# Scalable Active Temporal Constrained Clustering

Son T. Mai, Sihem Amer-Yahia, Ahlame Douzal Chouakria
CNRS, Univ. Grenoble Alpes, France
{mtson,sihem.amer-yahia,ahlame.douzal}@univ-grenoble-alpes.fr

## ABSTRACT

We introduce a novel interactive framework to handle both instance-level and temporal smoothness constraints for clustering large temporal data. It consists of a constrained clustering algorithm 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 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 snapshot and between snapshots via constraint inheritance and propagation to further enhance the results. Experiments show better or comparable clustering results than existing techniques as well as high scalability for large datasets.

## 1 INTRODUCTION

In semi-supervised clustering, domain knowledge is typically encoded in the form of instance-level *must-link* and *cannot-link* constraints [8]. Such constraints specify that two objects must be placed in the same clusters or not. Constraints have been successfully applied to improve clustering quality in real-world applications, e.g., identifying people from surveillance cameras [8] and aiding robot navigation [7]. However, current research on constrained clustering still suffers from several issues.

Most existing approaches assume that we have a set of constraints beforehand, and an algorithm will use this set to produce clusters [2, 7]. Davidson et al. show that the clustering quality varies significantly using different equi-size sets of constraints [5]. 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 randomly selecting them from the data has been the focus of many research efforts, e.g., [1, 11, 14].

Many approaches employ different *active learning* schemes to select the most meaningful pairs of objects and then query experts for constraint annotation [1, 11]. 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 randomized constraint selection. These constraints are then used as input for constrained clustering algorithms to operate. However, if users are not satisfied with the results, they are asked to provide another constraint set and start the clustering again, which is obviously time consuming and expensive.

Other algorithms follow a *feedback* schema which does not require a full set of constraints in the beginning [4]. 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 [4]. 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., [10, 11, 14]. 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 active feedback techniques are proven to be very useful in real-world tasks such as document clustering [10], they suffer from very high runtime since they have to repeatedly perform clustering as well as exploring all $O(n^2)$ pairs of objects to generate queries to users.

In this paper, we develop an efficient framework to cope with the above problems following the iterative active learning approach as in [10, 14]. 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 $\mu$-nearest neighbors for inferring additional constraints, thus further boosting performance. Finally, we revisit our approach in the context of evolutionary clustering. Evolutionary clustering aims to produce high quality clusters while ensuring that the clustering does not change dramatically between consecutive timestamps. We propose to formulate a temporal smoothness constraint and add a time-fading factor to our constraint propagation.

This paper's contributions are: (i) a new algorithm CVQE+ that extends CVQE [7] with weighted must-link and cannot-link constraints, (ii) a new algorithm, Border, that relies on active clustering and constraint inheritance to choose a small number of objects to solicit user feedback for, (iii) an evolutionary clustering framework which incorporates instance-level and temporal smoothness constraints, and (iv) experiments with 6 datasets that show the superiority of our algorithms over state-of-the-art ones.

## 2 PROBLEM FORMULATION

Let $D = \{(d, t)\}$ be a set of of $|D|$ vectors $d \in \mathbb{R}^p$ observed at time $t$. 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 $D_s = \{(d, t) \in D \mid ts_s \leq t < te_s\}$. 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 $ML_s = \{(x, y, w_{xy})\}$ be a set of *must-link* constraints related to $x, y \in D_s$ with a degree of belief of $w_{xy} \in [0, 1]$. Similarly, let $CL_s = \{(x, y, w_{xy})\}$ be a set of *cannot-link* constraints of $S_s$. Initially, $ML_s$ and $CL_s$ can be empty.

In this paper, we focus on the problem of grouping objects in all snapshots into clusters. 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.

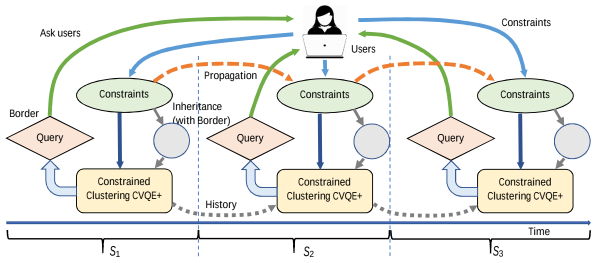

**Figure 1: Our active temporal clustering framework**

## 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 [4]. However, instead of passively waiting for user feedback as in [4], our algorithm, Border, *actively* examines the current cluster structure, selects $\beta$ pairs of objects whose 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 [10, 14], our algorithm, CVQE+, incrementally updates the cluster structures. We also aim to ensure a smooth transition between consecutive clusterings [3]. 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 for acquiring a desired level of clustering quality. To the best of our knowledge, Border is the first framework that combines active learning, instance-level and temporal smoothness constraints.

**The new algorithm CVQE+.** For each snapshot $S_s$, we use constrained kMeans for grouping objects. Any existing techniques such as MPCK-Means [2], CVQE [7] or LCVQE [13] can be used. Here we introduce CVQE+, an extension of CVQE [7] to cope with weighted constraints, to do the task. 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 $CL_i$ as follows. Note that, our $ML_i$ cost is symmetric compared to [7].

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

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

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

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

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

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

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 [6] or LCVQE [6], which is very sensitive to the cost change when some constraints share the same objects (changing these objects affects all their constraints). Thus, this scheme is expected to improve the clustering quality of CVQE+ compared to CVQE and LCVQE.

**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. For each object $a$ in cluster $C_i$, the border score of $a$ is defined as $bor(a) = \frac{(a-c_i)^2}{(a-c_{\varphi(i)})^2(1+ml(a))(1+cl(a))}$, 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. 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). Then, objects are taken sequentially from the outermost bins.

For each selected object $a$, we estimate the uncertainty of $a$ as $sco(a) = ent(\mu nn(a)) + \frac{vl(ml(a))+vl(cl(a))}{ml(a)+cl(a)+1}$, where $ent(\mu nn(a))$ is the entropy of class labels of $\mu$ 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.

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. 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).

We show $\beta$ 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 $\delta$.

**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 $\mu$ nearest neighbors below.

Let $h$ be the distance between an object $p$ and its $\mu$ 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 Figure 2. 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:

$$w_{ab} = w_{pq}\phi_h(p, a)\phi_h(q, b) \tag{3}$$

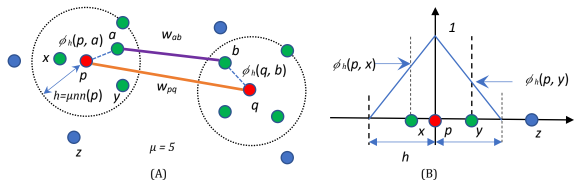

**Figure 2: (A) Constraint inheritance from $(p, q)$ to $(a, b)$. (B) The effect of the object $b$ on its neighbors**

The general intuition is that the label of an object $a$ tends to be consistent with its closest neighbors (which is commonly used in classification). This scheme is expected to increase the clustering quality, especially when combined with the active learning approach described above.

**Updating clusters.** For incrementally updating clusters, we only need to take the old cluster centers and update them following Equation 1 with the updated constraints set.

**Temporal smoothness.** The general idea of temporal smoothness [3] is that clusters not only have high quality in each snapshot but also do not change much between sequential time frames. We re-define the cost of cluster $C_i$ of snapshot $S_s$ in Equation 1 as follows:

$$TCost_{VQE_i} = (1-\alpha)Cost_{VQE_i} + \alpha Hist(C_i, S_{s-1}) \tag{4}$$

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 $\alpha$ is a regulation factor to balance the current clustering quality and the historical cost. We define the historical cost as follows:

$$Hist(C_i, S_{s-1}) = (c_i - \psi(C_i, S_{s-1}))^2 \tag{5}$$

where $\psi(C_i, S_{s-1})$ returns the closest cluster center to $C_i$ in snapshot $S_{s-1}$. Taking the derivation of (4) as in (1), we have $C_i = \frac{(1-\alpha)A+\alpha\psi(C_i,S_{s-1})}{(1-\alpha)B+\alpha}$, where $A$ and $B$ are respectively the numerator and the denominator given in Equation 1.

**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 $x, y \in S'$. 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:

$$w'_{xy} = w_{xy}\frac{te_s - ts_{s'}}{te_{s'} - ts_s} \tag{6}$$

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.

**Complexity analysis.** Let $n$ be the number of objects. Both CVQE+ and Border have linear time complexity to $O(n)$.

## 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. We use Normalized Mutual Information (NMI) [12] for assessing the clustering quality. All results are averaged over 10 runs.

**Performance of CVQE+.** Figure 3 shows comparisons among CVQE+ and existing techniques including kMeans, MPCK-Means [2], CVQE [7] and LCVQE [13]. CVQE+ consistently outperforms or acquires comparable results to CVQE and others, especially when the number of constraints is large. This can be explained

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

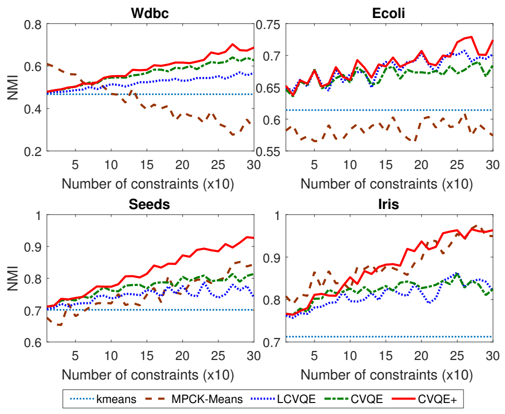

**Figure 3: Performance of CVQE+ compared to others**

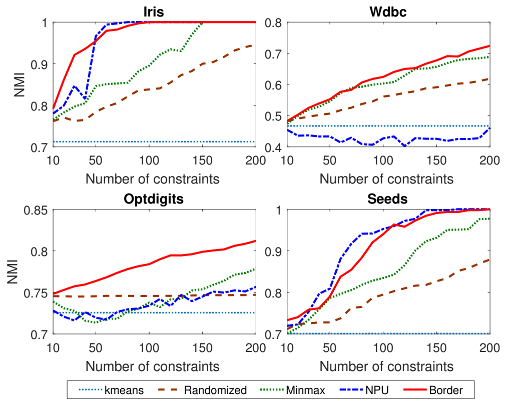

**Figure 4: Comparison among different active learning techniques**

by the way CVQE+ assigns objects to clusters. 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.

**Active constraint selection.** Unless otherwise stated, the budget limitation $\delta$ is set to 200, the query size $\beta = 10$ and the neighborhood size $\mu = 4$. Figure 4 shows comparisons between Border, NPU [14], Huang [14] (a modified version of [10] for working with non-document data), Min-max [11], Explorer-Consolidate [1], and a randomized method (Huang and Consolidate are removed from Figure 4 for readability). Border acquires better results than others on Libras, Wdbc and Optidigits, 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 others. However, Border has some parameters to set such as the query size $\beta$. Tuning these parameters is a difficult problem that requires deeper investigation.

For runtimes, we create five synthetics datasets of sizes 2000 to 10000 consisting of 5 Gaussian clusters and measure the time for acquiring 100 constraints. Border is orders of magnitude faster than others in selecting pairs to query. For 1000 objects, it take Border 0.1 seconds while NPU and Min-max need 439.4 and 3.0 seconds. For 10000 objects, Border, NPU and Min-max consumes 0.18, 5216.3 and 18.2 seconds, respectively. Additionally, the higher the number of objects and constraints, the higher the runtime differences. We omit the plots due to space limitations.

**Cluster update.** Figure 5 shows the NMI and the number of iterations of our algorithm for the Ecoli dataset. The NMI scores

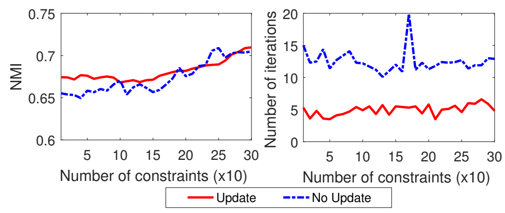

**Figure 5: Update vs. fully reclustering for the Ecoli dataset**

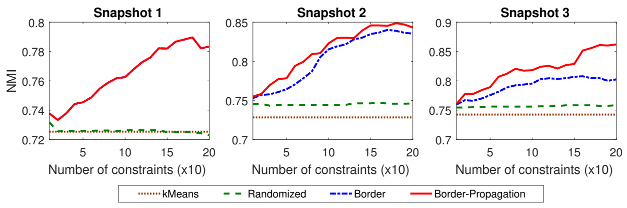

**Figure 6: Temporal clustering on the dataset Optdigits**

are comparable, while it takes fewer iterations for our algorithm to converge in its update mode.

**Temporal clustering.** Figure 6 shows the active temporal clustering results for three snapshots of the Optdigits dataset (we set $\alpha = 0.5$). As we see, our active learning scheme can help boost clustering quality inside each snapshot compared to the original kMeans or a randomized constraint selection method. With the constraint propagation scheme (Border-Propagation), the clustering results are further boosted compared to Border. Since we only consider forward propagation, the clustering result in Snapshot 3 will be more affected than Snapshot 2 and Snapshot 1. We can easily extend the algorithm for the backward propagation case.

## 5 RELATED WORK

**Constraint clustering.** There are many proposed constrained clustering algorithms such as MPC-kMeans [2], CVQE [7] and LCVQE [13]. 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 [7], 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 [6] for a comprehensive survey on constrained clustering methods.

**Active learning.** Most existing techniques employ *active learning* for acquiring a desired constraints set before or during clustering. In [1], the authors introduce the Explorer-Consolidating algorithm to select constraints by exploiting the connected-components of must-link ones. Min-max [11] extends the Consolidation phase of [1] 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. [10] 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 [14] 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 [1]. 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 [1, 14]. The inheritance approach is closely related to the constraint propagation in the multi-view clustering algorithm [9] for transferring constraints among different views. The major difference is that we use the $\mu$-nearest neighbors rather than the $\epsilon$-neighborhoods which is limited to Gaussian clusters and can lead to an excessive number of constraints.

**Temporal clustering.** Temporal smoothness has been introduced in the evolution framework [3] 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 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 [7], LCVQE [13] and MPC-kMeans [1]. 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 [14] and Min-max [11]. 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. We are currently using our framework to track group evolution of our patient data with sleeping disorder symptoms.

**Acknowledgments.** We thank anonymous reviewers for their valuable comments. This work is supported by the IDEX CDP LIFE Project.

## REFERENCES

[1] Sugato Basu, Arindam Banerjee, and Raymond J. Mooney. 2004. Active Semi-Supervision for Pairwise Constrained Clustering. In *SDM*. 333–344.

[2] Mikhail Bilenko, Sugato Basu, and Raymond J. Mooney. 2004. Integrating constraints and metric learning in semi-supervised clustering. In *ICML*.

[3] Deepayan Chakrabarti, Ravi Kumar, and Andrew Tomkins. 2006. Evolutionary clustering. In *SIGKDD*. 554–560.

[4] David Cohn, Rich Caruana, and Andrew Mccallum. 2003. *Semi-supervised Clustering with User Feedback*. Technical Report.

[5] Ian Davidson. 2012. Two approaches to understanding when constraints help clustering. In *KDD*. 1312–1320.

[6] Ian Davidson and Sugato Basu. 2007. A Survey of Clustering with Instance Level Constraints . *TKDD* (2007).

[7] Ian Davidson and S. S. Ravi. 2005. Clustering with Constraints: Feasibility Issues and the k-Means Algorithm. In *SDM*. 138–149.

[8] Ian Davidson, S. S. Ravi, and Martin Ester. 2007. Efficient incremental constrained clustering. In *KDD*. 240–249.

[9] Eric Eaton, Marie desJardins, and Sara Jacob. 2014. Multi-view constrained clustering with an incomplete mapping between views. *Knowl. Inf. Syst.* 38, 1 (2014), 231–257.

[10] Ruizhang Huang and Wai Lam. 2007. Semi-supervised Document Clustering via Active Learning with Pairwise Constraints. In *ICDM*. 517–522.

[11] Pavan Kumar Mallapragada, Rong Jin, and Anil K. Jain. 2008. Active query selection for semi-supervised clustering. In *ICPR*. 1–4.

[12] Xuan Vinh Nguyen, Julien Epps, and James Bailey. 2009. Information theoretic measures for clusterings comparison: is a correction for chance necessary?. In *ICML*. 1073–1080.

[13] Dan Pelleg and Dorit Baras. 2007. $K$ -Means with Large and Noisy Constraint Sets. In *ECML*. 674–682.

[14] Sicheng Xiong, Javad Azimi, and Xiaoli Z. Fern. 2014. Active Learning of Constraints for Semi-Supervised Clustering. *IEEE Trans. Knowl. Data Eng.* 26, 1 (2014), 43–54.