Adv Syst Sci Appl 2019; **04**; 25-44
Published online at http://ijassa.ipu.ru/index.php/ijassa/article/view/659

Improving Semi-Supervised Clustering Algorithms with Active Query Selection

Walid Atwa[*1], Mahmoud Emam[2]

1) Computer science department, Faculty of Computers and Information, Menoufia University, 32511, Egypt

E-mail: walid.atwa@ci.menofia.edu.eg

2) Mathematics and computer science department, Faculty of Science, Menoufia University, 32511, Egypt

E-mail: ma7moud_emam@yahoo.com

Received October 30, 2018; Revised April 3, 2019; Published December 31, 2019

Abstract: Semi-supervised clustering algorithms use a small amount of supervised data in the form of pairwise constraints to improve the clustering performance. However, most current algorithms are passive in the sense that the pairwise constraints are provided beforehand and selected randomly. This may lead to the use of constraints that are redundant, unnecessary, or even harmful to the clustering results. In this paper, addressing the problem of constraint selection to improve the performance of semi-supervised clustering algorithms. Based on the concepts of Maximum Mean Discrepancy, proposed method selects a batch of most informative instances that minimize the difference in distribution between the labeled and unlabeled data. Then, querying these instances with the existing neighborhoods to determine which neighborhood they belong. The experimental results with state-of-the-art methods on different real-world dataset demonstrate the effectiveness and efficiency of the proposed method.

Keywords: Semi-supervised clustering, Active Learning, Pairwise constraints.

1. INTRODUCTION

Recently, Semi-supervised clustering (also known as constraint-based clustering) algorithms have become a topic of significant interest for many researchers. These algorithms aim to improve the clustering performance with the help of user-provided side information. There are several types of side information but pairwise constraints are the most used one. There are two types of pairwise constraints: Must-Link (*ML*) and Cannot-Link (*CL*) constraints. The must-link constraint indicates that the two objects must be grouped into the same cluster while the cannot-link constraint indicates that the two objects must be in different clusters.

The existing constraint-based clustering algorithms assumed that they can improve the clustering performance with a suitable passively chosen set of constraints [1, 2]. However, if the constraints are selected improperly, they may also degrade the clustering performance [3, 4]. Moreover, selecting constraints typically requires a user to manually inspect the data instances that can be time consuming and costly. For those reasons, the proposed method optimize the selection of the constraints to improve the performance of semi-supervised clustering algorithms.

[*] Corresponding author: walid.atwa@ci.menofia.edu.eg

Active learning is well motivated in many supervised learning scenarios where unlabeled instances are abundant and easy to retrieve but labeled instances are difficult, time-consuming and expensive to obtain. For example, it is easy to gather large amounts of unlabeled documents or images from the internet, whereas querying them requires manual effort from experienced human annotators. Hence there is a need to select an optimal set of instances from the pool of unlabeled data for querying. Randomly selection of unlabeled instances for querying is inefficient in many situations, since non-informative or redundant instances might be selected. Active learning algorithms select the most informative unlabeled instances from enormous amount of unlabeled data for querying. Specifically, the goal of active learning is to query data as little as possible to achieve a certain performance, thus saving considerable cost for generating good queries.

In this paper, the active learning is applied in an iterative manner. In each iteration, set of queries are selected and queried with the existing neighborhoods to improve the clustering results. Specifically, selecting a batch of informative query instances such that the distribution represented by the selected query set and the available labeled data is closest to the distribution represented by the unlabeled data. In other words, the proposed method select a set of samples S from the unlabeled data, denoted by D_U, such that the probability distributions represented by $D_L \cup S$ and $D_U \setminus S$, where D_L is the set of available labeled data, are similar to each other. Then measuring the difference in the probability distribution between the two sets of data using the Maximum Mean Discrepancy (MMD) [5, 6]. Maximum Mean Discrepancy is a statistical test based on the fact that two distributions are different if and only if there exists at least one function in reproducing kernel Hilbert space (RKHS) [6] having different expectations on the two distributions.

Once the batch of informative query instances are selected, the proposed method query them with the existing neighborhoods to determine which neighborhood they belong. A neighborhood contains a set of data instances that are known to belong to the same cluster (i.e., connected by must-link constraints) and different neighborhoods are known to belong to different clusters (i.e., connected by cannot-link constraints). Well-formed neighborhoods can provide valuable information regarding what the underlying clusters look like.

We empirically evaluate the proposed method with baseline and state-of-the-art methods on UCI real datasets. The evaluation results demonstrate that the proposed method achieves consistent improvements over the baseline methods.

The remainder of the paper is organized as follows. Section 2 presents a brief review of the related work. Section 3 introduces the proposed method. Experimental results are presented in Section 4. Finally, conclude the paper and discuss future directions in Section 5.

2. RELATED WORK

Active learning has a long history in supervised learning algorithms [7, 8, 9, 10, 11]. Recently, few studies reported the result of using active learning in constrained-based clustering problems.
The first study was conducted by Basu et al. [1] that proposed an active *k*-means clustering using the farthest-first strategy that has two-phases (*Explore* and *Consolidate*). The first phase (*Explore*) uses the farthest-first scheme to form appropriate queries for getting the required pairwise disjoint neighborhoods. At the end of Explore, at least one point has been obtained per cluster. The second phase (*Consolidate*) iteratively expands the neighborhoods. Where in each iteration it selects a random point outside any neighborhood and queries it against the existing neighborhoods until a must-link is found. An improvement version called Min-Max approach [12], which modifies the Consolidate phase by selecting the most uncertain point to query, instead of selecting the point

randomly. The idea is to select the data point whose largest similarity to the skeleton is the smallest. By this way, data points with largest uncertainty in cluster membership are chosen first to express the user queries. However, both previous methods do not work well in the case of a dataset with a large number of clusters or unbalanced datasets with small clusters.

Xu et al. [13] proposed an active constrained spectral clustering algorithm that examines the eigenvectors to identify the boundary points (of two classes) and sparse points; then it queries the oracle for constraints based on these points. It has shown limited applicability because it requires many queries to the oracle and assumes that errors in the clustering result only occur on the boundary points. Wang et al. [14] presented another spectral active clustering technique that identifies informative pairs according to the entropy of the pair example. But, these approaches have limited their work to two classes, and the direct generalizations to multi-classes cases are not known.

Vu et al. [15, 16] proposed an active query selection method based on a constraint utility function called Ability to Separate between Clusters. This method relies on two aspects: (1) a *k*-nearest neighbors graph is used to determine the best candidate queries in the sparse regions of the dataset between the clusters, and (2) a propagation procedure allows each user query to generate several constraints which limits the user intervention. 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.

Recently, Xiong et al. [17] proposed an active learning method based on the classic uncertainty-based principle. They studied the selection of constraints by selecting the most informative instance to form queries accordingly. The responses to the queries are then used to improve the clustering results. However, this method selects only single instance that can become very slow for retraining with each single instance being queried. Furthermore, if a parallel querying system is available, e.g., multiple annotators working in parallel, these methods would not be able to make the effective use of the resources.

Li et al. [18] proposed an active learning method that makes embeddings of labeled examples to those of unlabeled ones and back via deep neural networks. The active scheme makes association cycles that end up at the same class from that the association was started, which considers both the informativeness and representativeness of examples.

The above mentioned methods form arrange of studies performed in active selection of 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. ACTIVE CONSTRAINT SELECTION METHOD

Semi-supervised clustering algorithms attempt to partition the unlabeled data into a set of clusters with the help of a small amount of pairwise constraints (must-link and cannot-link). In this section, addressing the problem of how to effectively choose pairwise constraints to produce accurate clustering results. The proposed method first selects a batch of most informative instances that minimize the difference in distribution between the labeled and unlabeled data. Then, querying these instances with the existing neighborhoods to determine which neighborhood they belong.

3.1 Batch Instances Selection

Traditional data mining and machine learning algorithms are based on the assumption that the training data (X, Y) represents the true underlying distributions of X and Y where $X=\{x_1, x_2,\ldots, x_n\}$ is the training data and their corresponding labels $Y=\{y_1, y_2,\ldots, y_n\}$. Hence a model learned on this data works well for the test data (X_{test}, Y_{test}) which is also drawn independently and identically distributed from the same distribution [5]. Thus, a batch of query instances can be selected from unlabeled data such that the distribution represented by the queried and labeled data is similar to the probability distribution of the unlabeled data set. In other words, selecting a batch of instances S from the unlabeled data (denoted by D_U) such that the joint probability distribution represented by $D_L \cup S$ and $D_U \setminus S$ are similar to each other, where D_L is set of available labeled data. This function is summarized in Algorithm 1.

To measure the difference between two distributions, Maximum Mean Discrepancy (MMD) has been shown to be an effective measure of the difference in their marginal probability distributions [5, 6]. The principal underlying the Maximum Mean Discrepancy is to find a function that assumes different expectations on two different distributions so that when evaluated empirically on samples drawn from the different distributions it would tell us whether the distributions are similar or not. Let $\mathcal{F}$ be a class of functions$f: \mathcal{X} \to \mathbb{R}$.. Let p and q be probability distributions defined on a domain $\mathcal{X}$, and let $X = (x_1, \ldots, x_m)$ and $Z = (z_1, \ldots, z_n)$ be samples composed of independent and identically distributed observations drawn from p and q, respectively. The maximum mean discrepancy (MMD) [5, 6] and its empirical estimate are defined as:

$$\mathrm{MMD}[\mathcal{F}, p, q] \coloneqq \sup_{f\in\mathcal{F}}\left(E_p[f(x)] - E_q[f(z)]\right) \tag{1}$$

$$\mathrm{MMD}[\mathcal{F}, X, Z] \coloneqq \sup_{f\in\mathcal{F}}\left(\frac{1}{m}\sum_{i=1}^{m} f\,(x_i) - \frac{1}{n}\sum_{i=1}^{n} f\,(z_i)\right) \tag{2}$$

There is a class of functions for which MMD may easily be computed, while retaining the ability to detect all discrepancies between p and q without making any simplifying assumptions. Let $\mathcal{H}$ be a complete inner product space (i.e., a Hilbert space) of functions$f: \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ is a nonempty compact set. Then $\mathcal{H}$ is termed a reproducing kernel Hilbert space if for all $x \in \mathcal{X}$, the linear point evaluation functional mapping $f \to f(x)$ exists and continuous. In this case, $f(x)$ can be expressed as an inner product via

$$f(x) = \langle f\phi(x)\rangle_{\mathcal{H}} \tag{3}$$

where $\phi : \mathcal{X} \to \mathcal{H}$ is known as the feature space map from $\mathcal{X}$ to $\mathcal{H}$.

When $\mathcal{F}$ is the unit ball in a characteristic RKHS [6], MMD is defined as the difference between the means of two distributions after mapping onto the characteristic RKHS. An empirical estimate of MMD is then obtained as follows:

$$\mathrm{MMD}[\phi, X, Z] \coloneqq \left\|\frac{1}{m}\sum_{i=1}^{m}\Phi(x_i) - \frac{1}{n}\sum_{i=1}^{n}\Phi(z_i)\right\|_{\mathcal{H}}^2 \tag{4}$$

Now, let us assume that we have u instances of unlabeled data D_U and l instances of labeled data D_L and we would like to select a batch S of b instances such that the distribution of $D_L \cup S$ is similar to the distribution of $D_U \setminus S$. Thus, the MMD between the sets $D_L \cup S$ and $D_U \setminus S$ is defined by $f(S)$, can be computed using the expression in Equation (4), as follows:

$$f(S) = \left\|\frac{1}{l+b}\sum_{j\in D_L\cup S}\Phi\left(x_j\right) - \frac{1}{u-b}\sum_{i\in D_U\setminus S}\Phi(x_i)\right\|_{\mathcal{H}}^2 \tag{5}$$

Since we want to select a set S from unlabeled data set D_U to minimize the mismatch between $D_L \cup S$ and $D_U \setminus S$. Defining a binary vector α of size u where each entry α_i indicates whether the data $x_i \in D_U$ is selected or not. If a point is selected, the corresponding entry α_i is 1 else 0. Thus the minimization problem reduces to finding α that minimizes the cost function $f(S)$:

$$\min_{\alpha:\alpha_i\in\{0,1\},\alpha^T\mathbf{1}=b} \left\| \frac{1}{l+b}\left(\sum_{j\in D_L}\Phi(x_j)+\sum_{i\in D_U}\alpha_i\Phi(x_i)\right)-\frac{1}{u-b}\sum_{i\in D_U}(1-\alpha_i)\Phi(x_i)\right\|_{\mathcal{H}}^2 \qquad (6)$$

where $\mathbf{1}$ is a vector of the same dimension as α with all entries 1 and symbol T is used to represent the matrix or vector transpose operation. Evidently, the cost function in Equation (6) is an alternative (equivalent) representation of the cost function $f(S)$ in Equation (5). The first term denotes the mean of the mapped features of the labeled and selected points. Note that if a point x_i is not selected in the current set then α_i will be 0 and this term would not get added in the summation. The second term is mean of the mapped features of the unlabeled data set minus the selected query set. The first constraint ensures that each entry in α is either 0 or 1 and the second constraint ensures that exactly b entries of α are 1, meaning exactly b instances are selected from the unlabeled data set, where b is specified a priori by the user.

Algorithm 1. QueryInstancesSelection(D_L, D_U, b);

Input: A set of labeled instances D_L; set of unlabeled instances D_U; batch size b.

Output: A batch of query instances S.

Compute α that minimize the distribution between the sets $D_L \cup S$ and $D_U \setminus S$ (Eq. (6))

Sort D_U in descending order of α

Select top b instances of D_U as S

Update D_L and D_U: $D_L \leftarrow D_L \cup S$, $D_U \leftarrow D_U \setminus S$

return S

3.2 A Neighborhood-based Active Learning Method

Once the batch of most informative query instances are selected, the proposed method query them against the existing neighborhoods to determine which neighborhood they belong to. In this section, introducing the concept of neighborhood, which is instrumental in the design of many existing methods for active learning of pairwise constraints [1, 12, 17]. Then, explain how the proposed active learning method can expand the neighborhoods using the selected query instances. A neighborhood contains a set of data instances that are known to belong to the same cluster (i.e. connected by must-link constraints). Different neighborhoods are connected by cannot-link constraints and thus are known to belong to different clusters. For example, Figure 1 shows a set of must-link constraints (x_1; x_2), (x_1; x_3), (x_4; x_5), and (x_4; x_6), as well as a set of cannot-link constraints (x_1; x_4) and (x_1; x_5). Two neighborhoods N_1 and N_2 can be generated as shown in Figure 1. Neighborhood N_1 includes three instances x_1, x_2, and x_3 as described by must-link constraints (x_1; x_2) and (x_1; x_3). Similarly, x_4, x_5, and x_6 should also be included in the same neighborhood. Indicated by cannot-link constraints (x_1; x_4) and (x_1; x_5), x_1 should not be in the neighborhood to which x_4 and x_5 belong. Therefore, neighborhood N_2 is discovered and it contains instances x_4, x_5, and x_6.

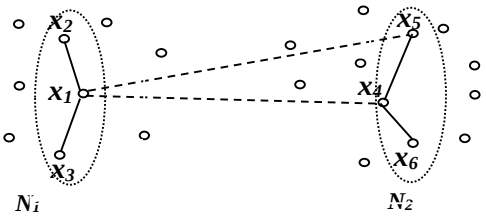


Fig. 1: An example of neighborhoods

A key advantage of using the neighborhood concepts is that by leveraging the knowledge of the neighborhoods, we can acquire a large number of pairwise constraints via a small number of queries. In particular, if we can identify the neighborhood of an instance x, we can infer its pairwise relationship with all other points that are currently confirmed to belong to any of the existing neighborhoods. This naturally encourages us to consider an active learning strategy that incrementally expands the neighborhoods by selecting the most informative instances and querying them against the known neighborhoods. We summarize the active learning method in Algorithm 2.

We begin by initializing the neighborhoods by selecting a random point to be the initial neighborhood (line 1). A selection criterion is then applied to select the batch query instances S as explained in the previous section (line 2). Each selected instance $s \in S$ is then queried against each existing neighborhood N_i to identify where s belongs, during which the constraint set C is updated (lines 5-13). To determine the neighborhood of s with the smallest number of queries, we go through the neighborhoods in decreasing order based on $p(s \in N_i)$, $i \in \{1, \cdots, k\}$, i.e. the probability of selected instance s belonging to the neighborhood N_i, which is assumed to be the average similarity between s and the instances in N_i .

$$p(s \in N_i) = \frac{\frac{1}{|N_i|} \sum_{x_j \in N_i} M(s,x_j)}{\sum_{p=1}^{k} \frac{1}{|N_p|} \sum_{x_j \in N_p} M(s,x_j)} \tag{7}$$

where $M(s, x_j)$ denote the similarity between instance s and instance x_j, $|N_i|$ indicates the number of instances in neighborhood N_i, and k is the total number of existing neighborhoods.

We should always start by querying s against the neighborhood that has the highest probability of containing s to minimize the total number of required queries. If a must-link is returned, we can stop with only one query. Otherwise, one should ask the next query against the neighborhood that has the next highest probability of containing s. This process is repeated until a must-link constraint is returned or we have a cannot-link constraint against all neighborhoods. If no must-link is achieved, a new neighborhood will be created using the instance s (lines 14-16). Finally, we apply the semi-supervised clustering algorithm using the selected active pairwise constraints to generate the final clusters (line 18). In this paper, we consider the semi-supervised clustering algorithm as a black-box and any existing algorithm can be used here.

Algorithm 2. The Proposed Active Learning Method

Input: A set of instances D (divided into labeled set D_L and unlabeled set D_U); total number of queries Q; batch size b.
Output: A set of clusters.

```
Initialization: set N_1 = {x}, where x is a random instance; C = ∅; q = 0;
while q<Q
  S = QueryInstancesSelection(D_L, D_U, b);
  for each instance s ∈ S
    for each neighborhood N_i ∈ N in decreasing order of p(s∈N_i)
      Query instance s against any instance x_i ∈ N_i ;
      q++;
      update the constraint set C based on the results;
      if a must-link achieved between s and x_i then
        add instance s to neighborhood N_i;
        break;
      end if
    end for
    if no must-link is achieved then
     create new neighborhood with the instance s;
    end if
  end for
  apply semi-supervised clustering(D, C);
end while
```

4. EXPERIMENTS

In this section, we evaluated the accuracy and efficiency of the proposed method on a diffrent real-world datasets. The results compared the proposed method with other constraint selection heuristics and evaluated in conjunction with two different constraint-based clustering algorithms to show the adaptability of the proposed method. The rest of this section is organized as follows. Section 4.1 mentions the datasets used for evaluating the proposed method. Section 4.2 describes some constraint selection heuristics that are compared with the proposed method and Section 4.3 describes different constraint-based clustering algorithms used for constraints evaluation. Section 4.4 explains the evaluation metrics used in this paper. The experimental results are described in Section 4.5.

4.1 Datasets

Experiments are conducted on 8 datasets from UCI Machine Learning Repository[†] (each with the following number of instances, attributes and clusters): Protein (116/20/6) [15], Heart (270/13/2) [17], Ionosphere (351/34/2) [13, 14], Breast (683/9/2) [15, 17], Yeast (1484/8/10) [15], Image Segmentation (2310/19/7) [15, 17], Digit-389 (3165/16/3) [17] and Magic (19020/10/2) [10]. These datasets have been chosen because they facilitate the reproducibility of the experiments and because some of them have already been used in constraint-based clustering articles. Also, provide

[†] http://www.ics.uci.edu/~mlearn/MLRepository.html

a good representation of different characteristics: number of instances ranges from 116 to 19020, dimensionalities from 8 to 34, and number of clusters from 2 to 10.

4.2 Constraint Selection Heuristics

In all experiments, we consider the following heuristics to select the constraints:

- **Random**: this policy corresponds to a completely random selection of the constraints. This method generates a set of *ML* and *CL* constraints based on the comparison of the labels of randomly chosen objects. If both labels are in the same cluster, a *ML* constraint is generated, and else, a *CL* constraint is generated.
- **Min-Max**: this approach is neighborhood-based approach that works in two phases [12]. In the first phase, it builds *c* disjoint neighborhoods using farthest-first traversal, where *c* is the total number of clusters. In the second phase, it incrementally expands the neighborhoods by selecting a point to query using a distance-based Min-Max criterion.
- **ASC:** an active learning algorithm that relies on a *k*-nearest neighbors graph and a new constraint utility function to generate queries to the human expert. ASC is based on two parameters (i.e. the number of nearest neighbors *k* and the threshold θ). These parameters *k* and θ are set to 6 and $\lfloor (k/2) + 1 \rfloor$ respectively as recommended in their method [15].
- **NPU**: an active learning method based on the classic uncertainty-based principle that takes a neighborhood based approach, and incrementally expands the neighborhoods by selecting a single instance to query each time [17].

4.3 Constraint-based Clustering Algorithms

In all experiments, we report the obtained results in conjunction with two different constraint-based clustering algorithms: the constrained K-Means (MPCKMeans) [19] and the Agglomerative Hierarchical Clustering with Constraints (AHCC) [20]. The choices of these algorithms are not critical and the proposed method can be used with any constraint-based clustering algorithm. They have been chosen because they are representative of the most popular clustering algorithms.

When evaluating the performance of particular methods on a given dataset *D*, we apply it to select up to 150 pairwise queries, starting from no query at all. The queries are answered based on the ground-truth class label for the dataset. The constraint-based clustering algorithms (MPCKMeans and AHCC) are then applied to the data with the selecting constraints.

4.4 Evaluation MetricsTo evaluate the performance of the methods, we used Normalized Mutual Information (NMI) and Pairwise F-measure as the clustering validation metrics. NMI is an external validation metric, which is used to estimate the quality of clustering with respect to the given true labels of the datasets. NMI measures how closely the clustering algorithm could reconstruct the underlying label distribution in the data. If *X* is the random variable representing the cluster assignments of the instances and *Y* is the random variable representing the class labels of the instances, then NMI is defined as follows:

$$NMI = \frac{I(X;Y)}{\big(H(X)+H(Y)\big)/2} \tag{8}$$

where $I(X; Y) = H(Y) - H(Y|X)$ is the mutual information between the random variables *X* and *Y*, H(*Y*) is the Shannon entropy of *Y* , and H(*Y*|*X*) is the conditional entropy of *Y* given *X* [21]. The range of NMI values is 0–1. In general, the larger the NMI value, the better the clustering quality.

Pairwise F-measure is another evaluation metric to evaluate how well we can predict the pairwise relationship between each pair of instances in comparison to the relationship defined by the ground truth class labels. Pairwise F-measure is defined as the harmonic mean of pairwise precision and recall, where the traditional information retrieval measures are adapted for evaluating clustering by considering pairs of points. For any pair of points, the decision to cluster this pair into same or different clusters is considered to be correct if it matches with the underlying class labeling available for the points. Pairwise F-measure is defined as follows:

$$Precision = \frac{n_c}{n_s}$$
$$Recall = \frac{n_c}{n_f}$$
$$F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (9)$$

where n_c is the number of point pairs that are correctly predicted as in the same cluster; n_s is the number of point pairs that are predicted as in the same cluster; n_f is the number of point pairs that are actually in the same cluster.

4.5 Evaluation Results

4.5.1 Effectiveness and Performance Analysis

In this section we present the evaluation results of the proposed method compared to Min–Max [12], ASC [15], NPU [17] heuristics and the Random selection of the constraints. To evaluate the performance of the proposed method to adapt to distinct clustering algorithms, the heuristics are compared in conjunction with the clustering algorithms (MPCKMeans and AHCC). Figures 2 and 3 show the clustering performance on MPCKMeans and AHCC respectively. To evaluate the performance in both MPCKmeans and AHCC, we repeat this process for 50 independent runs and report the average performance using evaluation criteria described above.

It can be observed from figures 2 and 3, that the proposed method generally outperforms other constraint selection heuristics in conjunction with the clustering algorithms MPCKMeans and AHCC. 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 the clustering performance while the other constraint selection heuristics drop in performance when the number of queries increases.

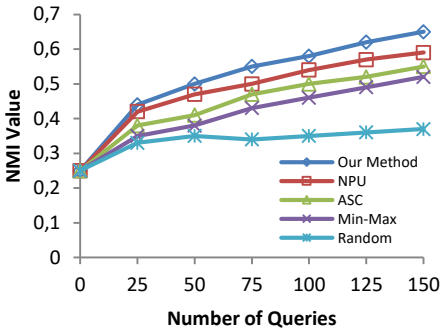
It is interesting to note that the Random selection of constraints degrade the clustering performance in some datasets when the number of queries increases (e.g. Heart, Breast, Segment, and Digit-389), while it is expected that the performance increases monotonically with the number of queries. This problem is a well-known issue in constraint-based clustering that has been addressed in [22, 23] and may be due to either the variability of the random approaches in some cases or due to a bad selection of some constraints that leads constraint-based clustering algorithm to poorer clustering results. This further demonstrates the importance of selecting the right set of constraints.

In comparison, Min-Max method obtains better results with MPCKMeans only for the Heart dataset. This can be explained by the fact that for simpler datasets with a small number of clusters like Heart dataset, the Min-Max method generally better manages to define the skeleton of *CL* constraints during its exploration step, which in turn promotes the initialization of the centers in

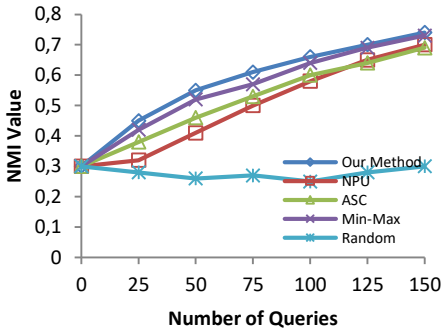
MPCKMeans. In the case of more complex datasets like Segment and Magic datasets, the Min-Max method needs more constraints to improve the clustering performance.

More generally, it can be seen from Figures 2 and 3, that the ASC method obtains better results than the proposed active learning method with small number of queries (e.g. Yeast, Segment and Digit-389 datasets). However, drop inefficiency with increasing the number of queries makes the proposed method superior to ASC. The superiority of ASC in a small number of queries comes from its propagation procedure that discovers new constraints from the information stored in previously chosen constraints and gives ASC the capability to select a well propagated set of constraints in a small number of queries. On the other hand, the efficiency of ASC is highly depending on its parameters k and θ. Any improper assignments of k and θ may result in an inefficient set of constraints. In addition, some values of k and θ may result in incorrectly propagated constraints in the propagation step. It means that ASC may provide incorrect-labeled constraints and mislead the clustering algorithms.

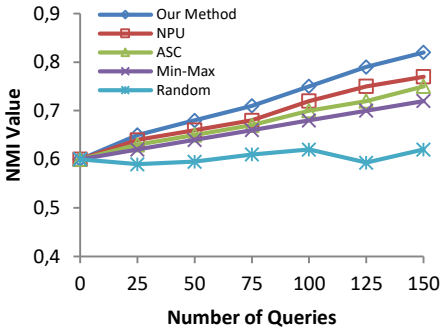
In comparison with NPU method that generally outperforms Random, Min-Max, and ASC method. NPU is generally able to improve the clustering performance consistently as increasing the number of queries. However, its performance is dominated by the proposed method in most cases.



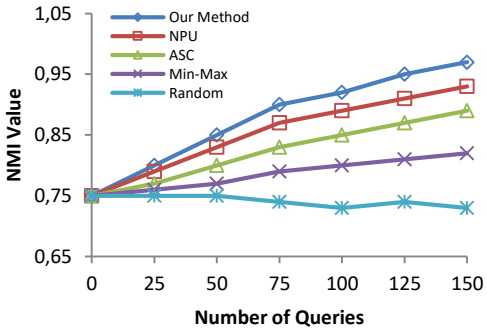
Protein



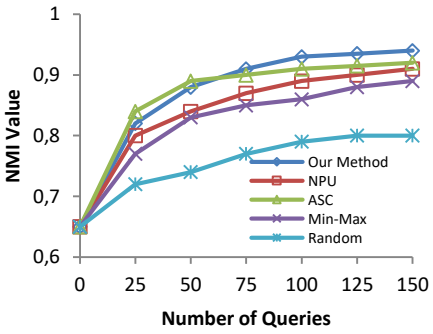
Heart



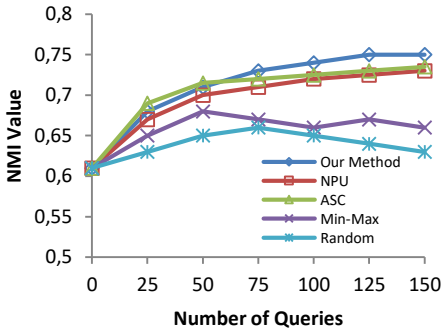
Ionosphere



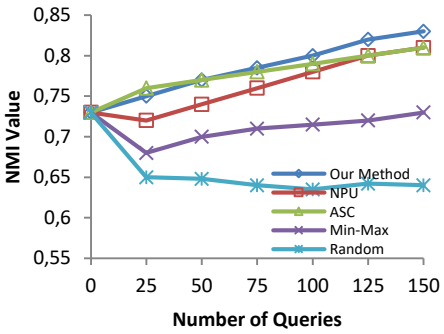
Breast



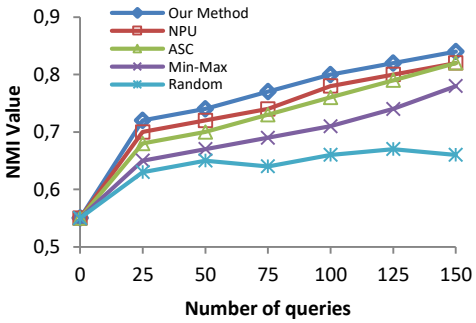
Yeast



Segment



Digit-389



Magic

Figure 2: Comparison of the constraint selection heuristics with MPCKMeans algorithm.

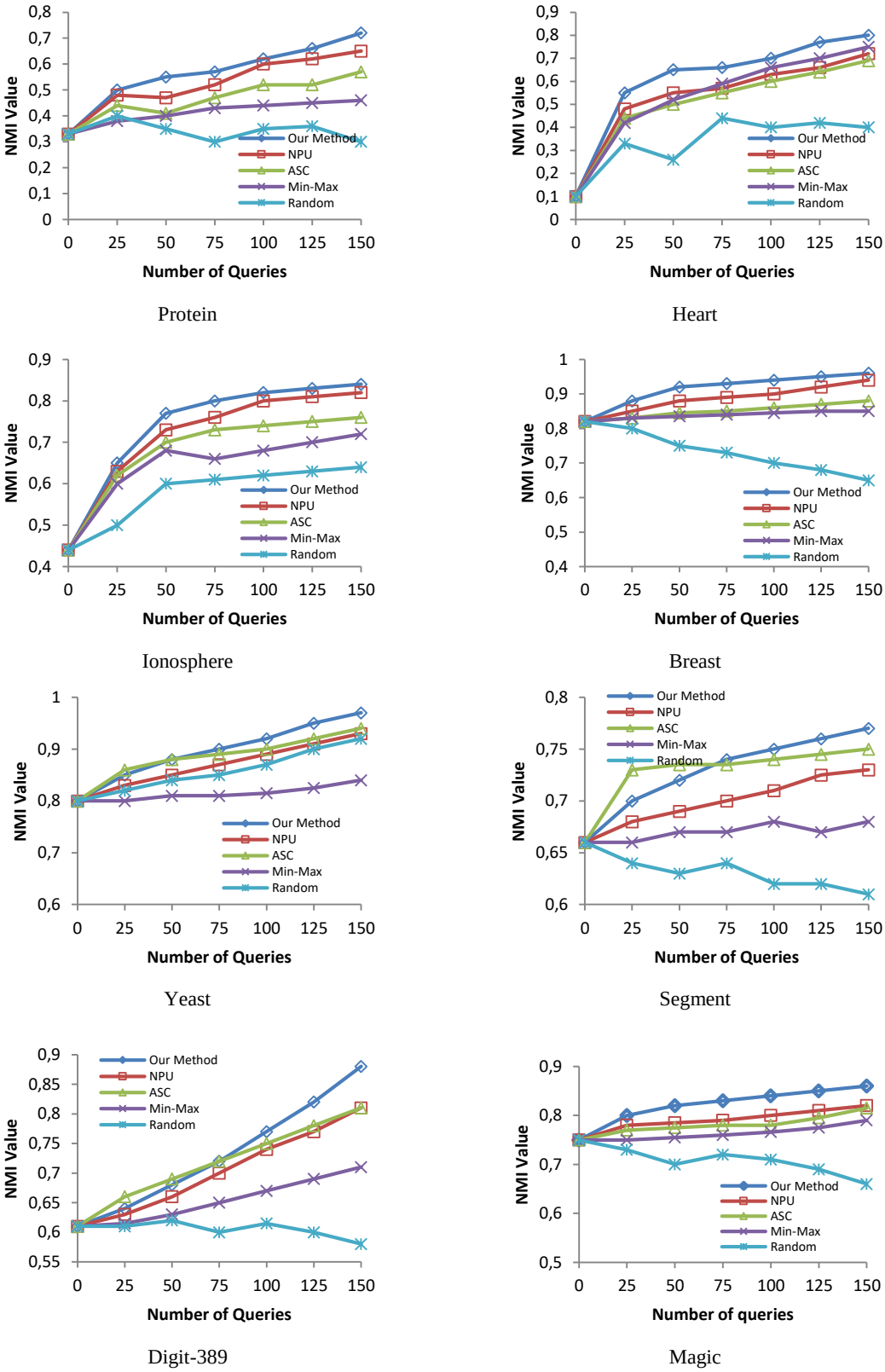


Fig. 3: Comparison of the constraint selection heuristics with AHCC algorithm.

We also use Pairwise F-measure to evaluate the clustering quality; as it focuses on how accurately we can predict the pairwise relationship between any pair of instances. Tables 1 - 2 show the pairwise F-measure results with different query size on the datasets with MPCKMeans and AHCC Algorithm respectively. The best performing method is then highlighted in boldface. Once again, the proposed active methods show a clear advantage over the baseline methods. When using small number of queries, the performance of the methods is fairly close. However, as we increase the number of queries, the proposed method becomes better than all other methods. The experimental results demonstrate that the constraints selected by the proposed active learning process are generally more beneficial for constraint-based clustering algorithms than other baseline methods.

Table 1: Comparison on Pairwise F-measure (mean ± std) with MPCKMeans Algorithm.

Dataset	Constraint Selection Heuristics	Number of Queries					
		25	50	75	100	125	150
Protein	Proposed Method	**0.63±0.045**	**0.65±0.054**	**0.66±0.055**	**0.68±0.066**	**0.72±0.066**	**0.74±0.047**
	NPU	0.59±0.047	0.60±0.042	0.62±0.046	0.64±0.048	0.66±0.052	0.69±0.052
	ASC	0.56±0.019	0.58±0.022	0.59±0.048	0.63±0.041	0.65±0.022	0.66±0.045
	Min-Max	0.49±0.001	0.52±0.004	0.54±0.022	0.55±0.005	0.55±0.005	0.57±0.015
	Random	0.44±0.035	0.45±0.035	0.48±0.038	0.52±0.038	0.55±0.038	0.56±0.038
Heart	Proposed Method	**0.68±0.002**	**0.71±0.006**	**0.73±0.005**	**0.75±0.005**	**0.75±0.008**	**0.76±0.009**
	NPU	0.65±0.011	0.66±0.044	0.67±00.72	0.68±0.007	0.70±0.036	0.71±0.074
	ASC	0.57±0.024	0.58±0.055	0.59±0.044	0.62±0.006	0.63±0.008	0.64±0.009
	Min-Max	0.55±0.004	0.57±0.012	0.57±0.012	0.57±0.014	0.58±0.015	0.59±0.015
	Random	0.48±0.025	0.49±0.030	0.52±0.030	0.55±0.033	0.55±0.036	0.56±0.036
Ionosphere	Proposed Method	**0.71±0.054**	**0.77±0.027**	**0.79±0.077**	**0.84±0.030**	**0.86±0.034**	**0.87±0.077**
	NPU	0.65±0.001	0.70±0.028	0.75±0.024	0.83±0.043	**0.86±0.025**	**0.86±0.035**
	ASC	0.67±0.024	0.69±0.056	0.74±0.032	0.81±0.064	0.83±0.057	0.84±0.065
	Min-Max	0.64±0.033	0.68±0.035	0.72±0.024	0.75±0.034	0.77±0.084	0.80±0.075
	Random	0.60±0.053	0.65±0.005	0.66±0.006	0.61±0.009	0.57±0.005	0.55±0.012
Breast	Proposed Method	**0.74±0.015**	**0.83±0.015**	**0.88±0.015**	**0.88±0.014**	**0.91±0.016**	**0.91±0.016**
	NPU	0.67±0.012	0.79±0.012	0.85±0.026	0.87±0.022	0.90±0.022	**0.90±0.022**
	ASC	0.66±0.018	0.77±0.022	0.84±0.018	0.84±0.041	0.85±0.034	0.86±0.015
	Min-Max	0.64±0.044	0.77±0.044	0.83±0.045	0.85±0.045	0.85±0.047	0.85±0.048
	Random	0.64±0.025	0.62±0.025	0.63±0.028	0.61±0.028	0.62±0.028	0.62±0.028
Yeast	Proposed Method	**0.83±0.023**	**0.85±0.035**	**0.88±0.039**	**0.91±0.045**	**0.93±0.067**	**0.94±0.078**
	NPU	0.81±0.021	0.83±0.022	0.84±00.22	0.86±0.023	0.89±0.024	0.91±0.024
	ASC	0.81±0.066	0.82±0.066	0.84±0.064	0.85±0.067	0.87±0.067	0.88±0.068
	Min-Max	0.78±0.064	0.79±0.064	0.81±0.068	0.82±0.067	0.84±0.067	0.86±0.069
	Random	0.66±0.012	0.70±0.012	0.74±0.013	0.77±0.014	0.73±0.013	0.78±0.014
Segment	Proposed Method	0.61±0.024	0.65±0.024	**0.68±0.025**	**0.69±0.026**	**0.70±0.027**	**0.71±0.027**
	NPU	0.57±0.031	0.59±0.032	0.61±0.035	0.62±0.035	0.63±0.038	0.64±0.041
	ASC	**0.64±0.081**	**0.66±0.004**	0.66±0.022	0.67±0.024	0.68±0.037	0.68±0.038
	Min-Max	0.55±0.014	0.55±0.015	0.54±0.017	0.54±0.017	0.53±0.020	0.53±0.022
	Random	0.52±0.024	0.54±0.025	0.54±0.025	0.52±0.024	0.51±0.024	0.51±0.027
Digit-389	Proposed Method	**0.73±0.025**	**0.75±0.025**	**0.77±0.025**	**0.84±0.025**	**0.86±0.026**	**0.88±0.026**
	NPU	0.68±0.018	0.70±0.018	0.75±0.019	0.81±0.021	0.83±0.022	0.84±0.023
	ASC	0.66±0.022	0.69±0.022	0.71±0.024	0.77±0.031	0.79±0.035	0.81±0.035
	Min-Max	0.65±0.014	0.68±0.014	0.70±0.015	0.73±0.015	0.74±0.017	0.75±0.018
	Random	0.70±0.055	0.73±0.055	0.75±0.058	0.70±0.058	0.72 ±0.045	0.73±0.046
Magic	Proposed Method	**0.69±0.045**	**0.74±0.045**	**0.75±0.046**	**0.75±0.041**	**0.76±0.041**	**0.76±0.039**
	NPU	**0.69±0.062**	0.68±0.062	0.69±0.065	0.70±0.065	0.71±0.066	0.72±0.067
	ASC	0.65±0.011	0.68±0.015	0.71±0.017	0.72±0.025	0.71±0.018	0.71±0.018
	Min-Max	0.61±0.057	0.61±0.057	0.62±0.057	0.61±0.057	0.61±0.059	0.61±0.062
	Random	0.60±0.074	0.60±0.074	0.61±0.074	0.61±0.078	0.60±0.078	0.60±0.079

Table 2: Comparison on Pairwise F-measure (mean ± std) with AHCC Algorithm.

Dataset	Constraint Selection Heuristics	Number of Queries					
		25	50	75	100	125	150
Protein	Proposed Method	**0.64±0.034**	**0.68±0.034**	**0.71±0.033**	**0.73±0.035**	**0.74±0.035**	**0.77±0.036**
	NPU	**0.64±0.061**	0.66±0.063	0.69±0.063	0.71±0.066	0.73±0.064	0.74±0.063
	ASC	0.63±0.025	0.64±0.022	0.65±0.027	0.65±0.025	0.66±0.022	0.67±0.022
	Min-Max	0.64±0.014	0.66±0.014	0.68±0.015	0.70±0.013	0.71±0.014	0.73±0.014
	Random	0.67±0.002	0.66±0.008	0.68±0.008	0.64±0.008	0.66±0.011	0.65±0.012
Heart	Proposed Method	**0.70±0.043**	**0.74±0.044**	**0.77±0.042**	**0.79±0.043**	**0.83±0.043**	**0.86±0.043**
	NPU	0.67±0.012	0.72±0.012	0.75±0.014	0.78±0.013	0.83±0.013	0.85±0.012
	ASC	0.68±0.026	0.71±0.022	0.74±0.033	0.78±0.064	0.82±0.021	0.84±0.032
	Min-Max	0.67±0.035	0.73±0.045	0.75±0.054	0.77±0.003	0.82±0.042	0.83±0.008
	Random	0.54±0.072	0.51±0.062	0.55±0.015	0.53±0.014	0.55±0.016	0.52±0.035
Ionosphere	Proposed Method	**0.75±0.023**	**0.78±0.023**	**0.81±0.024**	**0.84±0.024**	**0.87±0.025**	**0.90±0.030**
	NPU	0.74±0.031	0.77±0.038	0.80±0.034	0.83±0.033	**0.87±0.035**	**0.89±0.035**
	ASC	0.72±0.024	0.75±0.026	0.77±0.022	0.81±0.024	0.84±0.027	0.86±0.025
	Min-Max	0.68±0.031	0.69±0.031	0.74±0.032	0.75±0.034	0.77±0.034	0.78±0.035
	Random	0.65±0.003	0.65±0.005	0.68±0.006	0.64±0.007	0.65±0.009	0.67±0.010
Breast	Proposed Method	**0.81±0.038**	**0.83±0.038**	**0.85±0.039**	**0.87±0.040**	**0.89±0.042**	**0.90±0.042**
	NPU	**0.81±0.021**	**0.82±0.022**	0.83±00.22	0.84±0.023	0.87±0.024	0.88±0.024
	ASC	0.77±0.054	0.78±0.054	0.79±0.055	0.81±0.056	0.83±0.057	0.84±0.057
	Min-Max	0.71±0.084	0.71±0.084	0.74±0.088	0.75±0.087	0.76±0.087	0.77±0.089
	Random	0.66±0.005	0.65±0.005	0.60±0.006	0.59±0.007	0.57±0.008	0.55±0.008
Yeast	Proposed Method	0.82±0.043	0.84±0.045	**0.87±0.049**	**0.90±0.045**	**0.92±0.067**	**0.93±0.078**
	NPU	**0.84±0.051**	**0.86±0.052**	**0.87±00.52**	**0.90±0.023**	0.91±0.024	**0.92±0.024**
	ASC	0.80±0.066	0.83±0.066	0.84±0.064	0.85±0.067	0.86±0.067	0.87±0.068
	Min-Max	0.80±0.074	0.83±0.074	0.84±0.078	0.87±0.067	0.88±0.067	0.89±0.069
	Random	0.73±0.022	0.77±0.022	0.79±0.023	0.82±0.014	0.84±0.013	0.86±0.014
Segment	Proposed Method	0.60±0.035	0.61±0.037	**0.63±0.059**	**0.65±0.024**	**0.67±0.037**	**0.69±0.026**
	NPU	0.58±0.017	0.59±0.017	0.60±0.018	0.60±0.032	0.62±0.053	0.63±0.035
	ASC	**0.61±0.034**	**0.62±0.028**	0.62±0.030	0.62±0.004	0.62±0.021	0.63±0.054
	Min-Max	0.57±0.045	0.58±0.045	0.58±0.045	0.59±0.025	0.60±0.055	0.60±0.067
	Random	0.52±0.012	0.54±0.012	0.53±0.013	0.51±0.025	0.50±0.014	0.48±0.024
Digit-389	Proposed Method	0.80±0.055	**0.83±0.055**	**0.86±0.056**	**0.88±0.055**	**0.91±0.057**	**0.92±0.055**
	NPU	**0.81±0.001**	0.83±0.002	0.85±0.002	0.86±0.003	0.89±0.004	**0.91±0.004**
	ASC	0.78±0.026	0.81±0.026	0.83±0.025	0.85±0.027	0.86±0.027	0.88±0.028
	Min-Max	0.76±0.044	0.79±0.044	0.81±0.048	0.82±0.047	0.84±0.047	0.86±0.049
	Random	0.70±0.012	0.70±0.012	0.74±0.013	0.77±0.014	0.78±0.013	0.82±0.014
Magic	Proposed Method	**0.64±0.033**	**0.70±0.035**	**0.72±0.035**	**0.74±0.036**	**0.75±0.037**	**0.77±0.038**
	NPU	0.58±0.007	0.62±0.007	0.66±0.008	0.67±0.012	0.68±0.013	0.69±0.012
	ASC	0.57±0.024	0.58±0.028	0.61±0.030	0.64±0.011	0.66±0.021	0.67±0.022
	Min-Max	0.56±0.005	0.58±0.005	0.58±0.005	0.61±0.004	0.62±0.005	0.62±0.008
	Random	0.48±0.012	0.51±0.012	0.55±0.013	0.51±0.014	0.49±0.014	0.47±0.015

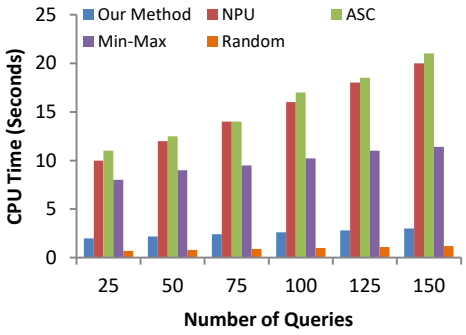
4.5.2 Efficiency and Scalability Analysis

One of the primary motivations of the proposed active learning method is to reduce the amount of computation in iterative active learning. Figure 4 shows the average CPU time of the proposed selection method (using batch size value b = 10) with the baseline methods on a 3.5 GHz Intel Core i5 and 4 GB main memory. The horizontal axis indicates the total number of queries and the vertical axis shows the CPU running time (in seconds).

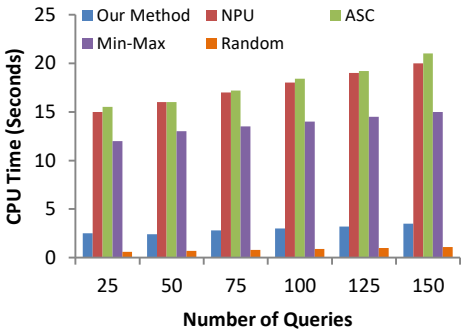
From the results, we clearly see that the proposed method is significantly more efficient and scalable than other methods. As we observe, when the dataset size increases, the time cost of NPU and ASC increases dramatically, while the proposed method increases linearly. Specifically, on the magic dataset with 150 queries, NPU takes about 240 seconds and ASC takes about 260 seconds, while the proposed method needs only about 24 second. Hence, we can conclude that the proposed method is more efficient and scalable than other methods for large applications. Finally, as indicated in Figure 4, the time cost of the proposed method increases moderately as the number

of queries increases. While, the time cost of other methods increases tremendously as the number of queries increases.

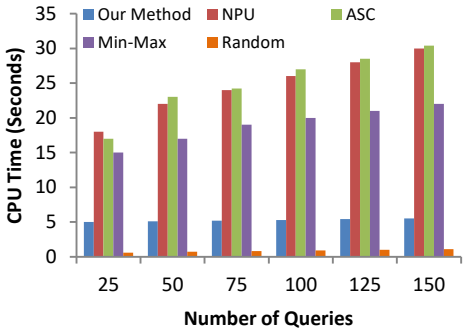
Generally, the results show that the proposed method is slower than Random method that selects set of constraints randomly without performing additional CPU time for selecting constraints.



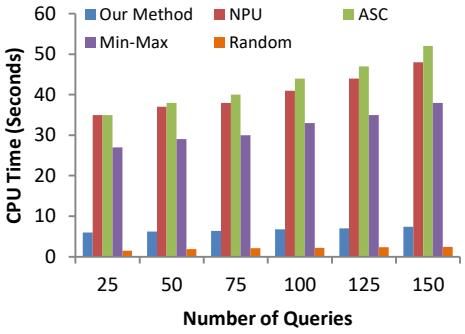
Protein



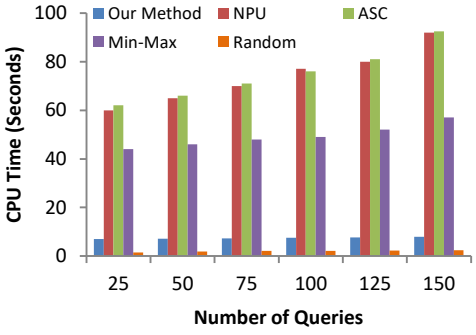
Heart



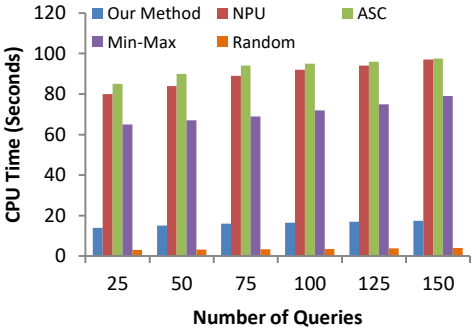
Ionosphere



Breast



Yeast



Segment

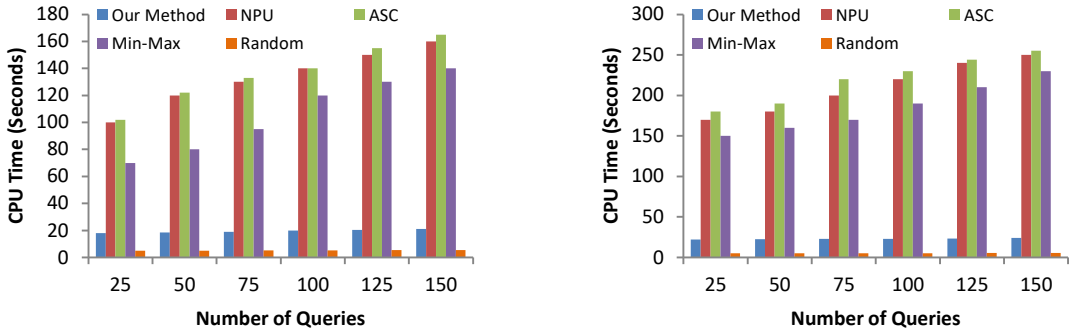
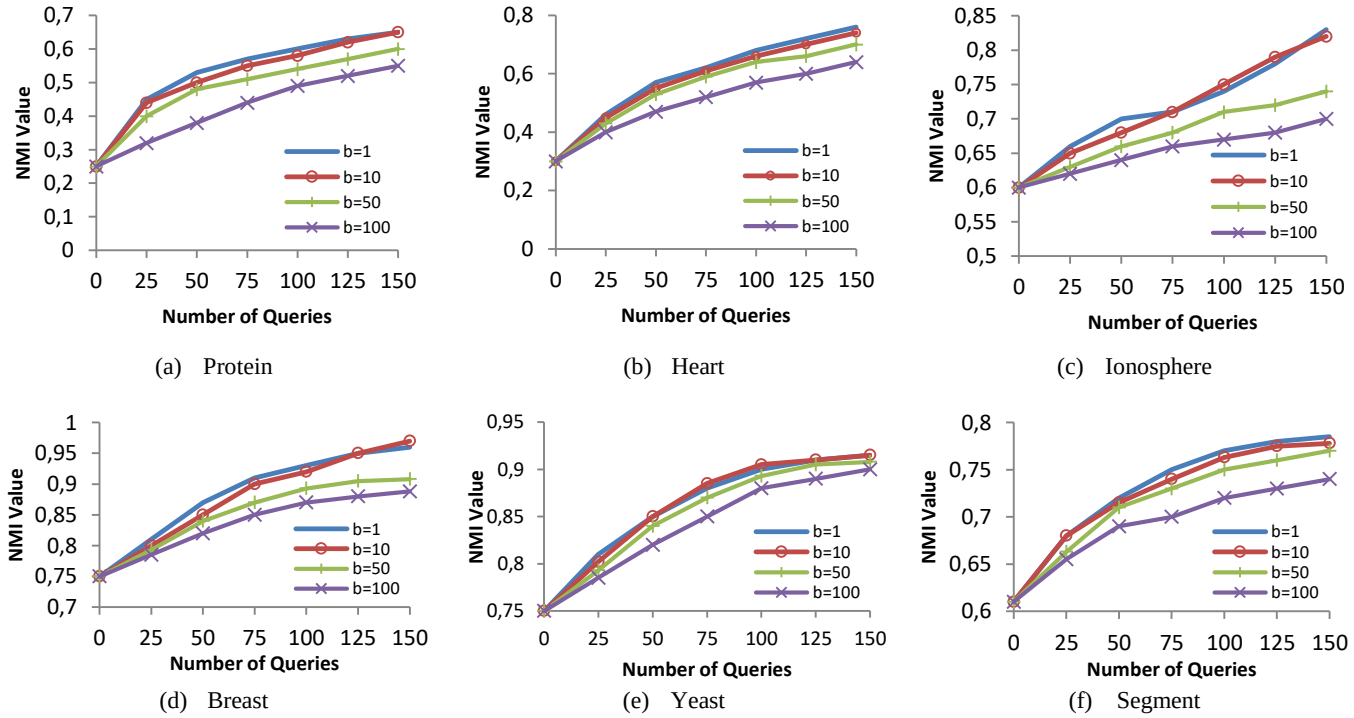
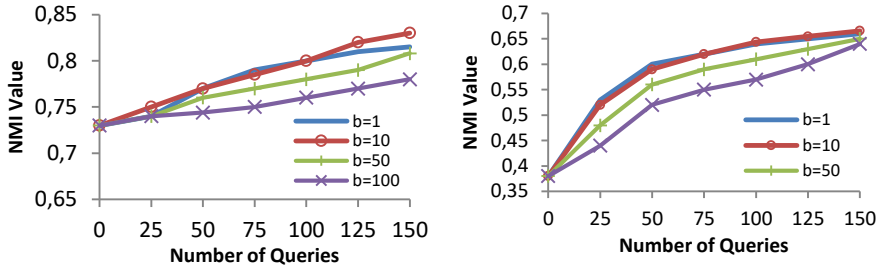


Fig. 4: Comparison of average run time over different number of queries with batch size value $b = 10$.

4.3.3 Analysis of Different Batch Size Values

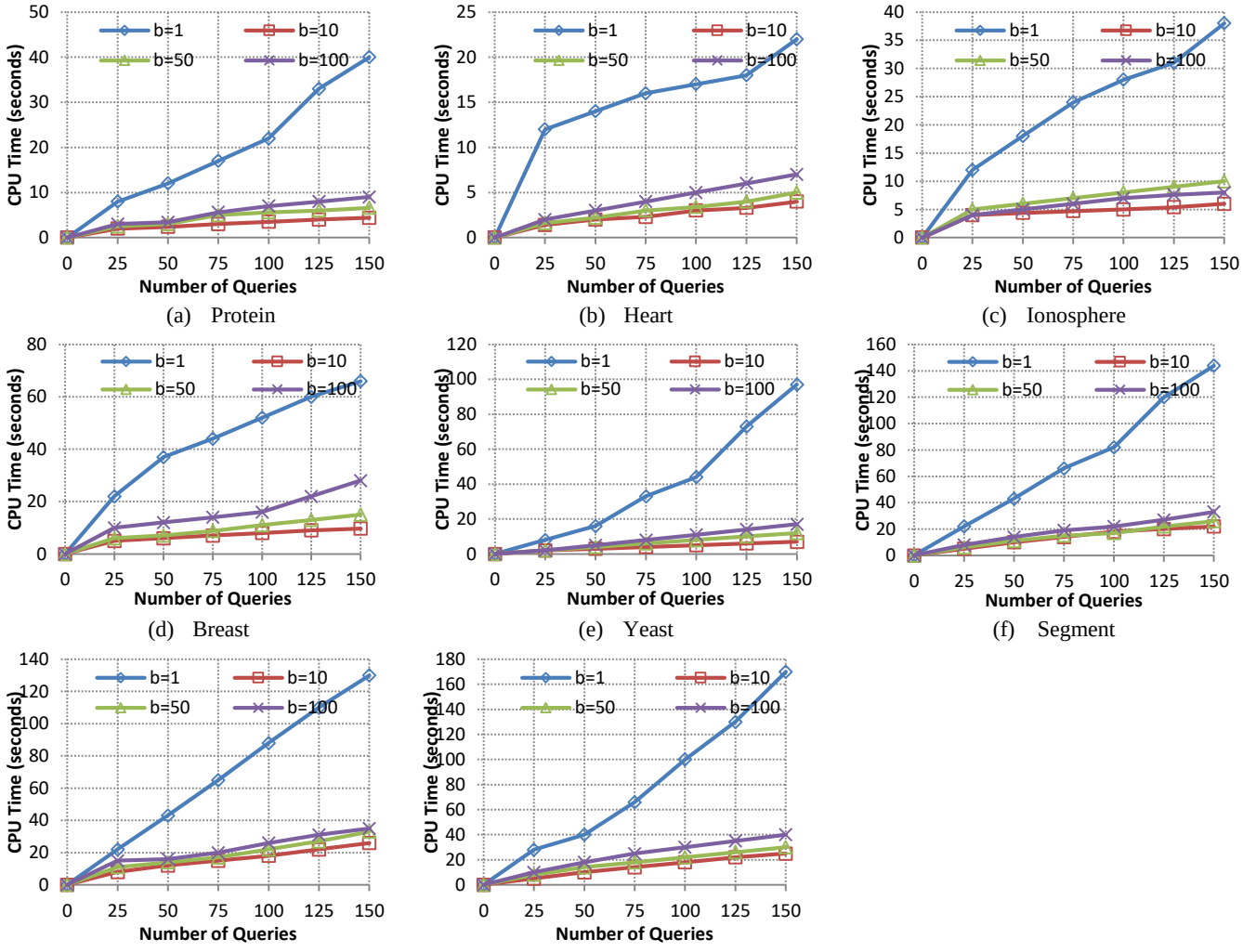
In this section, we carried out an analysis of the proposed active learning with varying the value of the batch size b. Figure 5 shows the performance versus the number of queries on the datasets with MPCKMeans algorithm. From Figure 5, selecting small b values results in similar (or better) performance compared to those obtained selecting only one instance. On the contrary, high b values decrease the performance without decreasing the computational time if compared to small b values. Figure 6 shows the computational time taken for different b values on both yeast and magic datasets. From Figure 6, it could be noticed that the largest learning time is obtained in the case where one instance is selected (i.e. $b = 1$). Also, we can notice that the CPU time is increased as the value of batch size b is increased. Therefore, it could be interesting to automatically identify the best value of the batch size b that can achieve the best clustering performance.





(g) Digit-389 (h) Magic

Fig.5: Clustering performance versus different batch size



(a) Protein (b) Heart (c) Ionosphere

(d) Breast (e) Yeast (f) Segment

(g) Digit-389 (h) Magic

Fig. 6: CPU time versus different batch size

5. CONCLUSION AND FUTURE WORK

Identifying the most beneficial set of clustering constraints was considered in this paper. We present an iterative active learning method that selects a batch of query instances from the

unlabeled data so that the marginal probability distribution represented by the labeled data after annotation, is similar to the marginal probability distribution represented by the unlabeled data. Moreover, incrementally expands the neighborhoods by using the selected queries. Experiments carried out on different real datasets show that the constraints selected by the proposed active learning process are generally more beneficial for constraint-based clustering algorithms than those provided by the NPU and ASC methods and achieving high clustering performance with minimizing the amount of computation for selecting the active constraints. However, the efficiency of the proposed method strongly depends on the value of the batch size *b*. In future work, we are interesting to automatically identify the best value of the batch size *b* that achieves the best clustering performance. Also, the problem of clustering the big data with an incrementally growing constraint set. To address this problem, we interest to consider an incremental semi-supervised clustering method.

REFERENCES

1. Basu, S., Banerjee, A., & Mooney, R. J. (2004, April). Active semi-supervision for pairwise constrained clustering. In *Proceedings of the 2004 SIAM international conference on data mining* (pp. 333-344). Society for Industrial and Applied Mathematics.
2. Yu, C., & Hansen, J. H. (2017). Active learning based constrained clustering for speaker diarization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(11), 2188-2198.
3. Van Craenendonck, T., & Blockeel, H. (2017). Constraint-based clustering selection. *Machine Learning*, 106(9-10), 1497-1521.
4. Yu, Z., Luo, P., You, J., Wong, H. S., Leung, H., Wu, S.,& Han, G. (2015). Incremental semi-supervised clustering ensemble for high dimensional data clustering. *IEEE Transactions on Knowledge and Data Engineering*, 28(3), 701-714.
5. Borgwardt, K. M., Gretton, A., Rasch, M. J., Kriegel, H. P., Schölkopf, B., & Smola, A. J. (2006). Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14), e49-e57.
6. Sriperumbudur, B. K., Gretton, A., Fukumizu, K., Schölkopf, B., & Lanckriet, G. R. (2010). Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11(Apr), 1517-1561.
7. Calma, A., Reitmaier, T., & Sick, B. (2018). Semi-supervised active learning for support vector machines: A novel approach that exploits structure information in data. *Information Sciences*, 456, 13-33.
8. Xiong, C., Johnson, D. M., & Corso, J. J. (2016). Active clustering with model-based uncertainty reduction. *IEEE transactions on pattern analysis and machine intelligence*, 39(1), 5-17.
9. Ngoc, M. T., & Park, D. C. (2018). Centroid Neural Network with Pairwise Constraints for Semi-supervised Learning. *Neural Processing Letters*, 48(3), 1721-1747.

10. Zeng, H., & Cheung, Y. M. (2011). Semi-supervised maximum margin clustering with pairwise constraints. *IEEE Transactions on Knowledge and Data Engineering*, 24(5), 926-939.
11. Wang, G., Hwang, J. N., Rose, C., & Wallace, F. (2018). Uncertainty-Based Active Learning via Sparse Modeling for Image Classification. *IEEE Transactions on Image Processing*, 28(1), 316-329.
12. Mallapragada, P. K., Jin, R., & Jain, A. K. (2008). Active query selection for semi-supervised clustering. In *2008 19th International Conference on Pattern Recognition*, 1-4.
13. Xu, Q., & Wagstaff, K. L. (2005). Active constrained clustering by examining spectral eigenvectors. In *International Conference on Discovery Science*, 294-307.
14. Wang, X., & Davidson, I. (2010). Active spectral clustering. In *2010 IEEE International Conference on Data Mining* , 561-568.
15. Vu, V. V., Labroche, N., & Bouchon-Meunier, B. (2012). Improving constrained clustering with active query selection. *Pattern Recognition*, 45(4), 1749-1758.
16. Vu, V. V., Labroche, N., & Bouchon-Meunier, B. (2010). An efficient active constraint selection algorithm for clustering. In 2010 *20th International Conference on Pattern Recognition*, 2969-2972.
17. Xiong, S., Azimi, J., & Fern, X. Z. (2013). Active learning of constraints for semi-supervised clustering. IEEE Transactions on Knowledge and Data Engineering, 26(1), 43-54.
18. Li, Y., li Wang, Y., Yu, D. J., Ning, Y., Hu, P., & Zhao, R. (2019). ASCENT: Active Supervision for Semi-supervised Learning. *IEEE Transactions on Knowledge and Data Engineering.*
19. Bilenko, M., Basu, S., & Mooney, R. J. (2004, July). Integrating constraints and metric learning in semi-supervised clustering. In Proceedings of the twenty-first international conference on Machine learning (p. 11). ACM.
20. Davidson, I., & Ravi, S. S. (2005). Agglomerative hierarchical clustering with constraints: Theoretical and empirical results. In *European Conference on Principles of Data Mining and Knowledge Discovery* , 59-70.
21. Cover, T. M., & Thomas, J. A. (2012*). Elements of information theory*. John Wiley & Sons.
22. Xiong, S., Pei, Y., Rosales, R., & Fern, X. Z. (2015). Active learning from relative comparisons. *IEEE Transactions on Knowledge and Data Engineering*, 27(12), 3166-3175.
23. Davidson, I., Wagstaff, K. L., & Basu, S. (2006). Measuring constraint-set utility for partitional clustering algorithms. In *European conference on principles of data mining and knowledge discovery*, 115-126.

