Pattern Recognition 47 (2014) 1443–1458

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

# Active selection of clustering constraints: a sequential approach

Ahmad Ali Abin [*], Hamid Beigy

*Department of Computer Engineering, Sharif University of Technology, Azadi Avenue, Tehran, Iran*

ARTICLE INFO

*Article history:*
Received 20 January 2013
Received in revised form
21 August 2013
Accepted 30 September 2013
Available online 9 October 2013

*Keywords:*
Active constraint selection
Constrained clustering
Pairwise constraints
Data description

ABSTRACT

This paper examines active selection of clustering constraints, which has become a topic of significant interest in constrained clustering. Active selection of clustering constraints, which is known as minimizing the cost of acquiring constraints, also includes quantifying utility of a given constraint set. A sequential method is proposed in this paper to select the most beneficial set of constraints actively. The proposed method uses information of boundary points and transition regions extracted by data description methods to introduce a utility measure for constraints. Since previously selected constraints affect the utility of remaining candidate constraints, a method is proposed to update the utility of remaining constraints after selection of each constraint. Experiments carried out on synthetic and real datasets show that the proposed method improves the accuracy of clustering while reducing human interaction.

## 1. Introduction

Data clustering is an exploratory and descriptive data analysis technique with a long history in a variety of scientific fields [1]. It is presented with data instances that must be grouped according to a notion of similarity [2]. Clustering is fundamentally done with making some initial assumptions on distance metric, data structure, number of clusters, data distribution, and so on. If there is no correspondence between these assumptions and the actual model of clusters, the algorithm results in poor clusters. Recently, constrained clustering has become popular because it can take advantage of side information when available. Incorporating domain knowledge into the clustering by addition of constraints enables users to specify desirable properties of the result and also improves the robustness of the clustering algorithm.

There are several families of constraints but the instance-level constraints are the most used. Wagstaff et al. [2] introduced side information in two types of instance-level constraints: must-link (ML) and cannot-link (CL) constraints. A must-link (positive) constraint requires two objects to be grouped into the same cluster while a cannot-link (negative) constraint requires two objects to be put in different clusters. The inclusion of instance-level constraints allows the user to precisely state which objects should belong to the same cluster and which objects should not, without the need to explicitly state what these clusters are. In addition, the side information may occur at different levels such as class labels for a subset of objects, knowledge about clusters' position, clusters' identity, minimum or maximum size of clusters, and distribution of data [3].

Existing methods, which use constraints in the form of must-link and cannot-link constraints, can be grouped into two main categories: constraint-based and distance-based methods. In the first category [2,4], the constraints state whether two instances should be grouped into the same cluster or not, and the clustering algorithm is adapted so that the available constraints bias the search for a suitable clustering of data. Algorithms in the second category are initially trained to learn a proper distance measure satisfying the given constraints and then use the learnt measure for clustering of data. Recent techniques in this category include: joint clustering and distance metric learning [5], topology preserving distances metric learning [6], kernel approaches for metric learning [7], learning a margin-based clustering distortion measure using boosting [8], learning Mahalanobis distances metric [9,10], learning distances metric based on similarity information [11], and learning a distance metric transformation that is globally linear but locally non-linear [12], to mention a few. Use of pairwise constraints is not limited only to clustering applications. Several authors used pairwise constraints in semi-supervised feature selection [13], and dimensionally reduction [14], to mention a few.

While there is a large body of researches on constrained clustering algorithms [2,4–9,11,12,15,16], recently some more fundamental issues have emerged [17]. Three important issues in constrained clustering are: (1) quantifying the utility of a given constraint set, (2) minimizing the cost of constraint acquisition, and (3) propagating the constraint information to nearby regions in order to reduce the number of needed constraints [17]. The second issue comprises the first one and refers to minimizing the

* Corresponding author. Tel.: +98 21 66166698.
*E-mail addresses:* abin@ce.sharif.edu, ali_abin@yahoo.com (A.A. Abin), beigy@sharif.edu (H. Beigy).

0031-3203/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2013.09.034

cost of constraint acquisition. Existing methods in constrained clustering assumed that the algorithm is fed with a suitable passively chosen set of constraints [2,6,8,9]. They reported the clustering performance averaged over multiple randomly generated constraints. This is not always an applicable assumption. Randomly selected constraints do not always raise the quality of clustering results [18]. In addition, averaging over several trials is impossible in many problems because of the nature of the given problems or the cost of constraint acquisition. On the other hand, there are $\frac{1}{2}N(N-1)$ possible constraints on a dataset with $N$ instances, and constraint specification can be burdensome for large datasets. An alternative way to easily find the most beneficial constraints is to actively acquire them. There are small range of studies on active selection of clustering constraints, which include active selection of constraints based on: “farthest-first” strategy [19], hierarchical clustering [20], theory of spectral decomposition [21], fuzzy clustering [22], Min–Max criterion [23], and graph theory [24]. The performance of each of them highly depends on the underlying assumptions like the data structure, the distance metric and so on. These methods ignore the effect of previously chosen constraints on the utility of remaining candidate constraints but this paper directly addresses the constraint utility dependencies as an important issue in constrained clustering.

This paper proposes a method to improve active selection of clustering constraints. Our proposed method is based on the sequential selection of constraints such that the selection heuristic takes into account the already chosen constraints. The information of boundary points and transition regions of data is used to introduce a time-varying utility measure for constraints. The efficiency of the selected constraints is evaluated in conjunction with some constrained clustering algorithms on some artificial and real datasets. Experimental results show the superiority of the proposed method on the chosen constraints, which increase the accuracy of methods in constrained clustering.

The rest of this paper is organized as follows. The related work is discussed in Section 2. The proposed constraint selection method is given in Section 3. Experimental results are presented in Section 4. This paper concludes with conclusions and future works in Section 5.

## 2. Related work

*Active learning* has a long history in supervised learning algorithms. The key idea behind active learning is that a machine learning algorithm can perform better with less training if it is allowed to query the labels of some instances. In the statistics literature active learning is sometimes also called optimal experimental design [25]. Recently, active learning approaches are used in constrained clustering problem. Few studies reported the result of using active learning for constrained clustering [19–24,26]. Klein et al. [20] suggested an active constraint selection method in which a hierarchical clustering algorithm identifies the $m$ best queries that should be asked from an expert.

Basu et al. [19] proposed an algorithm for active selection of constraints using the farthest first query selection (FFQS) algorithm. FFQS has two phases: *Explore* and *Consolidate*. Let $K$ be the true number of clusters in a dataset. The exploration phase explores the given data to find $K$ pairwise disjointed non-null neighborhoods (as skeleton of the clusters), belonging to different clusters. A preference to cannot-link queries is given by choosing the farthest point from the existing disjointed neighborhoods. The exploration will continue until $K$ points are found such that there is a cannot-link between each pair of these $K$ points. This phase is then followed by the consolidate phase. The consolidate phase selects non-skeleton points randomly and queries them against each point in the skeleton, until a must-link query is obtained.

Mallapragada et al. [23] generalized FFQS by introducing Min–Max criterion. Their method (referred to as MMFFQS) altered the consolidate phase of FFQS. The exploration in MMFFQS is done in the same way as FFQS but the random point selection of the consolidation phase is replaced with selection of data point with maximum uncertainty in cluster assignment. Both FFQS and MMFFQS have problem with unbalanced datasets or datasets containing a large number of clusters [24].

ACCESS [21] is an active constrained clustering technique which examines the eigenvectors derived from similarity matrix of data. ACCESS uses the theory of spectral decomposition to identify data items that are likely to be located on boundaries of clusters, and for which providing constraints can resolve ambiguity in the clustering. ACCESS identifies two types of informative points: (1) sparse points and (2) close and distant boundary points. The sparse points are identified by evaluating the first $m$ eigenvectors of the similarity matrix (where $m$ depends on how many sparse sub-clusters are found in the dataset), and the close and distant boundary points are identified by evaluating the $(m+1)$th eigenvector of the similarity matrix. These informative points are then used by ACCESS for active selection of constraints. However, limitation of ACCESS on problems with two clusters can be mentioned as an important shortcoming.

AFCC [22] as another active constrained clustering method minimizes a competitive cost function with fuzzy terms corresponding to pairwise constraints. AFCC uses an active method based on the least well-defined cluster to find the most informative must-link or cannot-link constraints. Fuzzy hyper-volume measure is used by AFCC to identify the least well-defined cluster and objects located on frontier of this cluster. For each object lying on the frontier, the closest cluster corresponding to its second highest membership value is found and the user is then asked whether one of these objects should be (or not) in the same cluster as the nearest object from the closest cluster. AFCC will fail when there are clusters with complex structure (shapes, distributions) in data.

Vu et al. [24,26] proposed an active query selection method based on the ability to separate between clusters. We refer to this algorithm as ASC. ASC has three basic steps: (1) the best candidate query in sparse regions of the dataset is determined by a $k$-nearest neighbor graph, (2) a constraint utility function is used in a query selection process, and (3) a propagation procedure is used to propagate each query to generate several constraints and limit the number of candidate constraints. The propagation procedure discovers new constraints from the information stored in already chosen constraints using the notion of strong paths. Subsequently, the size of the candidate set is reduced by a refinement procedure that removes constraints between objects that are likely to be in the same cluster. Specifically, the refinement procedure removes candidate constraints that are linked by a strong path.

The above-mentioned methods form a range of studies performed in active selection of clustering constraints. Each method considered a basic assumption on utility of constraints. Their applicability on a specific problem is highly dependent on correlation between their assumption and the actual structure of data.

## 3. The proposed active constraint selection method

In the literature, several methods have been proposed for active constraint selection but they ignore the constraint utility dependencies. In this section, we propose a sequential approach for active constraint selection (SACS) that considers dependencies among the constraints. The proposed approach is based on the following two assumptions.

**Assumption A.** Constraints will be more informative if the involved points are close together while they are from different disjointed regions of a multi-modal data distribution.

**Assumption B.** Constraints will be more informative if the involved points are distant and have major part of their length passing through the same region while they are from the same region of a data distribution.

Assumptions A and B are applicable to many problems. Assumption A implies that transition regions contain useful information about different disjointed regions and can help to identify relationship between disjointed regions of a data distribution. Querying from these regions clarifies some ambiguities of clustering algorithms in identification of cluster boundaries. On the other hand, Assumption B implies that long constraints contain useful information about internal structure of a region if the involved points are selected from the same region of data and have major part of their length passing through the region. Fig. 1 shows the applicability of these assumptions on a typical dataset. As shown in this figure, the involved points are selected from the boundary region of data, which helps SACS to query ambiguous transition regions of the data distribution (see constraints 1–3) or ask about region continuity (see constraints 4–6). SACS chooses constraints based on Assumptions A and B with *Assumption contribution factors* $\alpha$ and $1-\alpha$, respectively. This gives the proposed method the ability to choose $\alpha\lambda$ well-distributed inter-region constraints and $(1-\alpha)\lambda$ well-distributed intra-region constraints, where $\lambda$ is a user specified parameter that shows the number of constraints to be selected.

Based on these assumptions, a utility measure is proposed that shows the effectiveness of the selected constraints. In order to avoid selecting inefficient or redundant constraints, this utility measure will be updated after each constraint selection step. The proposed method has the following three basic phases: (1) finding the boundary points of data, (2) measuring the utility of constraints, and (3) choosing constraints sequentially. These phases are described in the following sections.

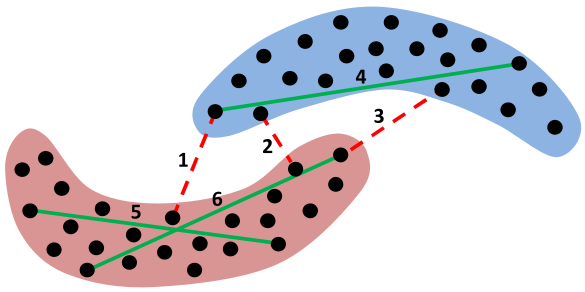

**Fig. 1.** Applicability of Assumptions A and B on a typical dataset.

### 3.1. Finding the boundary points of data

This phase is going to find the boundary of data. Any boundary detection method can be used to find it. Here, the idea of data domain description is used to find the boundary points. The most common approaches to data domain description are probabilistic in nature [27,28], which model the distribution of data by a probability density function (pdf) and specify a suitable threshold on such pdf to determine the description boundary. A drawback of these methods is that they often require large datasets, especially when high dimensional data is given. In addition, some problems may arise when large differences exist among densities of different areas. Alternative methods in data domain description do not rely on a probabilistic approach. Instead, they aim to minimize the volume of the description domain, which may be cast out in the form of linear programming [29] or quadratic programming [30]. Usually such methods can be applied to any distribution, as they do not make assumptions on distribution of data. These methods are robust against outliers in training set and are capable of tightening the description domain [31].

Support vector data description (SVDD) [30] is a data description method defined in the framework of kernel methods. SVDD casts out the problem of data description as a quadratic programming and minimizes the volume of description domain. SVDD obtains a spherically shaped boundary around the dataset by: (1) applying a non-linear transformation of data from *input space* to a high dimensional *feature space*, (2) looking for smallest enclosing sphere of radius $R$ in feature space (Fig. 2(b)), and (3) backing the enclosing sphere into the input space to result in the description boundary of data (Fig. 2(c)). Points outside the description domain in input space (outside the enclosing sphere in feature space) will be referred to as *bounded support vectors* or BSVs and points on the boundary of description domain (on the surface of the feature space sphere) will referred to as support vectors (SVs). SVDD has this advantage that it makes no assumption on data distribution. Moreover, it is robust against outliers and is capable of tightening the description domain.

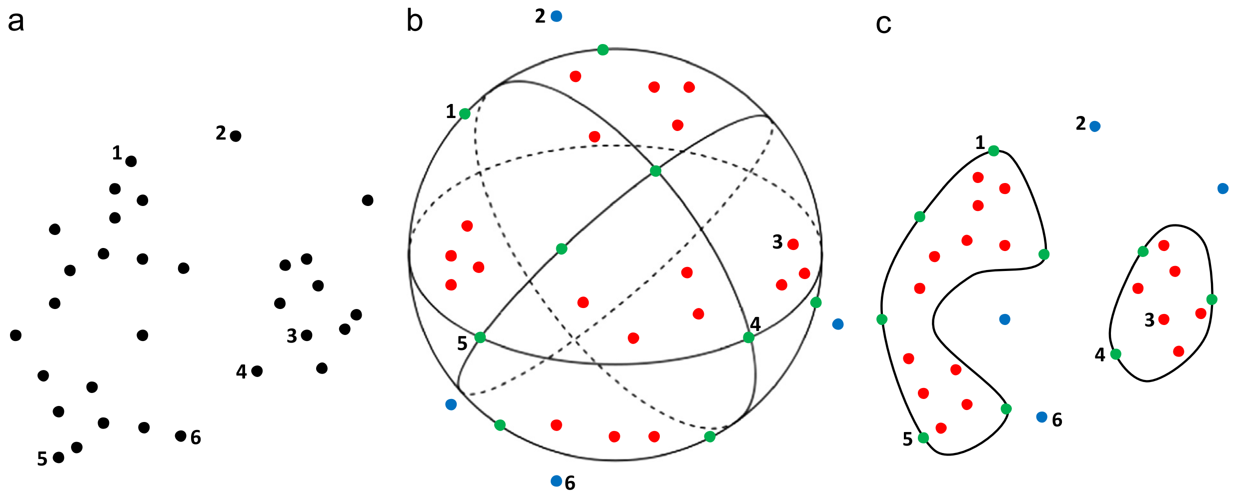

**Fig. 2.** (a) Synthetic data points in input space $\mathbb{R}^2$, (b) non-linear transformation of the data points to high dimensional feature space and smallest enclosing sphere of radius $R$, and (c) backing the enclosing sphere into the input space to result in description boundary of data.

Here, the idea of SVDD is used to find the boundary of a given data distribution. Let $N$ be the total number of data points and $N_{sv}$ ($N_{sv} \leq N$) be the number of support vectors found by SVDD. The proposed method only allows the $N_{sv}$ support vectors to participate in constraint selection. This not only reduces the total number of candidate constraints from $\binom{N}{2}$ to $\binom{N_{sv}}{2}$ but also reduces computational cost of selecting constraints.

### 3.2. Measuring the utility of constraints

At the end of the first step, we have $\binom{N_{sv}}{2}$ candidate constraints. These constraints have different utilities. Hence, we need a criterion to measure the effectiveness of each constraint. In this section, we introduce a measure to evaluate *individual utility* for each constraint without considering the effect of any other constraints. Let $S$ be a set of points chosen at the first step and $(i,j)_{i \neq j} \in S \times S$ be a constraint between two data points $x_i$ and $x_j$. Let $d_{ij}$ be the length of constraint $(i,j)_{i \neq j}$ and $v_{ij}^{out} = d_{ij}^{out}/d_{ij}$ be the fraction of $d_{ij}$ outside the description domain (see Fig. 3(a)), where $d_{ij}^{out}$ denotes a part of constraint $(i,j)$ that is outside the description domain.

As mentioned at the beginning of Section 3, Assumption A aims to choose constraints most similar to cannot-link constraints to ask about the transition regions of data. Based on this assumption, a shorter constraint with a greater value of $v_{ij}{}^{out}$ (constraint (1,2) in Fig. 3(b)) is superior than other constraints (e.g. constraint (3,4) or (6,7) in Fig. 3 (b)) to be selected. Based on this assumption, the individual utility $u_{ij}^{indA}$ of constraint $(i,j)$ is calculated by the following rule.

$$u_{ij}^{indA} = v_{ij}^{out} \times \left(1 - \frac{d_{ij}}{\max_{(k,l)_{k \neq l} \in S \times S} d_{kl}}\right). \tag{1}$$

The second term of Eq. (1) indicates that a shorter constraint has more utility $u_{ij}^{indA}$ than a longer one (e.g. constraint (1,2) against (3,4) in Fig. 3(b)). On the other hand, the first term of Eq. (1) implies that a constraint with the greater value of $v_{ij}^{out}$ will be more valuable because such a constraint gives information about two disjointed regions without intermediate regions (e.g. constraint (6,7) against (5,6) in Fig. 3(b)). Together, Assumption A gives greater value of $u_{ij}^{indA}$ to a constraint with the greater value of $v_{ij}^{out}$ and the smaller value of $d_{ij}$.

Unlike the first assumption, which aims to pick out the constraints from transition regions of data, Assumption B chooses intra-region informative constraints. Based on this assumption, a constraint will be more informative if major part of its length passes through the same region of data (constraint (1,2) in Fig. 3 (c)). The focus of this assumption is to choose constraints most similar to must-link constraints to discover the global structure of a given dataset. Therefore, a longer constraint with a smaller value of $v_{ij}{}^{out}$ (e.g. constraint (1,2) in Fig. 3(c)) is superior than other constraints (e.g. constraint (1,8) or (6,7) in Fig. 3(b)) to be selected. Based on this assumption, the individual utility $u_{ij}^{indB}$ of constraint $(i,j)$ is calculated using the following rule.

$$u_{ij}^{indB} = (1 - v_{ij}^{out}) \times \frac{d_{ij}}{\max_{(k,l)_{k \neq l} \in S \times S} d_{kl}}. \tag{2}$$

The second term of Eq. (2) implies that a longer constraint has more utility $u_{ij}^{indB}$ than the shorter one (e.g. constraint (1,2) against (6,7) in Fig. 3(c)). On the other hand, the first term of Eq. (2) implies that a constraint with the smaller value of $v_{ij}^{out}$ is more valuable because it provides more information about the inner structure of a region (e.g. constraint (1,2) against (1,8) in Fig. 3(c)). Together, Assumption B gives a greater value of $u_{ij}^{indB}$ to a constraint with the smaller value of $v_{ij}^{out}$ and the greater value of $d_{ij}$.

### 3.3. Sequential selection of constraints

In the previous section, each constraint is assigned an individual utility $u_{ij}^{ind.}$ based on Assumptions A and B without considering the dependency among constraints. Since the previously selected constraints have an influence on subsequent choices, the utility of candidate constraints should be updated after each constraint selection. In this section, we are going to update the individual utility of candidate constraints having the already selected constraints. Constraint utility dependencies are fully illustrated in Fig. 4. Let constraints 1, 2 and 3 have individual utilities 10, 8, and 5, respectively. Choosing constraint 1 will decrease the utility of constraint 2 as second option because its information exists somehow in constraint 1. Hence, constraint 3 with less individual utility than constraint 2 will be superior to be selected as the second most beneficial constraint. Therefore, the individual utility measures introduced in Section 3.2 should be reformulated in such a way that takes into account the constraint utility dependencies.

Let $C^t$ be $t$ selected constraints up to step $t$ and $d_{ijkl}$ be the distance between two constraints $(i,j)$ and $(k,l)$. The distance between a candidate constraint $(i,j)$ and constraints set $C^t$ is

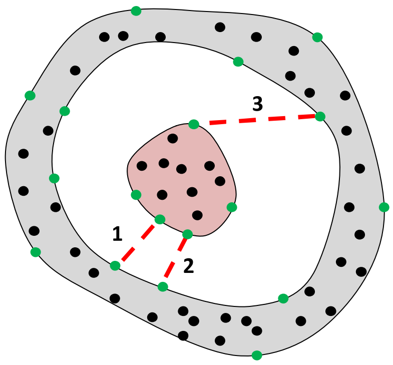

**Fig. 4.** Constraints 1, 2 and 3 have utilities 10, 8, and 5, respectively. Choosing constraint 1 will have an effect on the utility of constraints 2 and 3 as the next most beneficial constraint.

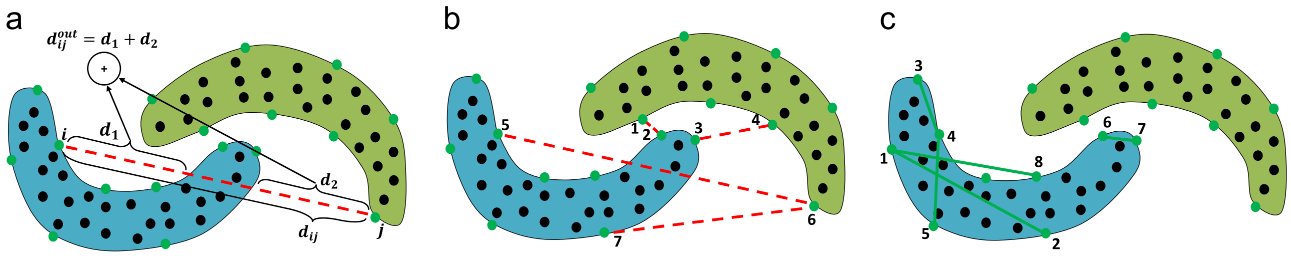

**Fig. 3.** (a) $d_{ij}^{out}$ as the part of constraint $(i,j)$ that is outside the description domain, (b) constraints with the greater value of $v_{ij}^{out}$ and the smaller value of $d_{ij}$ are assigned the greater value of $u_{ij}^{indA}$ based on Assumption A, and (c) constraints with the smaller value of $v_{ij}^{out}$ and the greater value of $d_{ij}$ are assigned the greater value of $u_{ij}^{indB}$ based on Assumption B.

denoted by $d_{ijC^t} = \min_{(k,l) \in C^t} d_{ijkl}$. Let $u_{ij}^{ind.}$ be individual utility assigned to constraint $(i,j)$ based on Assumptions A and B. The following equation is proposed to measure the utility of constraint $(i,j)$ at step $t+1$ in the presence of selected constraints $C^t$.

$$u_{ij}^{t+1} = u_{ij}^{ind.} \times \frac{d_{ijC^t}}{\max_{(k,l)_{k \neq l} \in S \times S} d_{klC^t}}. \quad (3)$$

Eq. (3) implies that constraints more away from $C^t$ are more valuable to be selected in the following step. This expression justifies the choice of constraint 3 as the second option in Fig. 4. Starting from $u_{ij}^{t=1} = u_{ij}^{ind.}$, the sequential selection of constraint with the highest-utility is repeated until $\lambda$ constraints are selected. Moreover, greedy selection may not result in the maximum *total utility* at $\lambda$ steps. To reach the maximum total utility, the following objective function must be optimized.

$$\arg\max_{C^\lambda}(\textit{total utility}) = \arg\max_{C^\lambda}\left(\sum_{t=1}^{\lambda} u_{ij}^t\right), \quad (4)$$

where $u_{ij}^t$ is the utility of constraint $(i,j)$ selected at step $t$. It should be noted that this function is sensitive to the sequence of choices. Finding sequence with the highest total utility requires all $\lambda$ permutations of $\binom{N_{sv}}{2}$ candidate constraints to be considered. The proposed method is presented in Algorithm 1, where the constraints are chosen based on two Assumptions A and B with the assumption contribution factors $\alpha$ and $1-\alpha$, respectively.

**Algorithm 1.** Sequential approach for active constraint selection (SACS).

```
1:  procedure GetConstraints $(X, \lambda, \alpha)$        ▹ $X = \{x_i\}_{i=1}^N, \alpha \le 1$
2:      $S \leftarrow$ set of support vectors identified by SVDD on dataset $X$
3:      $t \leftarrow 0$
4:      $C^t \leftarrow \emptyset$
5:      $\forall (i,j)_{i \neq j} \in S \times S$, set $u_{ij}^t = u_{ij}^{indA}$ via Eq. (1)
6:      while $t \le \alpha\lambda$ do
7:          $(i,j) \leftarrow \mathrm{argmax}_{(i,j)_{i \neq j \& (i,j) \notin C^t} \in S \times S} u_{ij}^t$
8:          $C^{t+1} \leftarrow C^t \bigcup \{(i,j)\}$
9:          $\forall (i,j)_{i \neq j \& (i,j) \notin C^{t+1}} \in S \times S$, update $u_{ij}^t$ to $u_{ij}^{t+1}$ via Eq. (2)
10:         $t \leftarrow t+1$
11:     end while
        }  Choosing $\alpha\lambda$ constraints based on Assumption A.
12:     $\forall (i,j)_{i \neq j} \in S \times S$, set $u_{ij}^t = u_{ij}^{indB}$ via Eq. (3)
13:     while $t \le \lambda$ do
14:         $(i,j) \leftarrow \mathrm{argmax}_{(i,j)_{i \neq j \& (i,j) \notin C^t} \in S \times S} u_{ij}^t$
15:         $C^{t+1} \leftarrow C^t \bigcup \{(i,j)\}$
16:         $\forall (i,j)_{i \neq j \& (i,j) \notin C^{t+1}} \in S \times S$ update $u_{ij}^t$ to $u_{ij}^{t+1}$ via Eq. (3)
17:         $t \leftarrow t+1$
18:     end while
        }  Choosing $(1-\alpha)\lambda$ constraints based on Assumption B.
19:     return $C^\lambda$        ◃ return $\lambda$ sequentially selected constraints
20: end procedure
```

## 4. Experimental results

In order to evaluate the proposed method, three experiments are conducted. The effect of SVDD parameters on the efficiency of the selected constraints is studied in the first experiment. In the second one, the proposed method is compared with other constraint selection methods. In the third experiment, the proposed method is evaluated in conjunction with three different constraint-based clustering algorithms to show its adaptability. In all experiments, the assumption contribution factor $\alpha$ was set to $\frac{1}{2}$. Euclidean distance function $d_{ij} = \|x_i - x_j\|_2$ is used to measure the length of constraint $(i,j)$. It can be replaced by other distance functions like Mahalanobis or graph-based distance functions. The distance between two constraints $(i,j)$ and $(k,l)$ is defined as $d_{ijkl} = \min\{(d_{ik}+d_{jl}), (d_{il}+d_{jk})\}$ in all experiments.

The rest of this section is organized as follows. Section 4.1 describes some constraint selection heuristics that are compared with the proposed method. Section 4.2 mentions the datasets used for evaluating the proposed method and Section 4.3 describes different constraint-based clustering algorithms used for constraints evaluation. Section 4.4 explains the evaluation criterion used in this paper. Experiments 1–3 are described in Sections 4.5–4.7, respectively.

### 4.1. Constraint selection heuristics

In all experiments, we consider five heuristics to select the constraints: random selection of constraints by generating ML and CL constraints based on the comparison of the labels of involved points denoted as Random, FFQS [19], MMFFQS [23], ASC [24], and the proposed method. As Random, FFQS, and MMFFQS are non-deterministic, the associated results are averaged over 50 runs for each dataset.

### 4.2. Datasets

Experiments are conducted on 11 datasets from UCI Machine Learning Repository[1] (each with the following number of objects,

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

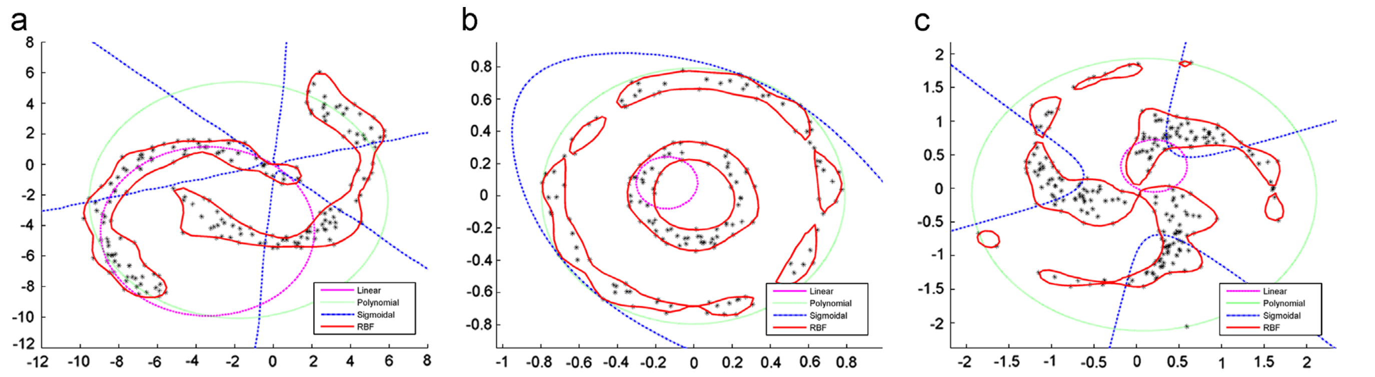

**Fig. 5.** Different kernel functions result in different description boundaries in the original input space. (a) Bananas dataset generated by uniformly distributing 100/100 data along the bananas and superimposing with a normal distribution in all directions, (b) two concentric rings contain 85/85 points, generated from two uniform angular distributions, and (c) pinwheel dataset generated by taking Gaussian distributions, stretching and rotating 80/80/80 points appropriately.

attributes and clusters): Iris (150/4/3), Glass (214/9/6), Breast (569/30/2), Soybean (47/34/4), Protein (116/20/6), Wine (178/13/3), Sonar (208,60,2), Heart (270,13,2), Balance (625/4/3), Diabetes (768/8/2), and Image Segmentation (2100/19/7). These datasets were chosen because they have already been used in constrained clustering researches.

### 4.3. Constraint-based clustering algorithms

In all experiments, we report the obtained results in conjunction with three different constraint-based clustering algorithms: K-Means + metric learning (Xiang) [9], K-Means + Relevant Component Analysis (RCA) [32], and MPC-KMeans [33], where $\mathcal{A}+\mathcal{B}$ means integration of clustering algorithm $\mathcal{A}$ and constraint-based metric learning strategy $\mathcal{B}$. They have been chosen because they are representative of popular clustering algorithms. Xiang's method learns a Mahalanobis metric for data, RCA looks relevant components for data using side-information in the form of equivalence constraints, and MPC-KMeans integrates metric learning and data clustering in a united framework.

### 4.4. Evaluation method

The adjusted Rand index (ARI) [34] is used in all experiments to evaluate agreement between true partition of a dataset ($P_1$) and output partition of the evaluated clustering algorithm ($P_2$). Let $N_{ij}$ be the number of items that appear in cluster $i$ in $P_1$ and in cluster $j$ in $P_2$. ARI is computed as follows:

$$ARI(P_1,P_2)=\frac{R-E[R]}{M[R]-E[R]}, \tag{5}$$

where $R=\Sigma_{ij}\binom{N_{ij}}{2}$, $E[R]=\left[\Sigma_i\binom{N_i}{2}\Sigma_j\binom{N_j}{2}\right]/\binom{N}{2}$ is the expected value of $R$ and $M[R]=\frac{1}{2}\left[\Sigma_i\binom{N_i}{2}+\Sigma_j\binom{N_j}{2}\right]$ is the maximum possible value for $R$ [34].

### 4.5. Experiment 1: the effect of SVDD parameters on the efficiency of selected constraints

In this experiment, the proposed method is investigated from type and parameters of kernel function used in SVDD. Different kernel functions result in different description boundaries in the original input space. The problem is to find a suitable kernel function $K(x_i,x_j)$, where $x_i$ and $x_j$ are vectors in the input space. We consider four kernel functions: (1) a linear kernel $K(x_i,x_j)=(x_i.x_j)$, (2) a polynomial kernel $K(x_i,x_j)=(x_i.x_j+r)^d$, where $r$ is a constant trading off the influence of higher-order versus lower-order terms in the polynomial and $d$ is the degree of the polynomial, (3) a sigmoid kernel $K(x_i,x_j)=\tanh(ax_i.x_j+r)$, where $a$ is a scaling parameter of the input data, and $r$ is a shifting parameter that controls the threshold of the mapping, and (4) an RBF (Gaussian) kernel $K(x_i,x_j)=e^{-\gamma\|x_i-x_j\|^2}$, where $\gamma=1/2\sigma^2$ and $\sigma$ is the scale parameter of the kernel. Linear and polynomial kernels, in general, do not result in good tight descriptions. For higher degrees $d$, the influence of objects most remote from the origin of the coordinate system increases and overwhelms all other inner products. On the other hand, a sigmoid kernel dampens this effect by reducing the influence of remote objects but ignores distinction among remote objects significantly. These effects on description boundaries are shown in Fig. 5 for linear kernel, polynomial kernel ($r=1,d=2$), sigmoid kernel ($a=3,r=0.5$), and RBF kernel ($\sigma=\sigma_X/2$), where $\sigma_X$ is the standard deviation of all $\binom{N}{2}$ pairwise distances among points in dataset $X=\{x_i\}_{i=1}^N$.

As Fig. 5 shows, linear, polynomial and sigmoid kernels do not result in good tight descriptions in original input space. A similar result is also concluded in [30]. To suppress the growing distances for larger feature spaces, RBF kernel is chosen to be used by SVDD in all remaining experiments.

The shape of the enclosing contours in input space is governed by two SVDD parameters: $\sigma$, the scale parameter of RBF kernel, and $C$, the soft margin constant. In the experiments conducted in this section, we will study the effects of these parameters on the performance of the selected constraints. In real data, clusters are usually not as well separated. Thus, to observe splitting of contours, one can allow for bounded support vectors (BSVs). The number of outliers is controlled by the parameter $C$. Only the choices for which the soft margin constant $C$ can have any influence on the description boundary are when $1/N\le C\le 1$. For $C<1/N$ no solution can be found, while for $C>1$ one can always find a solution [30]. For $C=1$ outliers are not allowed to be emerged. When $C$ is restricted to small values in SVDD, the cost of being outside the sphere is not very large and a larger fraction of the data are allowed to be outside the sphere. In practice, the value of $C$ is not very critical as described in [30]. As a result, for fixed value of $\sigma$, as $C$ is decreased, the number of SVs decreases because some of them turn into BSVs and the contours become smoother as illustrated in Fig. 6. Fig. 7 illustrates the performance effect of different values of $C$ specifying $\sigma=\sigma_X/2$ for UCI datasets. Fig. 7 (a) shows this effect on clustering accuracy ARI with $\lambda=25$ constraints in conjunction with the Xiang method [9]. Fig. 7 (b) and (c) illustrate changes in rates $N_{sv}/N$ and $N_{bsv}/N$ of SVDD, respectively. As Fig. 7 shows, the value of $C$ is not critical and $C>0.7$ results in near the same clustering accuracy, $N_{sv}/N$, and $N_{bsv}/N$ in all datasets. However, to consider the effect of probable outliers in real datasets, $C$ is set to 0.95 in all remaining experiments.

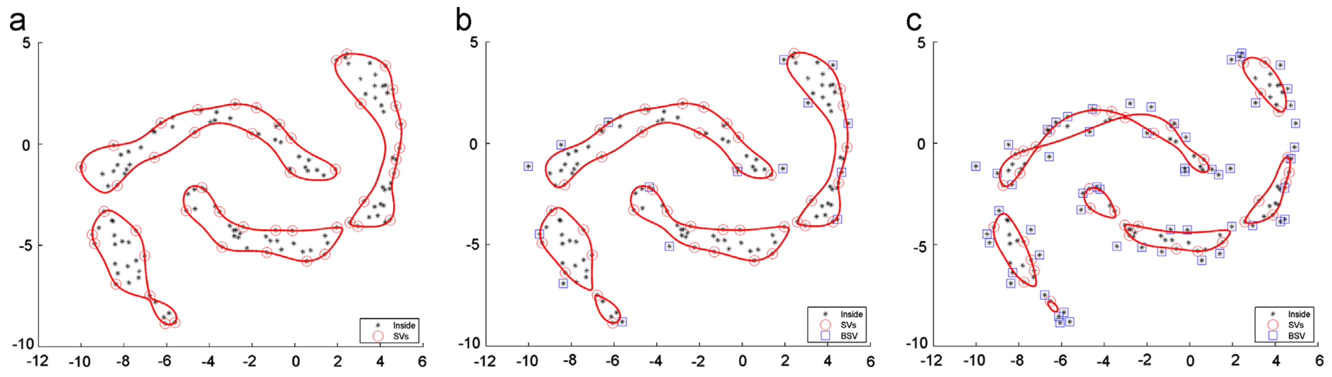

**Fig. 6.** Description of 70/70 bananas data using SVDD with a fixed value of $\sigma = \sigma_X/2$: (a) $C = 1, N_{sv} = 48, N_{bsv} = 0$, (b) $C = 0.7, N_{sv} = 39, N_{bsv} = 16$, and (c) $C = 0.2, N_{sv} = 37, N_{bsv} = 58$.

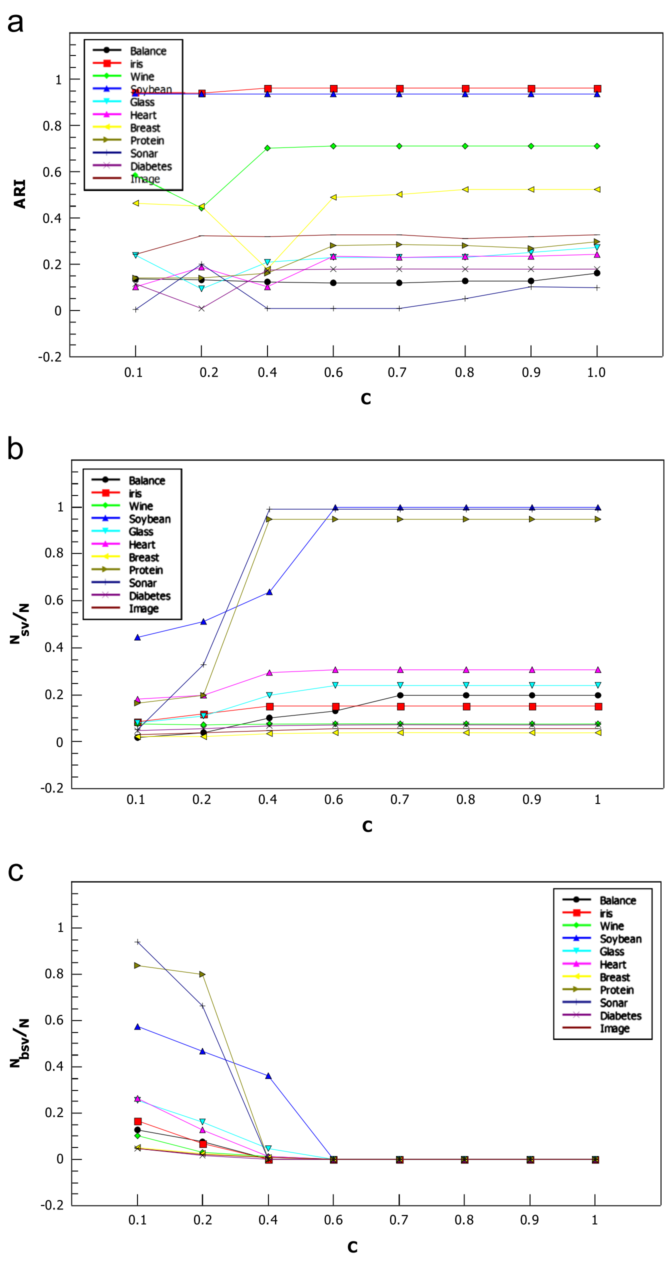

**Fig. 7.** The effect of different values of $C$ for fixed value of $\sigma = \sigma_X/2$ on: (a) clustering accuracy ARI with $\lambda = 25$ constraints selected by the proposed method in conjunction with the Xiang method [9], (b) changes in rate $N_{sv}/N$, and (c) changes in rate $N_{bsv}/N$.

The parameter $\sigma$ defines the scale parameter of the RBF Kernel. For a fixed value of $C$, as $\sigma$ is decreased, the boundary fits more tightly the data, and for small values of $\sigma$, the enclosing contour splits and forms several disjoint components. Fig. 8 shows the effect of different values of $\sigma$ on the boundary of a typical dataset. Experiments on synthetic datasets show that $\sigma \in [\sigma_X/4, \sigma_X]$ gives a good description of data, including a reasonable number of components as was compared in Fig. 8. The effectiveness of $\sigma \in [\sigma_X/4, \sigma_X]$ was also evaluated on mentioned UCI datasets and illustrated in Fig. 9. Fig. 9(a) shows the clustering efficiency with $\lambda$=25 constraints in conjunction with the Xiang method [9]. Fig. 9(b) shows the total utility obtained by Eq. (4) with $\lambda$=25 constraints selected by the proposed method. Fig. 9(c) shows changes in $N_{sv}/N$ on different values of $\sigma$.

Roughly speaking, there is a maximum accuracy when $\sigma \in [\sigma_X/4, \sigma_X]$ in all datasets as Fig. 9(a) shows. In the other ranges of $\sigma$, the Xiang method results in poor performance because of the low quality of selected constraints caused by poor description of data. A noticeable point in Fig. 9(b) is that, there is a drop in total utility when $\sigma > \sigma_X$ in all datasets, which makes $\sigma \in [\sigma_X/4, \sigma_X]$ as an appropriate choice for the scale parameter of RBF kernel. Fig. 9(c) also compares changes in $N_{sv}/N$ on different values of $\sigma$. As shown in this figure, for a fixed value of $C$, as $\sigma$ increases, the number of support vectors decreases because of the looser boundary around the data. Considering results illustrated in Fig. 9, $\sigma = \sigma_X/2$ is chosen to evaluate the proposed method in all experiments. As this figure shows, $\sigma = \sigma_X/2$ not only results well in ARI and total utility but also generates a reasonable number of support vectors.

## 4.6. *Experiment 2: investigating the behavior of the proposed method in constraint selection*

This experiment compares the behavior of three FFQS, ASC and the proposed methods in constraint selection on some synthetic datasets. Fig. 10 shows this experiment providing each figure with the method and total utility obtained by Eq. (4) in its caption. For FFQS and ASC methods, the total utilities are calculated for all permutations of constraints on the data domain described by SVDD and the maximum total utility is reported. As Fig. 10 shows, the proposed method results in a well-distributed set of constraints against FFQS and ASC heuristics. A noteworthy point in Fig. 10 is the distribution of constraints selected by the proposed method as the entire space has been spanned. Uniform distribution of constraints causes the constraints to provide useful information about the whole structure and actual distribution of data. Although FFQS and ASC may achieve such results at a large number of queries,

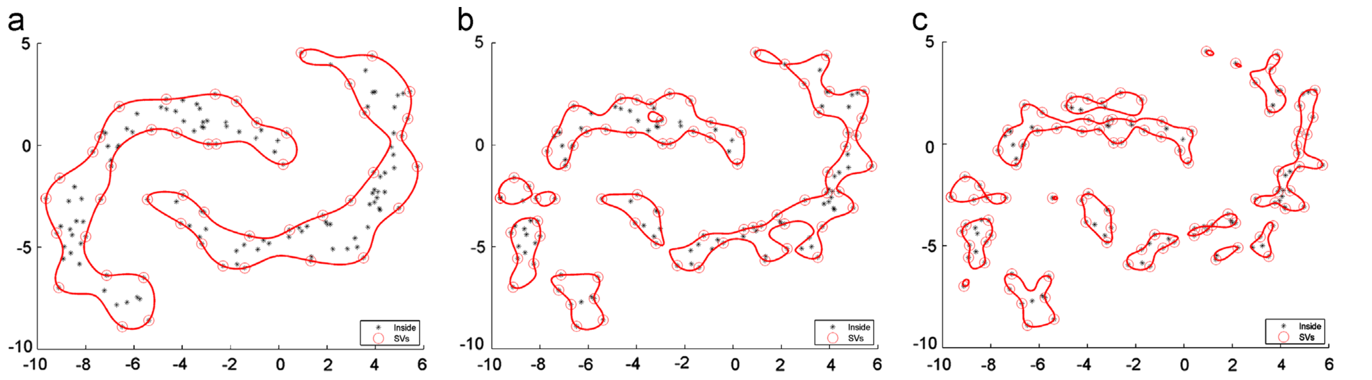

**Fig. 8.** Description of 70/70 bananas data using SVDD for different values of $\sigma$ and fixed value of $C=0.95$: (a) $\sigma=\sigma_X/2, N_{sv}=47, N_{bsv}=0$, (b) $\sigma=\sigma_X/4, N_{sv}=74, N_{bsv}=0$, and (c) $\sigma=\sigma_X/6, N_{sv}=97, N_{bsv}=0$.

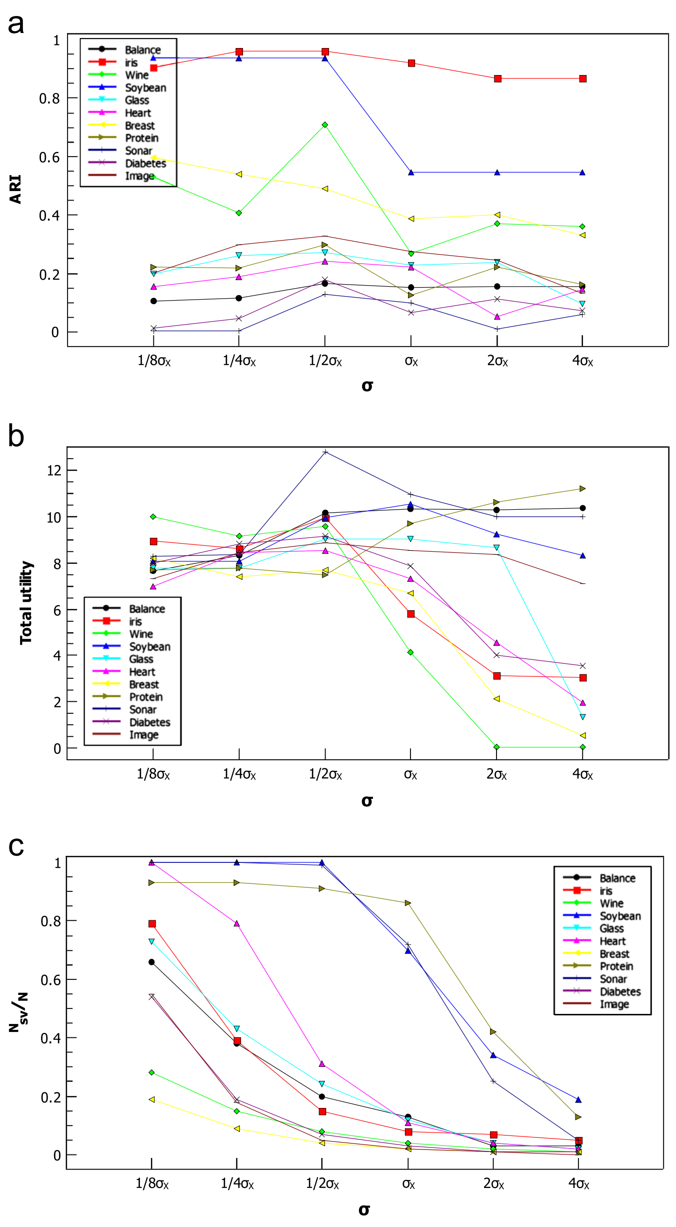

**Fig. 9.** Effectiveness of different values of $\sigma$specifying $C=0.95$ on: (a) clustering efficiency with $\lambda=25$ constraints selected by the proposed method in conjunction with the Xiang method [9], (b) total utility with $\lambda=25$ constraints selected by the proposed method obtained by Eq. (4) and, (c) changes in $N_{sv}/N$ from SVDD.

but a prominent feature of the proposed method is the high-quality description of data structure when the number of queries is low.

### *4.7. Experiment 3: comparison of the proposed method with some related methods*

The evaluation results of the proposed method in comparison with Random, FFQS, MMFFQS and ASC heuristics are presented in this section. Figs. 11–16 show these evaluation results. To evaluate the efficiency of the proposed method to adapt according to distinct clustering algorithms, it is compared in conjunction with three constraint selection algorithms Xiang, RCA, and MPC-KMeans. Each plot in Figs. 11–16 is provided with the dataset name and clustering algorithm (dataset name $\oplus$ clustering algorithm) in its caption. ARI of a classical K-Means clustering algorithm, averaged over 50 runs on benchmark datasets is also presented as a baseline in all experiments. For each dataset and each number of queries, the results of the non-deterministic methods such as Random, FFQS, and MMFFQS are averaged over 50 runs. Parameters $k$ and $\theta$ are set to 6 and $\lfloor k/2+1 \rfloor$ in ASC heuristic, respectively. The assumption contribution factor $\alpha$ is set to $\frac{1}{2}$ and the SVDD scale parameter $\sigma$ and soft margin $C$ are also set to $\frac{\sigma_X}{2}$ and 0.95 in the proposed method, respectively.

It can be observed from Figs. 11–16 that the proposed method generally outperforms other constraint selection heuristics in conjunction with three clustering algorithms (Xiang, RCA, and MPC-KMeans) except in Protein $\oplus$ RCA (Fig. 12(d)) and Diabetes $\oplus$ MPC−KMeans (Fig. 16(c)). On the other hand, the constraints selected by the proposed method make better results in Protein $\oplus$ Xiang (Fig. 12(b)), Protein $\oplus$ MPC−KMeans (Fig. 12(f)), and Diabetes $\oplus$ Xiang (Fig. 16(a)). This implies that the usefulness of constraints depends on how they are utilized by a clustering algorithm. Furthermore, the proposed method keeps a smooth increase in efficiency in Protein $\oplus$ RCA while the other constraint selection heuristics drop in efficiency when the number of queries increases.

More generally, it can be seen from Figs. 11–16 that the proposed method outperforms FFQS in conjunction with Xiang, RCA, and MPC-KMeans except in Protein $\oplus$ RCA (Fig. 12(d)), Diabetes $\oplus$ RCA (Fig. 16(b)) and Diabetes $\oplus$ MPC−KMeans (Fig. 16(c)). On the other hand, there are drops in efficiency in both Protein $\oplus$ RCA and Diabetes $\oplus$ RCA for $\lambda=50$ constraints that causes FFQS to be less accurate than the proposed method. In addition, the proposed method obtains a better result than FFQS with $\lambda=10$ constraints in Protein $\oplus$ RCA. Although, the proposed

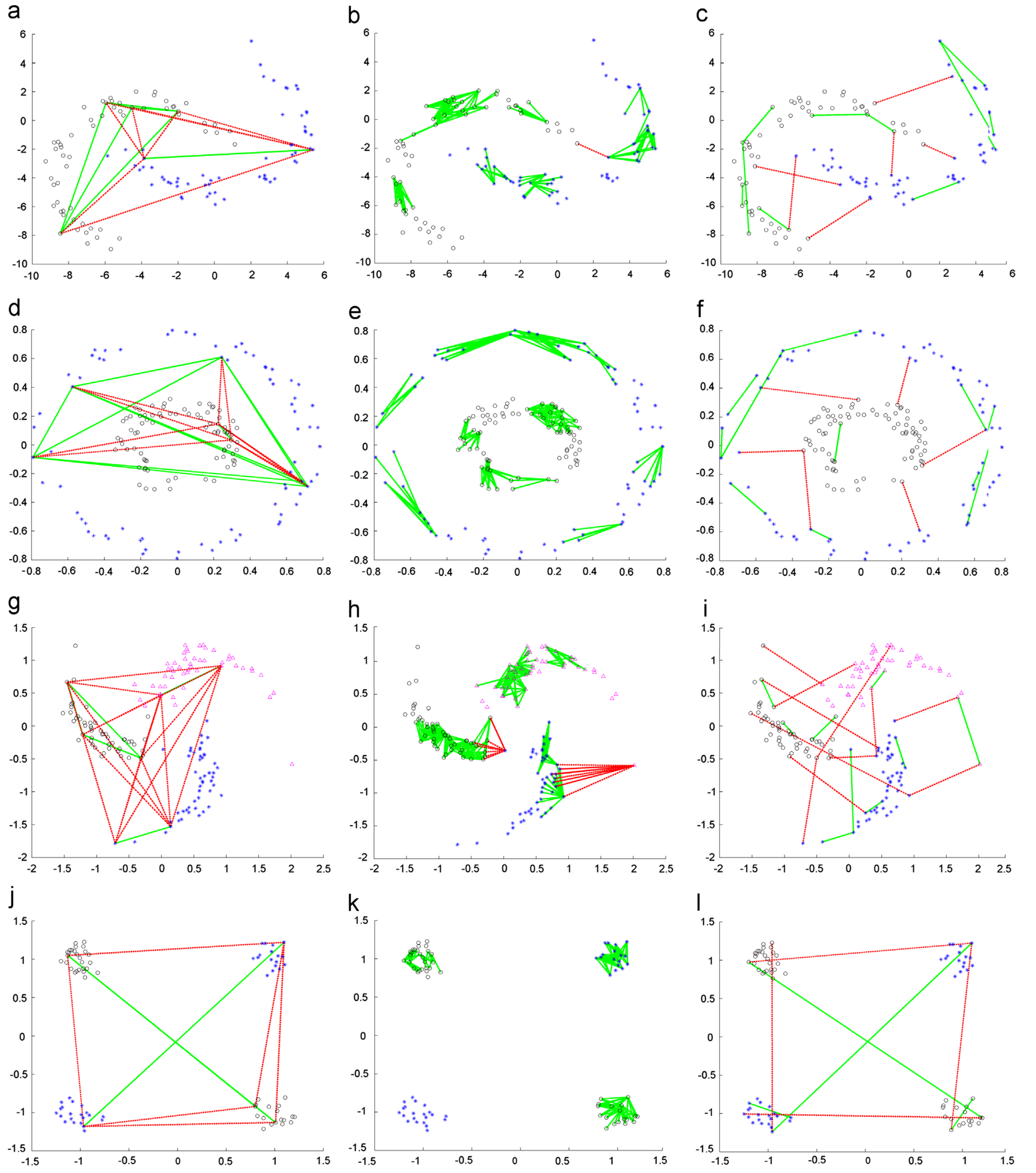

**Fig. 10.** Comparison of the behavior of three constraint selection methods FFQS, ASC and the proposed method on some synthetic datasets. Each figure is provided with the method and total utility obtained by Eq. (4) in caption. Must-link constraints are drawn with solid lines and cannot-link constraints are drawn with dashed lines in this figure. (a), (b) and (c): 60/60 Bananas dataset with $\lambda=16$; (d), (e) and (f): 85/85 two rings dataset with $\lambda=16$; (g), (h) and (i): 50/50/50 pinwheel dataset with $\lambda=20$; and (j), (k) and (l): 50/50/50 XOR dataset with $\lambda=8$. (a) *FFQS*, 1:41, (b) *ASC*, 1:36, (c) *Proposed*, 7:67, (d) *FFQS*, 2:34, (e) *ASC*, 0:89, (f) *Proposed*, 7:34, (g) *FFQS*, 1:18, (h) *ASC*, 1:35 (i) *Proposed*, 9:12, (j) *FFQS*, 1:03, (k) *ASC*, 1:14 and (l) Proposed, 1:41. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

method outperforms FFQS in Diabetes $\oplus$ MPC $-$ KMeans for $\lambda \leq 30$ constraints, FFQS surpasses the proposed method when the number of queries exceeds 30. This is because of that for simpler datasets with a small number of clusters like Diabetes, FFQS generally better defines the relevant components in RCA or the initialization of the centers in MPC-KMeans during its exploration step. In the case of more complex datasets like Glass (214 objects, 6 clusters), Protein (114 objects, 6 clusters), and Image

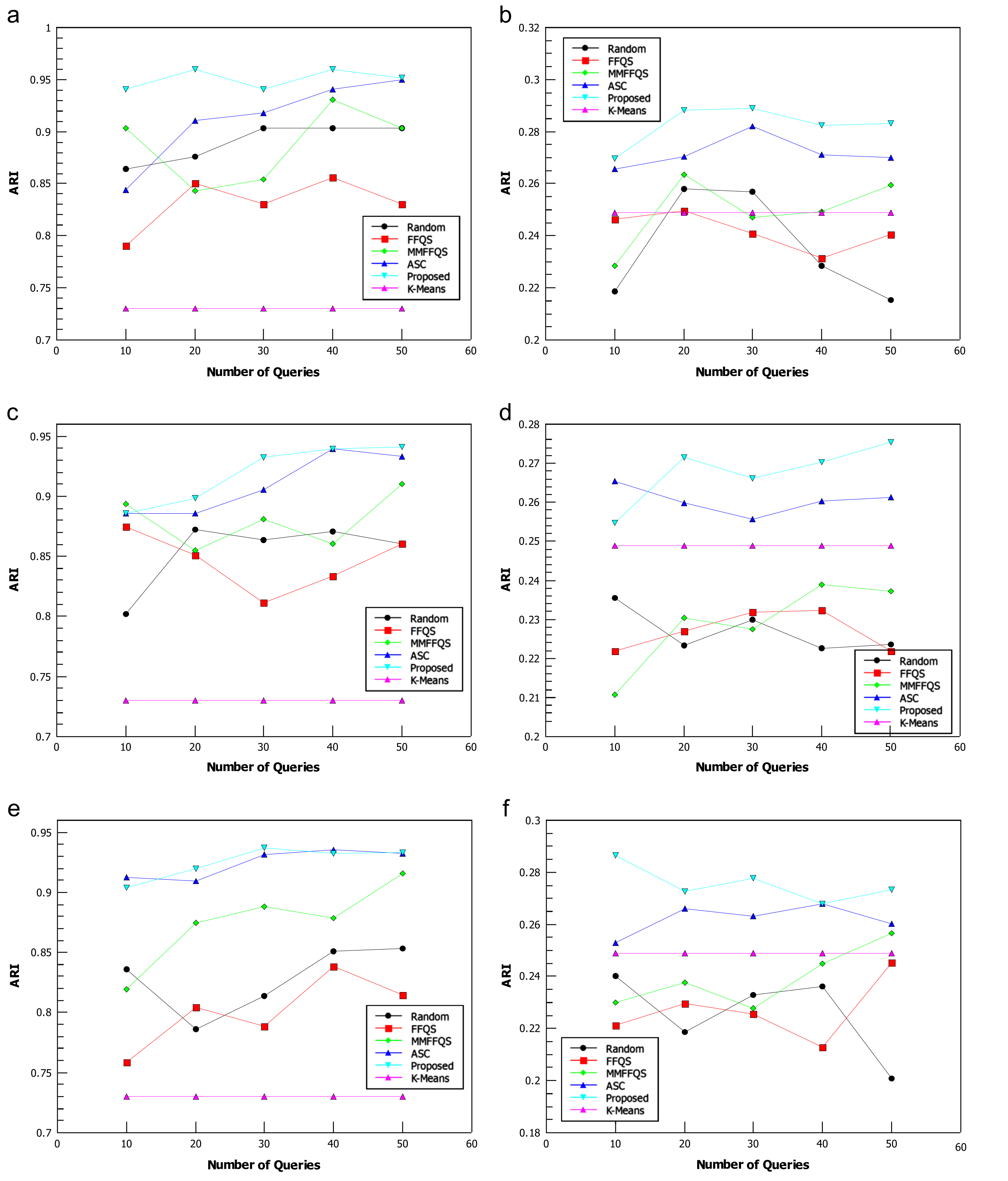

**Fig. 11.** Efficiency of different constraint selection methods in conjunction with three constraint-based clustering algorithms Xiang [9], RCA [32], and MPC-KMeans [33] on the quality of clustering. (a) Iris ⊕ Xiang, (b) Glass ⊕ Xiang, (c) Iris ⊕ RCA, (d) Glass ⊕ RCA, (e) Iris ⊕ MPC – KMeans and (f) Glass ⊕ MPC – KMeans.

segmentation (2100 objects, 7 clusters), FFQS needs more constraints than the proposed method to provide an equivalent ARI score.

As illustrated in Figs. 11–16, the proposed method makes better results than MMFFQS except in Diabetes ⊕ RCA (Fig. 16(b)) for $\lambda \leq 20$ queries. For $\lambda > 20$, a drop in efficiency puts MMFFQS at a lower accuracy than the proposed method while the proposed method keeps its efficiency in a stable level of accuracy greater than MMFFQS. This phenomenon is also observable in Breast ⊕ MPC – KMeans (Fig. 15(e)), which a drop in $\lambda = 20$ queries puts

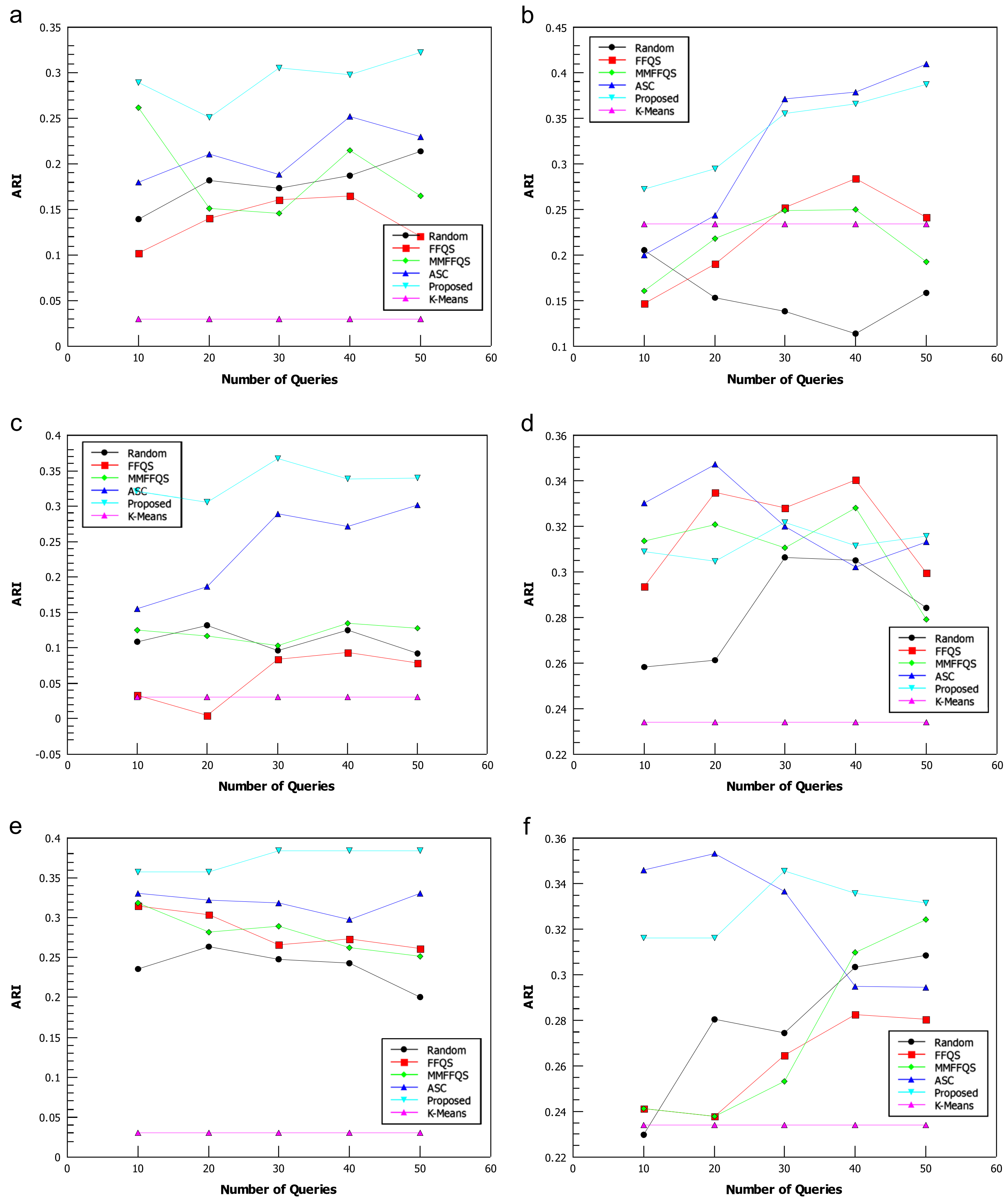

**Fig. 12.** Efficiency of different constraint selection methods in conjunction with three constraint-based clustering algorithms Xiang [9], RCA [32], and MPC-KMeans [33] on the quality of clustering. (a) Heart $\oplus$ Xiang, (b) Protein $\oplus$ Xiang, (c) Heart $\oplus$ RCA, (d) Protein $\oplus$ RCA, (e) Heart $\oplus$ MPC – KMeans and (f) Protein $\oplus$ MPC – KMeans.

MMFFQS and the proposed method in the same level of accuracy. Like FFQS, this can also be explained by the exploration step in MMFFQS causes better initialization of centers in MPC-KMeans and, consequently, a better definition of relevant components in RCA for simple datasets like Diabetes and Breast (both with two clusters of objects).

It should also be mentioned in comparison between the proposed method and Random heuristics that the proposed method provides more dominant results than Random heuristic in all experiments except in Diabetes $\oplus$ MPC – KMeans (Fig. 16(c)) for $\lambda > 30$ constraints. It can be concluded from the results that Random heuristic is not efficient enough to provide proper constraints

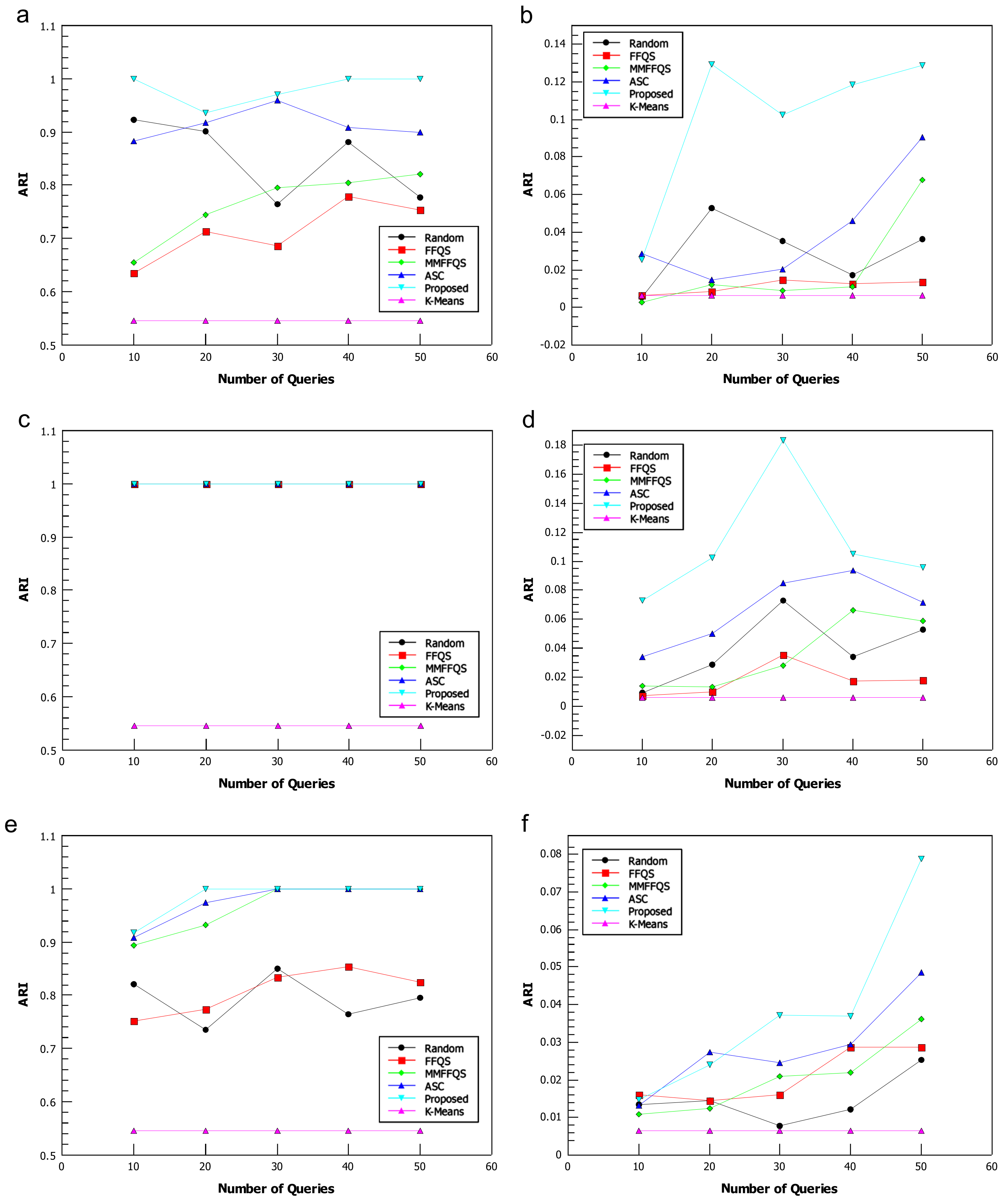

**Fig. 13.** Efficiency of different constraint selection methods in conjunction with three constraint-based clustering algorithms Xiang [9], RCA [32], and MPC-KMeans [33] on the quality of clustering. (a) Soybean $\oplus$ Xiang, (b) Sonar $\oplus$ Xiang, (c) Soybean $\oplus$ RCA, (d) Sonar $\oplus$ RCA, (e) Soybean $\oplus$ MPC − KMeans and (f) Sonar $\oplus$ MPC − KMeans.

for true metric learning (for Xiang and MPC-KMeans algorithms) and relevant components discovering (for RCA algorithm).

In comparison with ASC heuristic that generally outperforms Random, FFQS, and MMFFQS, the proposed method obtains better results except in Protein $\oplus$ RCA (Fig. 12(d)), Protein $\oplus$ MPC − KMeans (Fig. 12(f)), and Diabetes $\oplus$ RCA (Fig. 16(b)) for $\lambda \le 20$ constraints. In these cases, a drop in efficiency with $\lambda = 30$ constraints makes the proposed method superior to ASC for $\lambda > 30$. The superiority of ASC in a small number of queries in Protein $\oplus$ RCA, Protein $\oplus$ MPC − KMeans, and Diabetes $\oplus$ RCA comes from its *Propagation* procedure that discovers new constraints from the information stored in previously chosen constraints and gives ASC the capability to

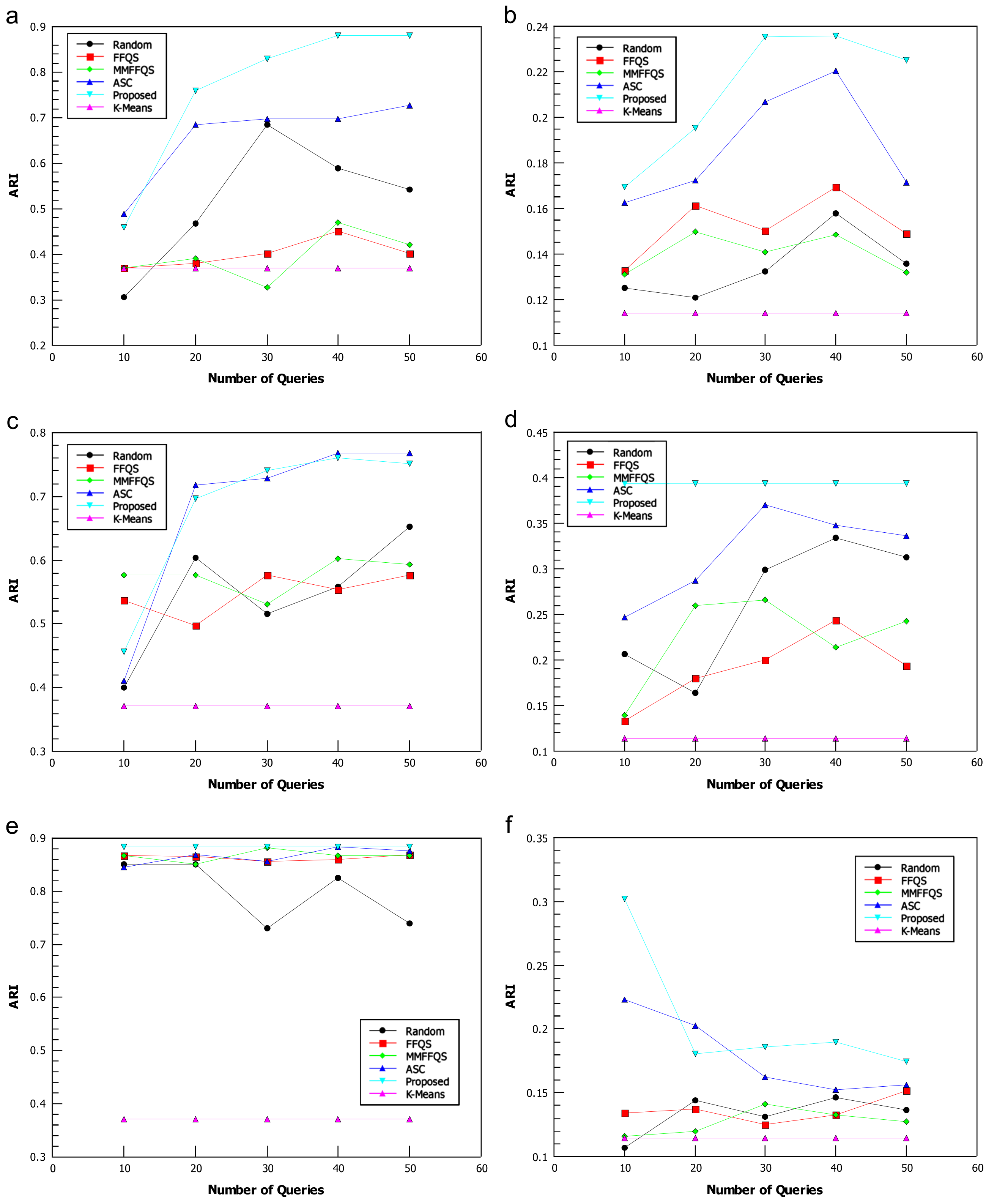

**Fig. 14.** Efficiency of different constraint selection methods in conjunction with three constraint-based clustering algorithms Xiang [9], RCA [32], and MPC-KMeans [33] on the quality of clustering. (a) Wine ⊕ Xiang, (b) Balance ⊕ Xiang, (c) Wine ⊕ RCA, (d) Balance ⊕ RCA, (e) Wine ⊕ MPC – KMeans and (f) Balance ⊕ MPC – KMeans.

select a well-propagated set of constraints in a small number of queries. On the other hand, the efficiency of ASC is highly depends on its parameters $k$ and $\theta$. Any improper assignments of $k$ and $\theta$ may result in an inefficient set of constraints. In addition, some values of $k$ and $\theta$ may result in incorrectly propagated constraints in the propagation step. It means that ASC may provide incorrect-labeled constraints and mislead the clustering algorithms e.g. Xiang and MPC-KMeans to learn accurate metrics or RCA to learn efficient relevant components.

Significant improvement in accuracy of the Xiang algorithm in conjunction with the proposed method is another noticeable point observed in these experiments. It is especially evident in

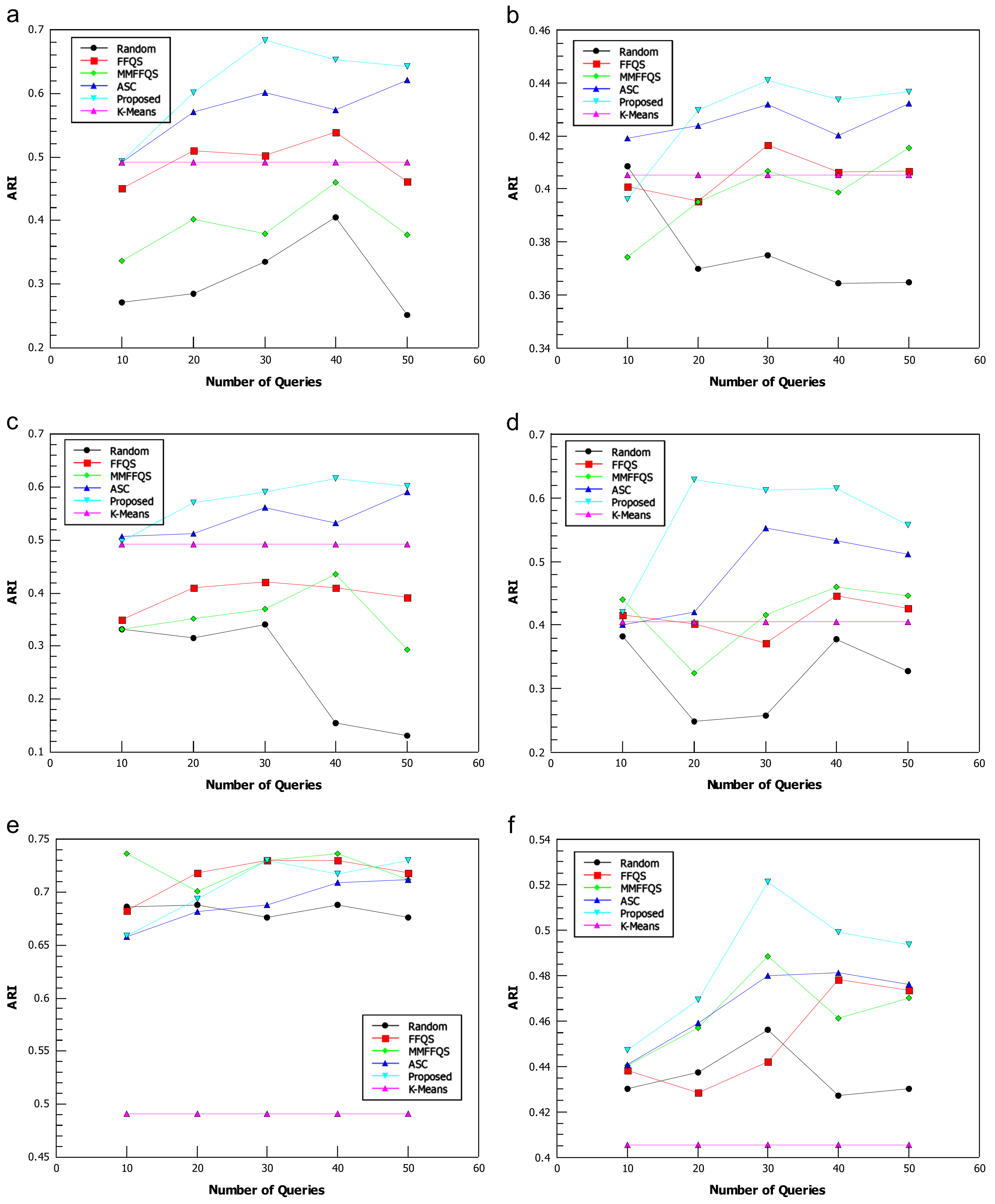

**Fig. 15.** Efficiency of different constraint selection methods in conjunction with three constraint-based clustering algorithms Xiang [9], RCA [32], and MPC-KMeans [33] on the quality of clustering. (a) Breast ⊕ Xiang, (b) Image ⊕ Xiang, (c) Breast ⊕ RCA, (d) Image ⊕ RCA, (e) Breast ⊕ MPC − KMeans and (f) Image ⊕ MPC − KMeans.

Heart ⊕ Xiang (Fig. 12(a)), Sonar ⊕ Xiang (Fig. 13(b)), Wine ⊕ Xiang (Fig. 14(a)), Balance ⊕ Xiang (Fig. 14(b)), Breast ⊕ Xiang (Fig. 15(a)), and Diabetes ⊕ Xiang (Fig. 16(a)). It comes from the appropriate distribution of selected constraints among disjointed regions, which gives the Xiang algorithm helpful information about the underlying data model and, consequently, the underlying metric.

A local decrease in accuracy with increasing the number of queries is a well-known issue in constrained clustering [18], which implies an inconsistency between the chosen queries. This issue is observable in

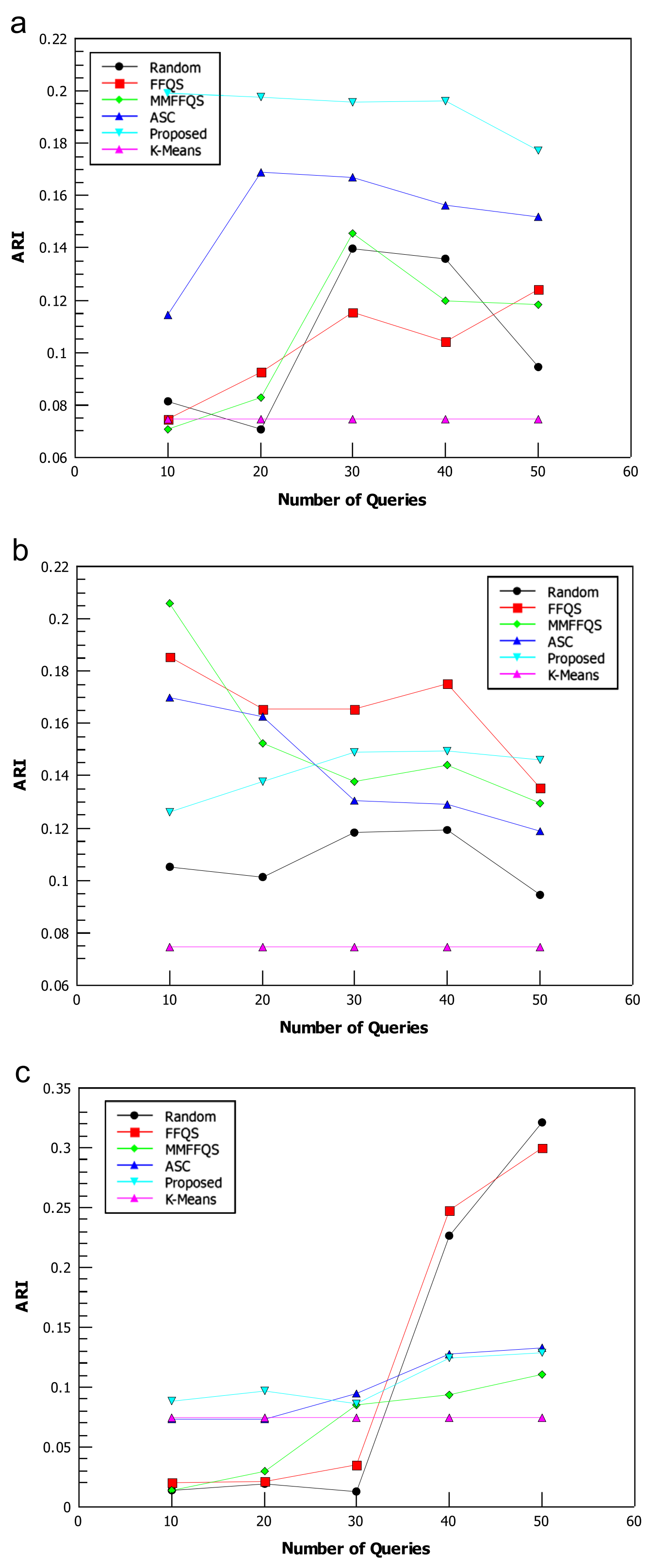

**Fig. 16.** Efficiency of different constraint selection methods in conjunction with three constraint-based clustering algorithms Xiang [9], RCA [32], and MPC-KMeans [33] on the quality of clustering. (a) Diabetes ⊕ Xiang, (b) Diabetes ⊕ RCA and (c) Diabetes ⊕ MPC – KMeans.

most of datasets for Random, FFQS and MMFFQS heuristics. Furthermore, this is less evident for ASC with significant occurrences in Protein ⊕ RCA (Fig. 12(d)), Protein ⊕ MPC – KMeans (Fig. 12(f)), Balance ⊕ Xiang (Fig. 14(b)), and Diabetes ⊕ RCA (Fig. 16(b)). On the other hand, the proposed method encountered this issue considerably in Sonar ⊕ RCA (Fig. 13(d)), and Balance ⊕ MPC – KMeans (Fig. 14(f)). In all remaining datasets ⊕ clustering algorithms, the proposed method results in a nearly smooth increase in accuracy, which implies the consistency of selected constraints. It means that the new constraints have been added with more consistency to the already selected constraints in the proposed method. As a result, it can be mentioned that the randomness in constraint selection heuristics makes this issue more likely to occur and the less a constraint selection method is supported by a random property, the more this issue is probable to occur.

It is also interesting to notice that the proposed method has kept the performance of clustering algorithms at a level of accuracy greater than the classical K-Means except for $\lambda = 10$ constraints in Image ⊕ Xiang (Fig. 15(b)). On the other hand, it is considerably evident for Random, FFQS, and MMFFQS especially in Image ⊕ Xiang (Fig. 15(b)), Image ⊕ RCA (Fig. 15(d)), Breast ⊕ Xiang (Fig. 15(a)), Breast ⊕ RCA (Fig. 15(c)), Protein ⊕ Xiang (Fig. 12(b)), Glass ⊕ Xiang (Fig. 11(b)), Glass ⊕ RCA (Fig. 11(d)), and Glass ⊕ MPC – KMeans (Fig. 11(f)). As the results show, Xiang and RCA algorithms do not efficiently result in conjunction with non-deterministic constraint selection heuristics. This can be explained by the fact that randomness in such heuristics misleads Xiang and RCA algorithms to discover the underlying actual model of the given data. Moreover, ASC outperforms K-Means in most of the experiments, which confirms efficiency of constraints selected by a reasonable assumption on the underlying model.

Applicability of the proposed method is another important issue, which should be considered. Although Random, FFQS, and MMFFQS heuristics sometimes increase the accuracy of clustering algorithms in comparison with classical K-Means, their performances are averaged over several runs because of their randomness. In many applications, it is not possible to average the performance over multiple runs because of the cost overhead and, etc. This issue reduces the applicability of such methods to real-world applications. On the other hand, the proposed and ASC methods are deterministic and this makes them more appropriate to be used for such applications. However, the proposed method can be a better option for constraint selection in real-world applications because it demonstrated more improvement and constraint consistency than the ASC method.

The time complexity of the proposed method depends on the time complexity to find $N_{sv}$ candidate points, time complexity of measuring the individual utilities for $\binom{N_{sv}}{2}$ candidate constraints, and time complexity of the sequential selection of $\lambda$ constraints. The quadratic programming problem of SVDD can be solved by the SMO algorithm [35], which was recently proposed as an efficient tool for solving such problems in SVM training. Benchmarks reported in [35] show that this algorithm converges in most cases in $O(N^2)$ kernel evaluations. The time complexity of the labeling part of SVDD is $O(N^2 d)$, where $d$ is dimension of data in input space. Hence, overall complexity of SVDD is $O(N^2 d)$. Measuring the individual utility of $\binom{N_{sv}}{2}$ candidate constraints is lead to worst case complexity $O(N^2 n_s)$, where $n_s$ denotes the number of uniformly sampled points along constraint $(i,j)$ to calculate $v_{ij}^{out}$. The worst case time complexity of the sequential selection of $\lambda$ constraints is $O(N^2 \lambda)$. Therefore, the overall time complexity of the proposed method is $O(N^2 d + N^2 n_s + N^2 \lambda)$ or $O(N^2.\max(d, n_s, \lambda))$.

## 5. Conclusion

Identifying the most beneficial set of clustering constraints was considered in this paper. The transition region and boundary information of data extracted by data description methods were utilized to introduce a time-varying utility measure for constraints in this paper. In the proposed method, the constraint selection was done sequentially so that the selection heuristic taken into account the constraint utility dependencies. It was performed by updating the utility of constraints each time a candidate constraint was selected to avoid selecting a similar constraint to an already selected one. Experiments carried out in conjunction with three different clustering algorithms on synthetic and real datasets show that the proposed method outperforms Random, FFQS, MMFFQS, and ASC heuristics in terms of clustering accuracy. The efficiency of the proposed method strongly depends on how much Assumptions A and B are held in the given data. A promising future direction is to develop a method to adaptively measure the assumption contribution factors for Assumptions A and B based on intrinsic properties of data. Parameters $C$ and $\sigma$ have also been tried to get the appropriate values during this paper. Assigning appropriate values to these parameters can increase the accuracy of clustering algorithms effectively. Given the importance of these parameters on the accuracy of clustering, adaptive initialization of these parameters, considering the special characteristics of the given data can be considered as another direction to the future work.

## Conflict of interest statement

None declared.

## Acknowledgments

The authors would like to thank reviewers for their time, comments and constructive criticism, which improved the paper.

## References

[1] A.K. Jain, Data clustering: 50 years beyond k-means, Pattern Recognition Letters 31 (8) (2010) 651–666.
[2] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl, Constrained k-means clustering with background knowledge, in: Proceedings of the 18th International Conference on Machine Learning, ICML '01, 2001, pp. 577–584.
[3] S. Basu, I. Davidson, K.L. Wagstaff, Constrained Clustering: Advances in Algorithms, Theory, and Applications, Chapman and Hall/CRC, 2008.
[4] J. Sinkkonen, S. Kaski, Clustering based on conditional distributions in an auxiliary space, Neural Computing 14 (2002) 217–239.
[5] J. Ye, Z. Zhao, H. Liu, Adaptive distance metric learning for clustering, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2007, 2007, pp. 1–7.
[6] M. Soleymani Baghshah, S. Bagheri Shouraki, Non-linear metric learning using pairwise similarity and dissimilarity constraints and the geometrical structure of data, Pattern Recognition 43 (2010) 2982–2992.
[7] P. Jain, B. Kulis, J.V. Davis, I.S. Dhillon, Metric and kernel learning using a linear transformation, Journal of Machine Learning Research 13 (2012) 519–547.
[8] T. Hertz, A. Bar-hillel, D. Weinshall, Boosting margin based distance functions for clustering, in: Proceedings of the 21th International Conference on Machine Learning, ICML '04, 2004, pp. 393–400.
[9] S. Xiang, F. Nie, C. Zhang, Learning a mahalanobis distance metric for data clustering and classification, Pattern Recognition 41 (12) (2008) 3600–3612.
[10] A.A. Abin, H. Beigy, Clustering at presence of side information via weighted constraints ratio gap maximization, in: Proceedings of the First International Workshop on Multi-View Data, High Dimensionality, and External Knowledge: Striving for a Unified Approach to Clustering, 3–12, 2012, pp. 27–38.
[11] M. Okabe, S. Yamada, Clustering with constrained similarity learning, in: Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence and International Conference on Intelligent Agent Technology, 2009, pp. 30–33.
[12] H. Chang, D. yan Yeung, Locally linear metric adaptation for semi-supervised clustering, in: Proceedings of the 21th International Conference on Machine Learning, ICML '04, 2004, pp. 153–160.
[13] M. Kalakech, P. Biela, L. Macaire, D. Hamad, Constraint scores for semi-supervised feature selection: a comparative study, Pattern Recognition Letters 32 (5) (2011) 656–665.
[14] Z. Zhang, M. Zhao, T.W.S. Chow, Marginal semi-supervised sub-manifold projections with informative constraints for dimensionality reduction and recognition, Neural Network 36 (2012) 97–111.
[15] X. Yin, S. Chen, E. Hu, D. Zhang, Semi-supervised clustering with metric learning: an adaptive kernel method, Pattern Recognition 43 (4) (2010) 1320–1333.
[16] L. Jiao, F. Shang, F. Wang, Y. Liu, Fast semi-supervised clustering with enhanced spectral embedding, Pattern Recognition 45 (12) (2012) 4358–4369.
[17] K. Wagstaff, Value, cost, and sharing: open issues in constrained clustering, in: Proceedings of the 5th International Workshop Knowledge Discovery in Inductive Databases, KDID '06, 2006, pp. 1–10.
[18] I. Davidson, K.L. Wagstaff, S. Basu, Measuring constraint-set utility for partitional clustering algorithms, in: Proceedings of the 10th European Conference on Principle and Practice of Knowledge Discovery in Databases, PKDD '06, 2006, pp. 115–126.
[19] S. Basu, A. Banerjee, R.J. Mooney, Active semi-supervision for pairwise constrained clustering, in: Proceedings of the 5th SIAM International Conference on Data Mining, ICDM '04, 2004, pp. 333–344.
[20] D. Klein, S.D. Kamvar, C.D. Manning, From instance-level constraints to space-level constraints: making the most of prior knowledge in data clustering, in: Proceedings of the 19th International Conference on Machine Learning, ICML '02, 2002, pp. 307–314.
[21] Q. Xu, M. desJardins, K.L. Wagstaff, Active constrained clustering by examining spectral eigenvectors, in: Proceedings of the 8th International Conference on Discovery Science, DS '05, 2005, pp. 294–307.
[22] N. Grira, M. Crucianu, N. Boujemaa, Active semi-supervised fuzzy clustering, Pattern Recognition 41 (5) (2008) 1834–1844.
[23] P.K. Mallapragada, R. Jin, A.K. Jain, Active query selection for semi-supervised clustering, in: Proceedings of the 19th International Conference on Pattern Recognition, ICPR '08, 2008, pp. 1–4.
[24] V.-V. Vu, N. Labroche, B. Bouchon-Meunier, Improving constrained clustering with active query selection, Pattern Recognition 45 (4) (2012) 1749–1758.
[25] B. Settles, Active Learning, Morgan & Claypool, 2012.
[26] V.-V. Vu, N. Labroche, B. Bouchon-Meunier, An efficient active constraint selection algorithm for clustering, in: Proceedings of the 20th International Conference on Pattern Recognition, ICPR '10, 2010, pp. 2969–2972.
[27] L. Parra, G. Deco, S. Miesbach, Statistical independence and novelty detection with information preserving nonlinear maps, Neural Computing 8 (2) (1996) 260–269.
[28] M.-F. Jiang, S.-S. Tseng, C.-M. Su, Two-phase clustering process for outliers detection, Pattern Recognition Letters 22 (6/7) (2001) 691–700.
[29] C. Campbell, K.P. Bennett, A linear programming approach to novelty detection, in: Advances in Neural Information Processing Systems 13, NIPS 2000, 2000, pp. 395–401.
[30] D.M.J. Tax, R.P.W. Duin, Support vector domain description, Pattern Recognition Letters 20 (1999) 1191–1199.
[31] P. Juszczak, D.M.J. Tax, E. Pekalska, R.P.W. Duin, Minimum spanning tree based one-class classifier, Neurocomputing 72 (2009) 1859–1869.
[32] A. Bar-Hillel, T. Hertz, N. Shental, D. Weinshall, Learning a mahalanobis metric from equivalence constraints, Journal of Machine Learning Research 6 (2005) 937–965.
[33] M. Bilenko, S. Basu, R.J. Mooney, Integrating constraints and metric learning in semi-supervised clustering, in: Proceedings of the 21th International Conference on Machine learning, ICML '04, 2004, pp. 11–18.
[34] L. Hubert, P. Arabie, Comparing partitions, Journal of Classification 2 (1) (1985) 193–218.
[35] J.C. Platt, Fast training of support vector machines using sequential minimal optimization, in: Advances in Kernel Methods—Support Vector Learning, 1999, pp. 185–208.

**Ahmad Ali Abin** received the B.S. in computer engineering from Iran University of Science & Technology, Iran, in 2005. In September 2008, he completed the M.S. degree in Computer Engineering at Sharif University of Technology, Iran. Currently, he is a Ph.D. candidate at the Department of Computer Engineering, Sharif University of Technology. His research interests focus on pattern recognition, machine learning and neural computing.

**Hamid Beigy** received the B.S. and M.S. degrees in Computer Engineering from the Shiraz University in Iran, in 1992 and 1995, respectively. He also received the Ph.D. degree in Computer Engineering from the Amirkabir University of Technology, Iran, in 2004. Currently, he is an Associate Professor in Department of Computer Engineering at the Sharif University of Technology, Tehran, Iran. His research interests include learning systems, high performance computing and soft computing.