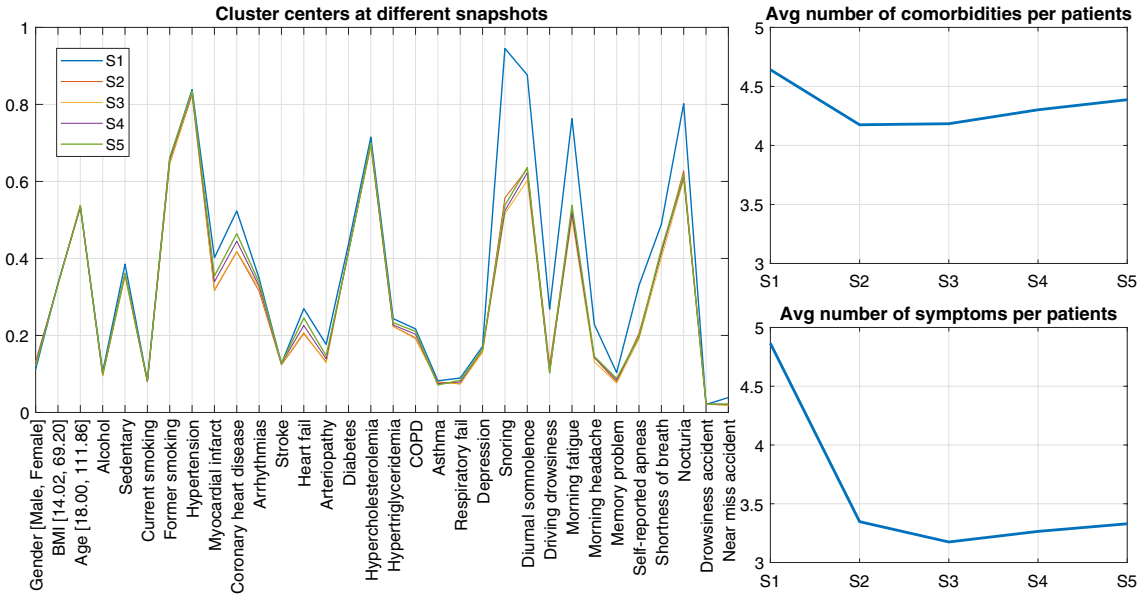

Fig. 20 The change of the group B (oldest, highest comorbidities, and medium OSA severe) from Snapshot 1 to 5

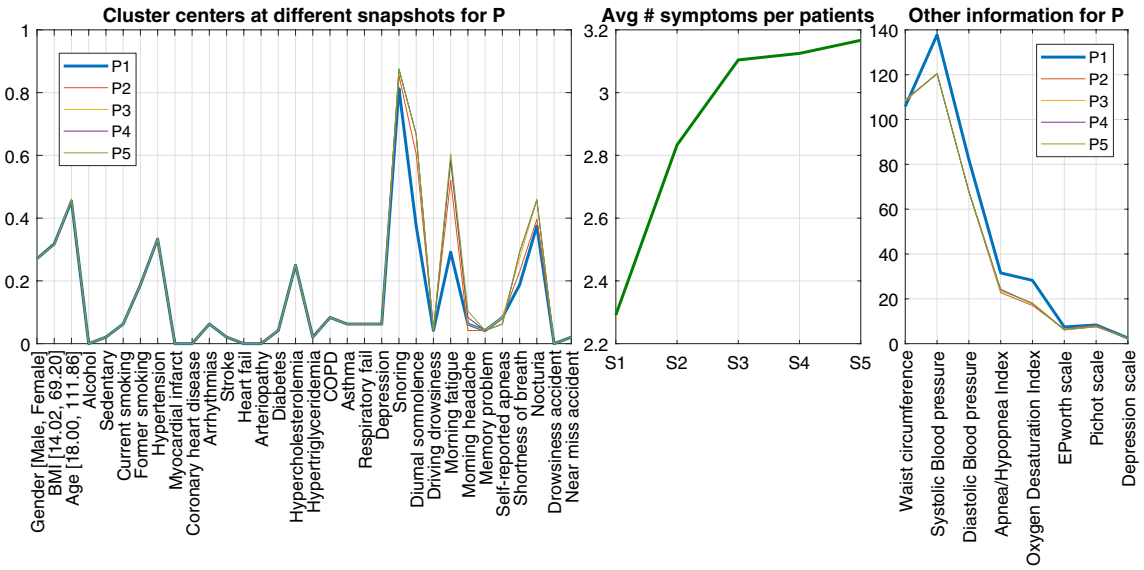

Fig. 21 The set P of 48 patients who start from the group C in S_1 with a low OSA severity but end up in the group F in S_5 with a medium OSA severity

6.8 Tracking Patients with a Specific Path

From the graph G, we can easily track the information of a set of patients whose diseases evolve by a specific path. Figure 21 shows the set P of 48 patients who has OSA severity changes from very low to high following the path C_{15}, C_{25}, C_{36}, C_{45}, and C_{54} in Fig. 18. As we can see, from S_1 to S_5, the average number of symptoms per patients increases considerably from 2.29 to 3.16 while the average number of comorbidities stands still. Some specific symptoms such as Morning fatigue, Nocturia, and Snoring increase over time. However, the changes in AHI and ODI are not clear from S_2 to S_5. However, they are still over 15, indicating a medium to severe OSA level. The rates of Oxygen therapy, RLS drug treatments, and ventilation are also higher than other groups, while CPAP and lifestyle are less used. Thus, increasing CPAP and lifestyle treatments for this group may be helpful.

7 Discussion

Throughout this section, we present how our algorithm Border can be used for finding or tracking the evolution of clinical meaningful groups of patients. Currently, our study is done on the largest and generalized collection of patients in the field with wide range of attributes. However, there are still some limitations and potential future works that we are aiming at. First, while we use the whole dataset for studying the heterogeneity of OSA patients, examining more specific patient cohorts may help to reveal more interesting information, e.g., patients with Oxygen therapy or patients with hypertension and nocturia [12]. Second, other environmental factors such as air pollutions is known to be related to OSA, especially for children [24]. Studying how the disease evolve over time wrt. the air pollution level at some specific locations will be a very interesting direction to pursue.

8 Conclusion

We introduce a scalable novel framework which incorporates an iterative active learning scheme, instance-level and temporal smoothness constraints for coping with large temporal data. Experiments show that our constrained clustering algorithm, CVQE+, performs better than existing techniques such as CVQE [10], LCVQE [36] and MPC-*k*means [2]. By exploring border objects and propagating constraints via nearest neighbors, our active learning algorithm, Border, results in good clustering results with much smaller constraint sets compared to other methods such as NPU [43] and min–max [32]. Moreover, it is orders of magnitude faster making it possible to cope with large datasets. Finally, we revisit our approach in the context of evolutionary clustering adding a temporal smoothness constraint and a time-fading factor to our constraint propagation among different data snapshots. Our future work aims at providing more expressive support for user feedback as well as improving the performance of CVQE+ on noisy constraints. We are currently using our framework to track group evolution of our patient data with sleeping disorder symptoms.

Acknowledgements This work is supported by the French National Research Agency in the framework of the "Investissements d'avenir" program (ANR-15-IDEX-02).

References

1. Bailly S, Destors M, Grillet Y, Richard P, Stach B, Vivodtzev I, Timsit JF, Lévy P, Tamisier R, Pépin JL, Scientific Council, Investigators of the French National Sleep Apnea Registry (OSFP) (2016) Obstructive sleep apnea: a cluster analysis at time of diagnosis. PLOS ONE 11(6):1–12
2. Basu S, Banerjee A, Mooney RJ (2004) Active semi-supervision for pairwise constrained clustering. In: SDM, pp 333–344
3. Bilenko M, Basu S, Mooney RJ (2004) Integrating constraints and metric learning in semi-supervised clustering. In: ICML
4. Birgé L, Rozenholc Y (2006) How many bins should be put in a regular histogram. ESAIM Probab Stat 10:2445. https://doi.org/10.1051/ps:2006001
5. Campos-Rodriguez F, Martinez-Garcia MA, Martinez M, Duran-Cantolla J, Pea MDL, Masdeu MJ, Gonzalez M, Campo FD, Gallego I, Marin JM, Barbe F, Montserrat JM, Farre RA (2013) Association between obstructive sleep apnea and cancer incidence in a large multicenter Spanish cohort. Am J Respir Crit Care Med 187(1):99–105
6. Chakrabarti D, Kumar R, Tomkins A (2006) Evolutionary clustering. In: SIGKDD, pp 554–560
7. Cohn D, Caruana R, Mccallum A (2003) Semi-supervised clustering with user feedback. Technical report
8. Davidson I (2012) Two approaches to understanding when constraints help clustering. In: KDD, pp 1312–1320
9. Davidson I, Basu S (2007) A survey of clustering with instance level constraints. TKDD
10. Davidson I, Ravi SS (2005) Clustering with constraints: feasibility issues and the *k*-means algorithm. In: SDM, pp 138–149
11. Davidson I, Ravi SS, Ester M (2007) Efficient incremental constrained clustering. In: KDD, pp 240–249
12. Destors M, Tamisier R, Sapene M, Grillet Y, Baguet JP, Richard P, Girey-Rannaud J, Dias-Domingos S, Martin F, Stach B, Housset B, Levy P, Pepin JL (2014) Nocturia is an independent predictive factor of prevalent hypertension in obstructive sleep apnea patients. Eur Respir J 44(Suppl 58):P1744
13. Eaton E, desJardins M, Jacob S (2010) Multi-view clustering with constraint propagation for learning with an incomplete mapping between views. In: CIKM, pp 389–398
14. Eaton E, desJardins M, Jacob S (2014) Multi-view constrained clustering with an incomplete mapping between views. Knowl Inf Syst 38(1):231–257
15. Han J (2005) Data mining: concepts and techniques. Morgan Kaufmann Publishers Inc., San Francisco
16. Hla KM, Young T, Hagen EW, Stein JH, Finn LA, Nieto FJ, Peppard PE (2015) Coronary heart disease incidence in sleep disordered breathing: the Wisconsin sleep cohort study. Sleep 38(5):677–684
17. Huang R, Lam W (2007) Semi-supervised document clustering via active learning with pairwise constraints. In: ICDM, pp 517–522
18. Huang Y, Mitchell TM (2006) Text clustering with extended user feedback. In: SIGIR, pp 413–420
19. Jensen A, Moseley P, Oprea T, Ellese S, Eriksson R, Schmock H, Jensen P, Jensen L, Brunak S (2014) Temporal disease trajectories condensed from population-wide registry data covering 6.2 million patients. Nat Commun 5:4022
20. Joosten SA, Hamza K, Sands S, Turton A, Berger P, Hamilton GS (2011) Phenotypes of patients with mild to moderate obstructive sleep apnoea as confirmed by cluster analysis. Respirology 17(1):99–107
21. Keenan BT, Kim J, Singh B, Bittencourt L, Chen NH, Cistulli PA, Magalang UJ, McArdle N, Mindel JW, Benediktsdottir B, Arnardottir ES, Prochnow LK, Penzel T, Sanner B, Schwab

RJ, Shin C, Sutherland K, Tufik S, Maislin G, Gislason T, Pack AI (2018) Recognizable clinical subtypes of obstructive sleep apnea across international sleep centers: a cluster analysis. Sleep 41(3):zsx214
22. Kendzerska T, Gershon AS, Hawker G, Leung RS, Tomlinson G (2014) Obstructive sleep apnea and risk of cardiovascular events and all-cause mortality: a decade-long historical cohort study. PLOS Med 11(2):1–15
23. Lacedonia D, Carpagnano GE, Sabato R, Storto MMl, Palmiotti GA, Capozzi V, Barbaro MPF, Gallo C, (2016) Characterization of obstructive sleep apnea-hypopnea syndrome (OSA) population by means of cluster analysis. J Sleep Res 25(6):724–730
24. Lawrence WR, Yang M, Zhang C, Liu RQ, Lin S, Wang SQ, Liu Y, Ma H, Chen DH, Zeng XW, Yang BY, Hu LW, Yim SHL, Dong GH (2018) Association between long-term exposure to air pollution and sleep disorder in Chinese children: the Seven Northeastern Cities study. Sleep 41:zsy122
25. Lévy P, Kohler M, McNicholas WT, Barbé F, McEvoy RD, Somers VK et al. (2015) Obstructive sleep apnoea syndrome. Nat Rev Dis Primers 1:15015
26. Mai ST, Amer-Yahia S, Chouakria AD (2018) Scalable active temporal constrained clustering. In: EDBT, pp 449–452
27. Mai ST, Amer-Yahia S, Chouakria AD, Nguyen KT, Nguyen A (2018) Scalable active constrained clustering for temporal data. In: DASFAA, pp 566–582
28. Mai ST, Assent I, Jacobsen J, Dieu MS (2018) Anytime parallel density-based clustering. Data Min Knowl Discov 32(4):1121–1176
29. Mai ST, Assent I, Storgaard M (2016) AnyDBC: an efficient anytime density-based clustering algorithm for very large complex datasets. In: SIGKDD, pp 1025–1034
30. Mai ST, Dieu MS, Assent I, Jacobsen J, Kristensen J, Birk M (2017) Scalable and interactive graph clustering algorithm on multicore CPUs. In: IEEE international conference on data engineering (ICDE), pp 349–360
31. Mai ST, He X, Hubig N, Plant C, Böhm C (2013) Active density-based clustering. In: ICDM, pp 508–517
32. Mallapragada PK, Jin R, Jain AK (2008) Active query selection for semi-supervised clustering. In: ICPR, pp 1–4
33. Nguyen XV, Epps J, Bailey J (2009) Information theoretic measures for clusterings comparison: is a correction for chance necessary? In: ICML, pp 1073–1080
34. Nieto FJ, Peppard PE, Young T, Finn L, Hla KM, Farré R (2012) Sleep-disordered breathing and cancer mortality. Am J Respir Crit Care Med 186(2):190–194
35. Pamidi S, Tasali E (2012) Obstructive sleep apnea and type 2 diabetes: is there a link? Front Eurol 3:126
36. Pelleg D, Baras D (2007) *K*-means with large and noisy constraint sets. In: ECML, pp 674–682
37. Settles B (2010) Active learning literature survey. Technical report 1648, University of Wisconsin–Madison
38. Son MT, Amer-Yahia S, Assent I, Birk M, Dieu MS, Jacobsen J, Kristensen J (2018) Scalable interactive dynamic graph clustering on multicore CPUs. IEEE Trans Knowl Data Eng (TKDE) (**to appear**)
39. Tsuchiya M, Lowe AA, Pae EK, Fleetham JA (1992) Obstructive sleep apnea subtypes by cluster analysis. Am J Orthod Dentofac Orthop 101(6):533–542
40. Tuia D, Muñoz-Marí J, Camps-Valls G (2012) Remote sensing image segmentation by active queries. Pattern Recognit 45(6):2180–2192
41. Vavougios GD, Natsios G, Pastaka C, Zarogiannis SG, Gourgoulianis KI (2016) Phenotypes of comorbidity in OSAS patients: combining categorical principal component analysis with cluster analysis. J Sleep Res 25(1):31–38
42. Voevodski K, Balcan MF, Röglin H, Teng SH, Xia Y (2012) Active clustering of biological sequences. J Mach Learn Res 13:203–225
43. Xiong S, Azimi J, Fern XZ (2014) Active learning of constraints for semi-supervised clustering. IEEE Trans Knowl Data Eng 26(1):43–54
44. Ye L, Pien GW, Ratcliffe SJ, Björnsdottir E, Arnardottir ES, Pack AI, Benediktsdottir B, Gislason T (2014) The different clinical faces of obstructive sleep apnoea: a cluster analysis. Eur Respir J 44(6):1600–1607
45. Zhao W, He Q, Ma H, Shi Z (2012) Effective semi-supervised document clustering via active learning with instance-level constraints. Knowl Inf Syst 30(3):569–587