

Active Learning of Constraints for Semi-Supervised Clustering

Sicheng Xiong, Javad Azimi, and Xiaoli Z. Fern

Abstract—Semi-supervised clustering aims to improve clustering performance by considering user supervision in the form of pairwise constraints. In this paper, we study the active learning problem of selecting pairwise must-link and cannot-link constraints for semi-supervised clustering. We consider active learning in an iterative manner where in each iteration queries are selected based on the current clustering solution and the existing constraint set. We apply a general framework that builds on the concept of neighborhood, where neighborhoods contain “labeled examples” of different clusters according to the pairwise constraints. Our active learning method expands the neighborhoods by selecting informative points and querying their relationship with the neighborhoods. Under this framework, we build on the classic uncertainty-based principle and present a novel approach for computing the uncertainty associated with each data point. We further introduce a selection criterion that trades off the amount of uncertainty of each data point with the expected number of queries (the cost) required to resolve this uncertainty. This allows us to select queries that have the highest information rate. We evaluate the proposed method on the benchmark data sets and the results demonstrate consistent and substantial improvements over the current state of the art.

Index Terms—Active learning, clustering, semi-supervised learning

1 INTRODUCTION

SEMI-SUPERVISED clustering aims to improve clustering performance with the help of user-provided side information. One of the most studied types of side information is pairwise constraints, which include must-link and cannot-link constraints specifying that two points must or must not belong to the same cluster. A number of previous studies have demonstrated that, in general, such constraints can lead to improved clustering performance [1], [2], [3]. However, if the constraints are selected improperly, they may also degrade the clustering performance [4], [5]. Moreover, obtaining pairwise constraints typically requires a user to manually inspect the data points in question, which can be time consuming and costly. For example, for document clustering, obtaining a must-link or cannot-link constraint requires a user to potentially scan through the documents in question and determine their relationship, which is feasible but costly in time. For those reasons, we would like to optimize the selection of the constraints for semi-supervised clustering, which is the topic of active learning.

While active learning has been extensively studied in supervised learning [6], [7], [8], [9], [10], [11], the research on active learning of constraints for semi-supervised clustering is relatively limited [1], [5], [12], [13], [14]. Most of the existing work on this topic has focused on selecting an initial set of constraints prior to performing semi-supervised clustering [1], [5], [13], [14]. This is not suitable if we wish

to iteratively improve the clustering model by actively querying the user.

In this paper, we consider active learning of constraints in an iterative framework. Specifically, in each iteration we determine what is the most important information toward improving the current clustering model and form queries accordingly. The responses to the queries (i.e., constraints) are then used to update (and improve) the clustering. This process repeats until we reach a satisfactory solution or we reach the maximum number of queries allowed. Such an iterative framework is widely used in active learning for supervised classification [7], [8], [9], [10], and has been generally observed to outperform noniterative methods, where the whole set of queries is selected in a single batch.

We focus on a general approach based on the concept of neighborhoods, which has been successfully used in a number of previous studies on active acquisition of constraints [1], [12], [13]. A neighborhood contains a set of data points that are known to belong to the same cluster according to the constraints and different neighborhoods are known to belong to different clusters. Simply put, neighborhoods can be viewed as containing the “labeled examples” of different clusters. Well-formed neighborhoods can provide valuable information regarding what the underlying clusters look like. Analogous to supervised active learning, an active learner of constraints will then seek to select the most informative data point to include in the neighborhoods. Once a point is selected, we query the selected point against the existing neighborhoods to determine to which neighborhood it belongs.

Specifically, our approach builds on the classic uncertainty-based principle. Here, we define the uncertainty in terms of the probability of the point belonging to different known neighborhoods and propose a novel nonparametric approach using random forest [15] for estimating the probabilities. Different from supervised learning where each point only requires one query to obtain its label, in

• S. Xiong and X.Z. Fern are with the Oregon State University, 1148 KEC, Corvallis, OR 97330. E-mail: {xiongxi, xfern}@eecs.oregonstate.com.

• J. Azimi is with InsightsOne Inc., 355 N Wolfe Rd Apt 734, Sunnyvale, CA 94085. E-mail: azimi@insightstone.com

Manuscript received 9 July 2012; revised 21 Oct. 2012; accepted 27 Nov. 2012; published online 24 Jan. 2013.

Recommended for acceptance by J. Sander.

For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2012-07-0481. Digital Object Identifier no. 10.1109/TKDE.2013.22.

semi-supervised clustering, we can only pose pairwise queries and it typically takes multiple queries to determine the neighborhood of a selected point. In general, points with higher uncertainty will require larger number of queries. This suggests that there is a tradeoff between the amount of information we acquire by querying about a point, and the expected number of queries (cost) for acquiring this information. We propose to balance this tradeoff by normalizing the amount of uncertainty of each data point by the expected number of queries required to resolve this uncertainty, and as such, select queries that have the highest rate of information.

Note that an obvious alternative approach would be to evaluate all potential pairs and select the one that has the highest uncertainty regarding whether they are must-linked or cannot-linked. This idea has previously been explored by Huang and Lam [12] in the context of document clustering. In this paper, we note a critical issue with this approach that it only considers the pairwise uncertainty of the first query and fails to measure the benefit of the ensuing queries that are required to determine the neighborhood for a point. Our method, instead, focuses on the point-based uncertainty, allowing us to select the queries according to the total amount of information gained by the full sequence of queries as a whole.

We empirically evaluate the proposed method on eight data sets of different complexity. The evaluation results demonstrate that our method achieves consistent and substantial improvements over three competing methods.

The remainder of the paper is organized as follows: Section 2 presents a brief review of the related work on active learning of constraints. Section 3 introduces our general active learning framework and the proposed method within the framework. Experimental evaluations are presented in Section 4. Finally, we conclude the paper and discuss future directions in Section 5.

2 RELATED WORK

Active learning has been studied extensively for supervised classification problems [6], [7], [8], [9], [10], [11]. In contrast, the research on active learning for constraint-based clustering has been limited. As mentioned previously, most of the existing research studied the selection of a set of initial constraints prior to performing semi-supervised clustering. Specifically, the first study on this topic was conducted by Basu et al. [1]. They proposed a two-phase approach, which we refer to as the Explore and Consolidate (E & C) approach. The first phase (Explore) incrementally selects points using the farthest-first traversal scheme and queries their relationship to identify c disjoint neighborhoods, where c is the total number of clusters. 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. More recently, Mallapragada et al. [13] proposed an improvement to Explore and Consolidate named Min-Max, which modifies the consolidate phase by choosing the most uncertain point to query (as opposed to randomly).

Xu et al. [14] proposed to select constraints by examining the spectral eigenvectors of the similarity matrix, which is unfortunately limited to two-cluster problems. In [5], [16],

constraints are selected by analyzing the co-association matrix (obtained by applying cluster ensembles to the data). A key distinction of our method from the above-mentioned work is that we iteratively select the next set of queries based on the current clustering assignment to improve the solution. This is analogous to supervised active learning where data points are selected iteratively based on the current classification model such that the model can be improved most efficiently [7], [8], [9], [10].

More relevant to our work is an active learning framework presented by Huang and Lam [12] for the task of document clustering. Specifically, this framework takes an iterative approach that is similar to ours. In each iteration, their method performs semi-supervised clustering with the current set of constraints to produce a probabilistic clustering assignment. It then computes, for each pair of documents, the probability of them belonging to the same cluster and measures the associated uncertainty. To make a selection, it focuses on all unconstrained pairs that has exactly one document already “assigned to” one of the existing neighborhoods by the current constraint set, and among them identifies the most uncertain pair to query. If a “must-link” answer is returned, it stops and moves onto the next iteration. Otherwise, it will query the unassigned point against the existing neighborhoods until a “must-link” is returned.

While Huang’s method is developed specifically for document clustering, one could potentially apply the underlying active learning approach to handle other types of data by assuming appropriate probabilistic models. We would like to highlight a key distinction between Huang’s method and our work, that is Huang’s method makes the selection choice based on *pairwise uncertainty*, whereas we focus on the uncertainty of a point in terms of which neighborhood it belongs to. This difference is subtle, but important. Pairwise uncertainty captures only the relationship between the two points in the pair. Depending on the outcome of the query, we may need to go through a sequence of additional queries. Huang’s method only considers the pairwise uncertainty of the first query, and fails to measure the benefit of the ensuing queries. This is why our method instead focuses on point-based uncertainty, which measures the total amount of information gained by the full sequence of queries as a whole. Furthermore, our method also takes into account the expected number of queries to resolve the uncertainty of a point, which has not been considered previously.

Finally, we want to mention another line of work that uses active learning to facilitate clustering [17], [18], where the goal is to cluster a set of objects by actively querying the distances between one or more pairs of points. This is different from the focus of this paper, where we only request pairwise must-link and cannot-link constraints, and do not require the user to provide specific distance values.

3 METHODOLOGY

The problem addressed in this paper is how to effectively choose pairwise queries to produce an accurate clustering assignment. Through active learning, we aim to achieve query efficiency, i.e., we would like to reduce the number of queries/questions asked to achieve a good clustering

performance. We view this as an iterative process such that the decision for selecting queries should depend on what has been learned from all the previously formulated queries. In this section, we will introduce our proposed method. Below, we will begin by providing a precise formulation of our active learning problem.

3.1 Problem Formulation

Formally, we define the problem as follows: given a set of data instances $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, we assume that there exists an underlying class structure that assigns each data instance to one of the c classes. We denote the unknown labels by $\mathbf{y} = \{y_1, \dots, y_n\}$, each label $y_i \in \mathcal{Y} \triangleq \{1, \dots, c\}$, $\forall i \in \{1, \dots, n\}$. In this setting, we cannot (directly) observe these labels. Instead, information can be obtained through query of the form: *Do instances \mathbf{x}_i and \mathbf{x}_j belong to the same class?* We denote a query by a pair of instances $(\mathbf{x}_i, \mathbf{x}_j)$, and the answer to the query by $l_{ij} \in \mathcal{A} \triangleq \{\text{ML}, \text{CL}\}$. In particular, the label “ML” (“CL”) is returned if $y_i = y_j$ ($y_i \neq y_j$). In each iteration, we need to select one or more queries based on \mathcal{D} and the current set of constraints \mathcal{C} .

Note that must-link and cannot-link constraints satisfy the following properties:

- $(\mathbf{x}_i, \mathbf{x}_j, \text{ML}) \wedge (\mathbf{x}_i, \mathbf{x}_k, \text{ML}) \Rightarrow (\mathbf{x}_j, \mathbf{x}_k, \text{ML})$
- $(\mathbf{x}_i, \mathbf{x}_j, \text{ML}) \wedge (\mathbf{x}_i, \mathbf{x}_k, \text{CL}) \Rightarrow (\mathbf{x}_j, \mathbf{x}_k, \text{CL})$.

Based on these properties, we introduce the concept of neighborhood, which is instrumental in the design of many existing methods for active learning of pairwise constraints [1], [12], [13].

3.2 A Neighborhood-Based Framework

Definition 1. A neighborhood contains a set of data instances that are known to belong to the same class (i.e., connected by must-link constraints). Furthermore, different neighborhoods are connected by cannot-link constraints and, thus, are known to belong to different classes.

Given a set of constraints denoted by \mathcal{C} , we can identify a set of l neighborhoods $\mathcal{N} = \{N_1, \dots, N_l\}$, such that $l \leq c$ and c is the total number of classes. Consider a graph representation of the data where vertices represent data instances, and edges represent must-link constraints. The neighborhoods, which are denoted by $N_i \subset \mathcal{D}, i \in \{1, \dots, l\}$, are simply the connected components of the graph that have cannot-link constraints between one another. Note that if there exists no cannot-link constraints, we can only identify a single known neighborhood even though we may have multiple connected components because some connected components may belong to the same class. In such cases, we will treat the largest connected component as the known neighborhood.

Fig. 1 illustrates two examples that explain how we can form the neighborhoods from a set of pairwise constraints. The nodes denote data instances, and the solid lines denote must-link constraints while the dashed lines denote cannot-link constraints. Note that in our definition, each neighborhood is required to have a cannot-link constraint with all other neighborhoods. Therefore, Fig. 1a contains three neighborhoods: $\{\mathbf{x}_1, \mathbf{x}_2\}$, $\{\mathbf{x}_3\}$, and $\{\mathbf{x}_4\}$, whereas Fig. 1b contains only two known neighborhoods, which can be either $\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3\}$ or $\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_4\}$.

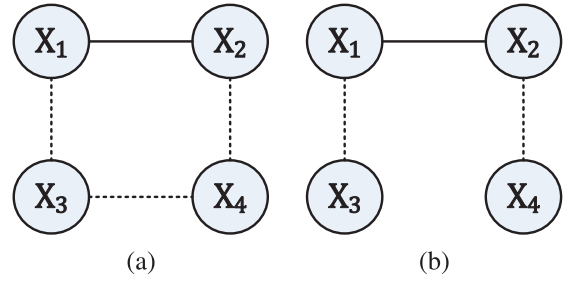


Fig. 1. Two examples to show how to identify neighborhoods from a set of pairwise constraints.

One way to interpret the neighborhoods is to view them as the “labeled examples” of the underlying classes because instances belonging to different neighborhoods are guaranteed to have different class labels, and instances of the same neighborhood must belong to the same class. A key advantage of using the neighborhood concepts is that by leveraging the knowledge of the neighborhoods, we can acquire a large number of constraints via a small number of queries. In particular, if we can identify the neighborhood of an instance \mathbf{x} , we can immediately infer its pairwise relationship with all other points that are currently confirmed to belong to any of the existing neighborhoods. This naturally motivates us to consider an active learning strategy that incrementally expands the neighborhoods by selecting the most informative data point and querying it against the known neighborhoods. We summarize this strategy in Algorithm 1.

Algorithm 1. The Neighborhood-based Framework.

Input: A set of data points \mathcal{D} ; the total number of classes c ; the maximum number of pairwise queries T .

Output: a clustering of \mathcal{D} into c clusters.

```

1: Initializations:  $\mathcal{C} = \emptyset$ ;  $N_1 = \{\mathbf{x}\}$ , where  $\mathbf{x}$  is a random
   point in  $\mathcal{D}$ ;  $\mathcal{N} = N_1$ ;  $l = 1$ ;  $t = 0$ ;
2: repeat
3:    $\pi = \text{Semi-supervised-Clustering}(\mathcal{D}, \mathcal{C})$ ;
4:    $\mathbf{x}^* = \text{MostInformative}(\mathcal{D}, \pi, \mathcal{N})$ ;
5:   for each  $N_i \in \mathcal{N}$  in decreasing order of  $p(\mathbf{x}^* \in N_i)$ 
     do
6:     Query  $\mathbf{x}^*$  against any data point  $\mathbf{x}_i \in N_i$ ;
7:      $t++$ ;
8:     Update  $\mathcal{C}$  based on returned answer;
9:     if  $(\mathbf{x}^*, \mathbf{x}_i, \text{ML})$  then  $N_i = N_i \cup \{\mathbf{x}^*\}$ ; break;
10:  end for
11:  if no must-link is achieved
12:    then  $l++$ ;  $N_l = \{\mathbf{x}^*\}$ ;  $\mathcal{N} = \mathcal{N} \cup N_l$ ;
13: until  $t > T$ 
14: return Semi-supervised-clustering( $\mathcal{D}, \mathcal{C}$ )

```

Briefly, the algorithms begin by initializing the neighborhoods by selecting a random point to be the initial neighborhood (line 1). In each iteration, given the current set of constraints \mathcal{C} , it performs semi-supervised clustering on \mathcal{D} to produce a clustering solution π (line 3). A selection criterion is then applied to select the “most informative” data point \mathbf{x}^* based on the current set of neighborhoods and the clustering solution π (line 4). The selected point \mathbf{x}^* is then queried against each existing neighborhood N_i to identify where \mathbf{x}^* belongs, during which the constraint set \mathcal{C}

is updated (lines 5-12). In line 5, we go through the neighborhoods in decreasing order based on $p(\mathbf{x}^* \in N_i)$, $i \in \{1, \dots, l\}$, i.e., the probability of \mathbf{x}^* belonging to each neighborhood, which is assumed to be known. This query order will allow us to determine the neighborhood of \mathbf{x}^* with the smallest number of queries. This process is repeated until we reach the maximum number of queries allowed (line 13).

In this work, we consider the semi-supervised clustering algorithm as a black box and any existing algorithm can be used here. The key question we aim to answer is how to select the “most informative” instance to query against, i.e., the design of the function `MostInformative` in line 4. In the remaining part of this section, we will focus on this question and describe our proposed solution.

3.3 Selecting the Most Informative Instance

Given a set of existing neighborhoods, we would like to select an instance such that knowing its neighborhood will allow us to gain maximal information about the underlying clustering structure of the data. Our method is based on the following key observation. If we can predict with high certainty to which neighborhood an instance belongs based on our current understanding of the clustering structure, querying about that instance will not lead to any gain of information. Similar observations have been used to motivate the widely used uncertainty-based sampling principle for active learning of classifiers [11].

To apply uncertainty-based sampling for selecting the most informative instance, we need to address a number of important issues. First, how can we reliably measure the uncertainty of a data point? Second, we can only ask pairwise queries, and it may take multiple queries to resolve the uncertainty about a data point. How can we take this into consideration in our decision? Below we present our approach for dealing with these two issues.

3.3.1 Measuring Uncertainty

In uncertainty-based sampling for supervised learning, an active learner queries the instance about which the label uncertainty is maximized. Numerous studies have investigated different approaches for measuring uncertainty given probabilistic predictions of the class labels [11]. In our context, one can take a similar approach and measure the uncertainty of each data instance belonging to different clusters.

For example, one could also take a model-based clustering approach such as mixture of Gaussians, which will allow us to produce a probabilistic assignment of each instance to different Gaussian clusters and then compute the associated uncertainty in cluster assignment using measures such as entropy. While this may appear to be a natural approach, there are some potential issues. First, this approach can be overly sensitive to the current clustering solution. An ill-formed clustering solution will lead to meaningless models and poor choices of queries. Furthermore, it is also sensitive to the modeling assumption. If the clusters cannot be properly represented using the assumed model, this approach will not provide reliable uncertainty estimation.

To address the first issue, we will not estimate the probability of an instance belonging to each of the learned clusters. Instead, we estimate the probability of it belonging to each of the existing “labeled” neighborhoods. In

particular, we will estimate the neighborhood assignment probability of a particular instance by focusing on its relationship with only the labeled instances (those with known neighborhood assignments). All other instances are ignored for this purpose because their cluster assignment may be inaccurate and misleading. As such, we reduce the sensitivity of our method to poor clustering results. Since different neighborhoods contain “labeled” examples of different underlying clusters, this will allow us to more accurately estimate the probability of an instance belonging to the true underlying clusters. Given the neighborhood assignment probability, we then compute the uncertainty of an instance using a measure such as entropy.¹

To address the second issue, i.e., the sensitivity to the probabilistic assumptions, we will avoid making such assumptions at all. Instead, our approach estimates the probability of each instance belonging to each neighborhood using a similarity-based approach, where the similarity measure is discriminatively learned under the supervision of the current clustering solution. That is, for instances belonging to different clusters (according to the current solution) their learned similarity will likely to be low and vice versa. This learning-based approach allows us to transfer the knowledge that we have learned from the constraints to the similarity measures.

Below we explain in detail how we learned the similarities and estimate the probabilities.

We are given a current clustering assignment π , and a set of neighborhoods $\mathcal{N} = \{N_1, \dots, N_l\}$. Our goal is to estimate $p(\mathbf{x} \in N_i)$, i.e., for each data instance $\mathbf{x} \in \mathcal{D}$, the probability of it belonging to N_i for $i \in \{1, \dots, l\}$.

Supervised learning of similarities. We first learn a similarity measure using a random forest-based approach. In particular, we leverage the current clustering assignment π by creating a labeled training set using $\pi(\mathbf{x})$ as the label of \mathbf{x} for all $\mathbf{x} \in \mathcal{D}$. Using this training set, we build a random forest classifier (containing 50 decision trees) that predicts the cluster label $\pi(\mathbf{x})$ from \mathbf{x} . Random forest [15] is an ensemble learning algorithm that learns a collection of decision trees. Each decision tree is trained using a randomly bootstrapped sample of the training set and the test for each node of the tree is selected from a random subset of the features. While one could also use other supervised classifiers, we choose random forest because it is not prone to overfitting [19], and as described below it provides a natural definition of similarities (proximities) among instances once a random forest is built.

Given the learned random forest classifier, we compute the similarity between a pair of instances by sending them down the decision trees in the random forest and count the number of times they reach the same leaf, normalized by the total number of trees. This will result in a value between 0 and 1, with 0 for no similarity and 1 for maximum similarity. Note that random forest has previously been successfully applied to estimating similarities between unsupervised objects [20]. In that work, a random forest classifier is built to distinguish the observed data from synthetically generated data, whereas our work

1. Note that the choice to use entropy here is not critical. Other uncertainty or certainty (to be minimized) measures could potentially be used as well. For example, one could measure the certainty of a point by the maximum probability, or by the difference between the largest and the second largest probabilities.

builds the random forest classifier to distinguish the different clusters. Because the clusters are identified by applying constraint-based clustering to data using the constraint set \mathcal{C} , thus the resulting proximities can be also viewed as a supervised similarity measure learned indirectly using the constraint set \mathcal{C} .

Estimating neighborhood probability. Given the similarity matrix M generated by previous steps, let $M(\mathbf{x}_i, \mathbf{x}_j)$ denote the similarity between instance \mathbf{x}_i and instance \mathbf{x}_j . For any unconstrained data point \mathbf{x} , we assume that its probability of belonging to a neighborhood N_i to be proportional to the average² similarity between \mathbf{x} and the instances in N_i . More formally, we estimate the probability of an instance \mathbf{x} belonging to neighborhood N_i as

$$p(\mathbf{x} \in N_i) = \frac{\frac{1}{|N_i|} \sum_{\mathbf{x}_j \in N_i} M(\mathbf{x}, \mathbf{x}_j)}{\sum_{p=1}^l \frac{1}{|N_p|} \sum_{\mathbf{x}_j \in N_p} M(\mathbf{x}, \mathbf{x}_j)}, \quad (1)$$

where $|N_i|$ indicates the number of instances in neighborhood N_i , and l is the total number of existing neighborhoods. Note that in early stages of our algorithm, when all neighborhoods are small, it is possible for an unconstrained data point \mathbf{x} to have zero average similarity with every neighborhood. In such cases, we assign equal probabilities to all neighborhoods for \mathbf{x} . This will essentially treat instance \mathbf{x} as highly uncertain, making it a good candidate to be selected by our algorithm. This behavior is reasonable because it will encourage the discovery of more neighborhoods in early stages.

Finally, we measure the uncertainty of an instance by the entropy of its neighborhood membership, which we denote $H(\mathcal{N} | \mathbf{x})$:

$$H(\mathcal{N} | \mathbf{x}) = - \sum_{i=1}^l p(\mathbf{x} \in N_i) \log_2 p(\mathbf{x} \in N_i), \quad (2)$$

where l is the total number of existing neighborhoods.

3.3.2 Normalizing Uncertainty with Expected Cost

Note that we query a selected instance against the existing neighborhoods to determine to which neighborhood it belongs. Given a selected data instance, it may take multiple pairwise queries to decide its neighborhood. In our selection criterion, we should take this into consideration. In particular, we can consider the number of queries required to reach a must-link as the cost associated with each data instance. To define and quantify this cost more precisely, let us take a closer look at the querying process.

Given a selected instance \mathbf{x} , and the probabilities of it belonging to different neighborhoods, which neighborhood should we query against first? Assume the estimated probabilities $p(\mathbf{x} \in N_i)$ are accurate for all $\mathbf{x} \in \mathcal{D}$ and $N_i \in \mathcal{N}$, we should always start by querying \mathbf{x} against the neighborhood that has the highest probability of containing \mathbf{x} 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 \mathbf{x} . This procedure is repeated until a must-link constraint is returned or we have a cannot-link constraint against all neighborhoods, at which point a new neighborhood will be created using \mathbf{x} .

Let $q(\mathbf{x})$ denote the random variable of the total number of queries that we need to determine the neighborhood membership of \mathbf{x} . Assuming that the neighborhoods are ranked based on their probability of containing \mathbf{x} in descending order, i.e., $p(\mathbf{x} \in N_1) \geq p(\mathbf{x} \in N_2) \geq \dots \geq p(\mathbf{x} \in N_l)$, where l is the total number of existing neighborhoods, it is straightforward to show that $p(q(\mathbf{x}) = i) = p(\mathbf{x} \in N_i)$. The expectation $\mathbb{E}[q(\mathbf{x})]$ is, thus, computed by the following equation:

$$\mathbb{E}[q(\mathbf{x})] = \sum_{i=1}^l i * p(\mathbf{x} \in N_i), \quad (3)$$

where l is the total number of existing neighborhoods.

If we consider $H(\mathcal{N} | \mathbf{x})$, the entropy of the neighborhood membership of \mathbf{x} (defined by (2)), as the amount of information we gain by querying about data instance \mathbf{x} , $\mathbb{E}[q(\mathbf{x})]$ is simply the cost for obtaining this information as measured by the number of queries consumed. Ideally, we would like to maximize the gain of information, i.e., $H(\mathcal{N} | \mathbf{x})$, and at the same time minimize the cost, i.e., $\mathbb{E}[q(\mathbf{x})]$. However, these two objectives are at odds and we trade off them by selecting the instance that maximizes the ratio between them,

$$\mathbf{x}^* = \underset{\mathbf{x} \in \mathcal{U}}{\operatorname{argmax}} \frac{H(\mathcal{N} | \mathbf{x})}{\mathbb{E}[q(\mathbf{x})]}, \quad (4)$$

where \mathcal{U} denotes the set of unconstrained instances (i.e., the set of points that do not belong to any neighborhood). This criterion can be interpreted as selecting the instance that has the highest rate of information per query.

So far, we have described our proposed method for selecting the most informative instance to query. We summarize this selection algorithm in Algorithm 2. This completes the description of our overall algorithm which is summarized in Algorithm 1.

Algorithm 2. MostInformative($\mathcal{D}, \pi, \mathcal{N}$).

Input: A set of data instances \mathcal{D} ; the cluster assignments π ; A set of neighborhoods $\mathcal{N} = \bigcup_{i=1}^l N_i$;

Output: The most informative data point \mathbf{x}^* ;

- 1: Learn a random forest classifier on $\mathcal{D}' = \{\mathbf{x}_i, \pi(\mathbf{x}_i)\}_{i=1}^n$, and compute the similarity matrix M ;
- 2: **for** each $\mathbf{x} \in \mathcal{D}$, and $\mathbf{x} \notin \bigcup_{i=1}^l N_i$ **do**
- 3: **for** $i = 1$ to l **do**
- 4: Compute $p(\mathbf{x} \in N_i)$ using (1);
- 5: **end for**
- 6: Compute $H(\mathcal{N} | \mathbf{x})$ using (2);
- 7: Compute $\mathbb{E}[q(\mathbf{x})]$ using (3);
- 8: **end for**
- 9: **Return**
 $\mathbf{x}^* = \underset{\mathbf{x} \in \mathcal{U}}{\operatorname{argmax}} \frac{H(\mathcal{N} | \mathbf{x})}{\mathbb{E}[q(\mathbf{x})]} \text{ where } \mathcal{U} = \mathcal{D} \setminus \bigcup_{i=1}^l N_i$

2. Note that instead of average, one can also consider minimum or maximum here. We have observed empirically that minimum and maximum tend to be more sensitive to outliers and lead to less robust solutions.

3.4 Runtime Analysis

In this section, we analyze the runtime of our proposed algorithm. In particular, we will focus on Algorithm 2 since it is the core part of our active learning algorithm.

In line 1, we build a random forest classifier, whose running time is $O(N_T n \log n)$,³ where N_T is the number of decision trees in RF and n is the number of instances in the data [19]. Once the RF classifier is built, constructing a full similarity matrix will take $O(n^2)$. However, we do not need to estimate the full similarity matrix, instead we only need to estimate a subset of the matrix of size $m \times n$, where m is the total number of points in the neighborhoods. As a result, the total runtime of line 1 is $O(N_T n \log n + nm)$.

The for-loop in line 2 is executed at most $O(n)$ times, and the runtime of each execution is $O(m + c)$, where m is the total number of “labeled” instances, i.e., the instances that are assigned to a known neighborhood. We can generally bound both m and c by T , the total number of queries allowed to ask, because it takes at least one query to assign an instance to a neighborhood and T is generally greater than c . Therefore, we can bound the total runtime between line 2-8 by $O(nT)$.

Putting it together, the total runtime of Algorithm 2 is $O(N_T n \log n + nT)$. Empirically, with a nonoptimized Matlab implementation on an Intel 8-Core i7-2600 CPU at 3.40 GHz, the average time to select an instance to query for the largest data set we experimented with (i.e., Digits-389 with 3,165 instances) is approximately 0.02 second (using random forest of 50 decision trees). For significantly larger data sets with millions of instances and thousands of features, additional strategies could be applied to scale up our method. For example, the random forest learning step can be easily parallelized to increase the efficiency. Another possibility would be to develop and apply an incremental semi-supervised clustering method that updates the clustering solution incrementally when new constraints are incorporated.

4 EXPERIMENTS

In this section, we empirically evaluate the performance of our proposed method in comparison with current state-of-the-art methods. Below we will first explain our experimental setup.

4.1 Experimental Setup

4.1.1 Data Sets

In our experiments, we use eight benchmark UCI data sets [21] that have been used in previous studies on constraint-based clustering [1], [4]. Our data sets include breast [22], pen-based recognition of handwritten digits (3, 8, 9), ecoli, glass identification, statlog-heart, parkinsons [23], statlog-image segmentation, and wine. For the ecoli data set, we removed the smallest three classes, which only contain 2, 2, and, 5 instances, respectively. The characteristics of the eight data sets are shown in Table 1.

3. Strictly speaking it should be $O(N_T d n \log n)$, where d corresponds to the size of the random feature subset that is considered for selecting each test. Here, we drop d because it is generally much smaller than n and can be viewed as a constant factor in this case.

TABLE 1
Characteristics of the Data Sets

Datasets	# of Classes	# of Features	# of Examples
Breast	2	9	683
Digits-389	3	16	3165
Ecoli	5	7	327
Glass	6	9	214
Heart	2	13	270
Parkinsons	2	22	195
Segment	7	19	2310
Wine	3	13	178

4.1.2 Experimental Setting

Our active learning framework assumes the availability of a constraint-based clustering algorithm. For this purpose, we use the well-known MPCKMeans [3] algorithm, as implemented in the WekaUT package [24]. We set the maximum number of iterations of MPCKmeans to 200, and used default values for other parameters. Note that the choice of this algorithm is not critical and our method can be used with any constraint-based clustering algorithm.

When evaluating the performance of a particular method on a given data set \mathcal{D} , we apply it to select up to 150 pairwise queries, starting from no constraint at all. The queries are answered based on the ground-truth class label for the data set. MPCKmeans is then applied to the data with the resulting constraints (and their transitive closures). To account for the randomness in both active learning and MPCKmeans, we repeat this process for 50 independent runs and report the average performance using evaluation criteria described below.

4.1.3 Evaluation Criteria

Two evaluation criteria are used in our experiments. First, we use normalized mutual information (NMI) to evaluate the clustering assignments against the ground-truth class labels [25]. NMI considers both the class label and clustering assignment as random variables, and measures the mutual information between the two random variables, and normalizes it to a zero-to-one range. In general, let C be the random variable representing the cluster assignments of instances, and K be the random variable representing the class labels of the instances, the NMI is computed by the following equation:

$$NMI = \frac{2I(C; K)}{H(C) + H(K)},$$

where $I(X; Y) = H(X) - H(X | Y)$ is the mutual information between random variables X and Y . $H(X)$ is the entropy of X , and $H(X | Y)$ is the conditional entropy X given Y .

Second, we consider F-measure as another criterion 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 [1]. F-measure is defined as the harmonic mean of precision and recall, which are computed by the following equations:

$$\begin{aligned}
\text{Precision} &= \frac{\# \text{PairsCorrectlyPredictedInSameCluster}}{\# \text{TotalPairsPredictedInSameCluster}}, \\
\text{Recall} &= \frac{\# \text{PairsCorrectlyPredictedInSameCluster}}{\# \text{TotalPairsActuallyInSameCluster}}, \\
\text{F-measure} &= \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.
\end{aligned}$$

4.1.4 Baseline Methods

To demonstrate the effectiveness of the proposed method, we first compare its performance to a set of competing methods, including a random policy, the Min-Max approach introduced by Mallapragada et al. [13], and a variant of Huang’s method [12] to make it applicable to nondocument data types. Below we briefly explain these baseline methods.

Random. This policy selects random pairwise queries that are not included in or implied by the current set of constraints \mathcal{C} . It is not a neighborhood-based approach, and is a commonly used baseline for active learning studies.

Min-Max. As reviewed in Section 2, Min-Max is a neighborhood-based approach that works in two phases. The first phase, Explore, builds c disjoint neighborhoods using farthest-first traversal, where c is the total number of clusters. The second phase, Min-Max, incrementally expands the neighborhoods by selecting a point to query using a distance-based Min-Max criterion [13].

Huang’s method. As noted in Section 2, the language-model-based method by Huang and Lam is only applicable to document clustering because it assumes a specific language model for each cluster. This model is used to estimate the probability of each document belonging to different clusters. In our experiments, to apply the underlying active learning method to general-type data, we replace the language with a discriminative approach for estimating the probabilities. In particular, we train a random forest classifier using the cluster labels as classes. We then estimate for each data point x its probability of belonging to different clusters using the out-of-bag probabilistic prediction for x . That is, we consider all decision trees in the random forest that were trained without using the data point x , and use them to estimate the probability of x belonging to different clusters.

4.2 Experimental Results

This section presents the experiment results, which compare our proposed method to the baseline methods. In the remaining discussion, we will refer to our method as the normalized point-based uncertainty (NPU) method.

4.2.1 Evaluation Based on Clustering Performance

The NMI values of NPU and the baseline methods are shown in Fig. 2. The x -axis indicates the total number of pairwise queries and the y -axis shows the resulting clustering performance (as measured by NMI) by running MPCKmeans with the constraints returned from the queries (and their transitive closures). As mentioned previously, each curve shows the average performance of a method across 50 independent random runs. The error bar on each data point indicates the confidence interval (t -test at 95 percent significance level). Note that we use up to 150 queries for all but two data sets, namely breast and wine. For these two data

sets, NPU converges before using up 150 queries, therefore we show the results up to 100 queries.

From Fig. 2, we can see that the constraints selected by NPU generally leads to clustering results that are more consistent with the underlying class labels, as can be seen by the dominating curve of NPU compared to other baseline curves. It is interesting to note that random actually degrades the performance in some data sets as we include more constraints, namely the breast, heart, and wine data sets. Previous studies on semi-supervised clustering [4], [5], [26] have reported similar results, where randomly selected constraints actually degrades the clustering performance for some data sets. This further demonstrates the importance of selecting the right set of constraints. In comparison, Min-Max and Huang’s methods are generally able to improve the performance consistently as we increase the number of queries, but their performance are dominated by NPU in most cases.

We also note that in the early stages, the performance of the three nonrandom methods are fairly close. As we increase the number of queries, the performance advantage of our method becomes more and more pronounced. This is expected because our method make more explicit usage of the current clustering solution when selecting the queries. As we increase the number of queries, the clustering solution will become better and better, leading to more pronounced performance advantage of our method.

4.2.2 Evaluation Based on Pairwise Relationship

F-measure focuses on how accurately we can predict the pairwise relationship between any pair of instances. In Table 2, we show the F-measure values achieved by different methods with query sizes of 20, 40, 60, 80, and 100. For each query size, we compare different methods against each other using paired t -test at 95 percent significance level and the best performing method(s) are then highlighted in boldface. Finally, Table 3 provides a summary of the win/tie/loss counts of the proposed method versus the other methods.

This set of results are very similar to what we observe when evaluating using NMI. When using only 20 queries, the performance of the nonrandom methods often do not demonstrate statistically significant difference. However, as we increase the number of queries, our method begins to dominate all other methods.

4.2.3 Further Analysis of Results

Below we provide some more in-depth analysis of the performance to understand what are the key factors contributing to the performance advantage of our method.

With or without explore. In the Min-Max method, the first phase is Explore, which uses furthest-first traversal to find at least one example from each neighborhood to obtain a good “skeleton” of the clusters. Basu et al. [1] showed that given a set of c disjoint balls (clusters) of uneven sizes, Explore is guaranteed to get at least one example from each cluster with a small number of queries. Our method does not use a separate Explore stage to deliberately build c neighborhoods. Does this help or hurt our performance?

To answer this question, we consider a two-phase variant of NPU, which performs Explore first (as used by Min-Max), followed by the NPU selection criterion. We

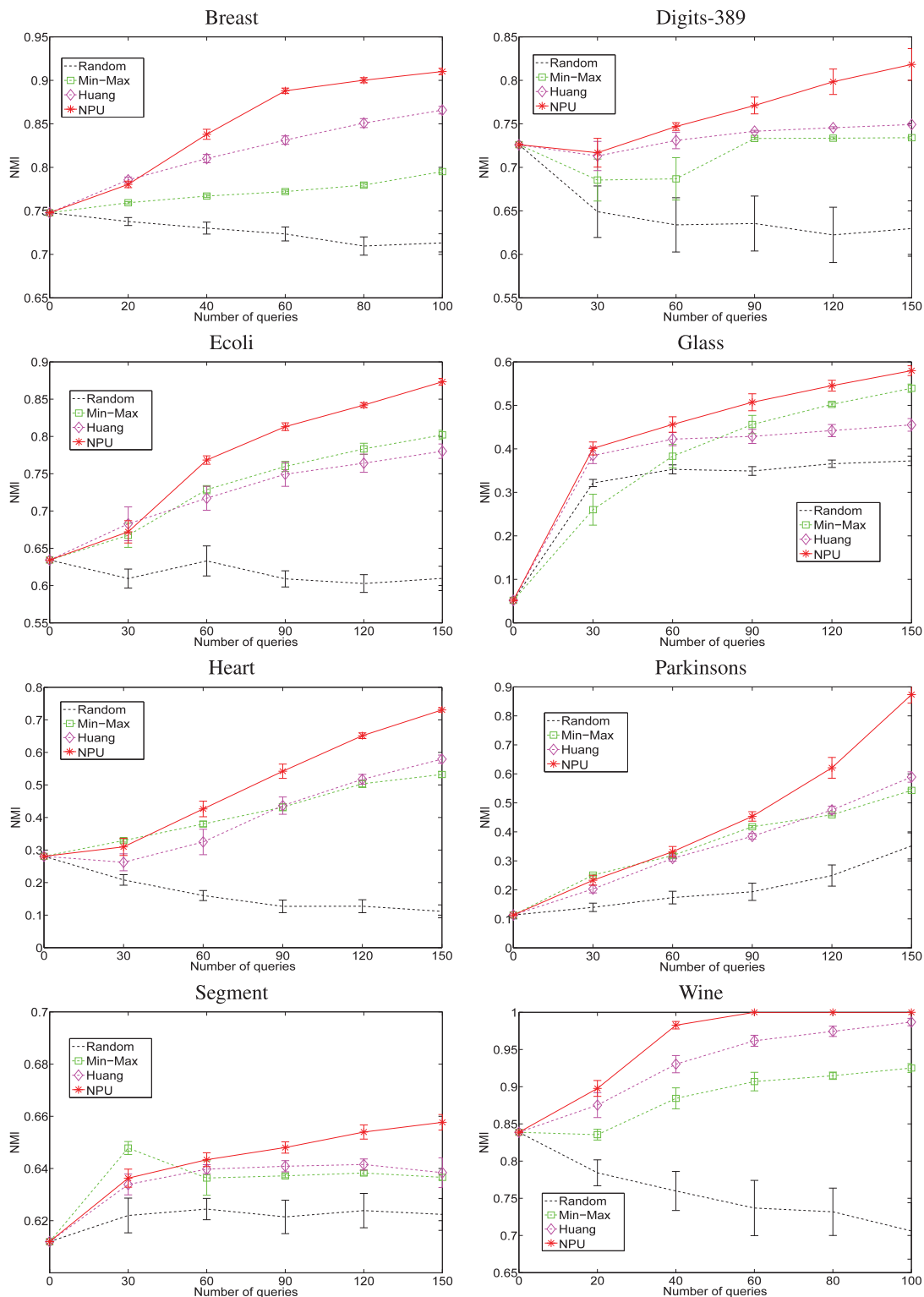


Fig. 2. The NMI values of different methods on eight data sets as a function of the number of pairwise queries (mean and the confidence interval of t -test at 95 percent significance level).

examine this method (referred to as E & NPU) and compare it with NPU. The results are shown in Fig. 3, where each point in the figure represents a comparison between the two methods using NMI on one of the eight data sets, and with one of five different query sizes including 20, 40, 60, 80, and 100. The x -axis shows the performance of E & NPU, and the y -axis shows the performance of NPU. We observe that

most points lie above the diagonal line, suggesting that NPU is superior to E & NPU. This indicates that eliminating the Explore stage actually allows more efficient use of the queries. Note that the difference is most pronounced in the areas of low NMI values, indicating they happen in the early stages (small query sizes). This is understandable because the two variants differ only by the initial stages.

TABLE 2
Comparison on F-Measure (Mean \pm std)

Data	Algorithm	Number of queries				
		20	40	60	80	100
Breast	Random	0.927 \pm 0.005	0.924 \pm 0.008	0.921 \pm 0.010	0.916 \pm 0.013	0.918 \pm 0.012
	Min-Max	0.934 \pm 0.001	0.937 \pm 0.002	0.939 \pm 0.002	0.941 \pm 0.002	0.946 \pm 0.002
	Huang	0.941 \pm 0.004	0.951 \pm 0.005	0.957 \pm 0.004	0.963 \pm 0.004	0.967 \pm 0.004
	NPU	0.943 \pm 0.003	0.959 \pm 0.005	0.972 \pm 0.003	0.976 \pm 0.003	0.978 \pm 0.003
Digits-389	Random	0.762 \pm 0.104	0.774 \pm 0.100	0.749 \pm 0.108	0.752 \pm 0.107	0.752 \pm 0.107
	Min-Max	0.805 \pm 0.080	0.788 \pm 0.093	0.797 \pm 0.087	0.842 \pm 0.001	0.842 \pm 0.002
	Huang	0.814 \pm 0.072	0.826 \pm 0.061	0.842 \pm 0.034	0.851 \pm 0.002	0.853 \pm 0.003
	NPU	0.808 \pm 0.077	0.848 \pm 0.011	0.857 \pm 0.013	0.870 \pm 0.023	0.883 \pm 0.032
Ecoli	Random	0.642 \pm 0.076	0.628 \pm 0.050	0.700 \pm 0.092	0.653 \pm 0.064	0.659 \pm 0.076
	Min-Max	0.648 \pm 0.043	0.779 \pm 0.060	0.836 \pm 0.009	0.851 \pm 0.008	0.858 \pm 0.010
	Huang	0.687 \pm 0.058	0.762 \pm 0.069	0.801 \pm 0.047	0.833 \pm 0.036	0.829 \pm 0.046
	NPU	0.673 \pm 0.032	0.798 \pm 0.048	0.858 \pm 0.007	0.879 \pm 0.008	0.900 \pm 0.007
Glass	Random	0.440 \pm 0.051	0.403 \pm 0.033	0.410 \pm 0.037	0.410 \pm 0.032	0.413 \pm 0.034
	Min-Max	0.432 \pm 0.034	0.418 \pm 0.067	0.463 \pm 0.059	0.484 \pm 0.060	0.493 \pm 0.040
	Huang	0.480 \pm 0.039	0.481 \pm 0.042	0.476 \pm 0.043	0.474 \pm 0.038	0.473 \pm 0.039
	NPU	0.493 \pm 0.036	0.492 \pm 0.045	0.481 \pm 0.056	0.496 \pm 0.056	0.495 \pm 0.043
Heart	Random	0.659 \pm 0.033	0.636 \pm 0.035	0.612 \pm 0.036	0.598 \pm 0.043	0.587 \pm 0.041
	Min-Max	0.700 \pm 0.012	0.726 \pm 0.013	0.743 \pm 0.015	0.760 \pm 0.017	0.790 \pm 0.018
	Huang	0.680 \pm 0.042	0.682 \pm 0.064	0.709 \pm 0.080	0.744 \pm 0.076	0.789 \pm 0.045
	NPU	0.682 \pm 0.046	0.725 \pm 0.049	0.766 \pm 0.047	0.812 \pm 0.016	0.845 \pm 0.014
Parkinsons	Random	0.594 \pm 0.023	0.607 \pm 0.023	0.635 \pm 0.032	0.657 \pm 0.040	0.682 \pm 0.059
	Min-Max	0.593 \pm 0.005	0.615 \pm 0.015	0.666 \pm 0.007	0.705 \pm 0.010	0.747 \pm 0.008
	Huang	0.593 \pm 0.015	0.605 \pm 0.012	0.652 \pm 0.024	0.694 \pm 0.023	0.736 \pm 0.033
	NPU	0.597 \pm 0.012	0.637 \pm 0.031	0.695 \pm 0.036	0.759 \pm 0.039	0.814 \pm 0.068
Segment	Random	0.546 \pm 0.028	0.553 \pm 0.030	0.552 \pm 0.030	0.548 \pm 0.035	0.549 \pm 0.036
	Min-Max	0.571 \pm 0.009	0.582 \pm 0.022	0.569 \pm 0.022	0.566 \pm 0.015	0.569 \pm 0.004
	Huang	0.567 \pm 0.017	0.576 \pm 0.015	0.577 \pm 0.013	0.573 \pm 0.024	0.575 \pm 0.012
	NPU	0.565 \pm 0.027	0.579 \pm 0.017	0.581 \pm 0.014	0.585 \pm 0.013	0.587 \pm 0.014
Wine	Random	0.871 \pm 0.044	0.853 \pm 0.073	0.836 \pm 0.102	0.843 \pm 0.082	0.827 \pm 0.095
	Min-Max	0.909 \pm 0.018	0.935 \pm 0.037	0.945 \pm 0.035	0.953 \pm 0.010	0.959 \pm 0.012
	Huang	0.931 \pm 0.042	0.964 \pm 0.031	0.982 \pm 0.012	0.988 \pm 0.011	0.994 \pm 0.007
	NPU	0.945 \pm 0.025	0.992 \pm 0.008	1.000 \pm 0.000	1.000 \pm 0.000	1.000 \pm 0.000

The best performance and its comparable performances based on paired t -tests at 95 percent significance level are highlighted in boldface.

Note that NPU can discover c neighborhoods by itself, where c is the number of classes. Could our method be more efficient in discovering all of the neighborhoods than Explore? We record the number of queries used before finding all c neighborhoods for each data set using Explore and NPU. Table 4 shows the average number of queries required by each method to find c neighborhoods for each data set. The results show that despite the theoretical guarantees of Explore, NPU is empirically more efficient in finding all neighborhoods for multiclass data sets (three or more classes). This is possibly due to the fact that real data sets may not have ball-shaped clusters, violating the condition that is assumed by Explore for its theoretical guarantees.

Can our method still outperform its competitors, if we put them on equal footing in terms of the use of Explore? Figs. 4 and 5 represent the comparison between our method

and Min-Max and Huang's method when each method is applied with Explore as the common initial stage.

From this comparison, we observe that majority of the points lie either on or above the diagonal lines. It is worth noting that many of the points that are on or close to the diagonal line have low NMI values, suggesting that they correspond to comparisons at the early stages of the active learning process. This is precisely what we expect because

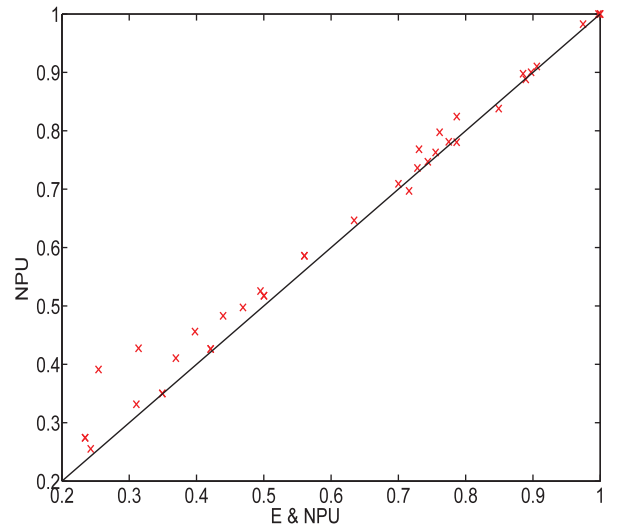


Fig. 3. Comparing the performance (NMI) of NPU with and without Explore.

TABLE 3
Win/Tie/Loss Counts of NPU versus the Other Methods with Varied Numbers of Queries Based on F-Measure

Algorithms	Number of queries					In All
	20	40	60	80	100	
Random	5/3/0	8/0/0	8/0/0	8/0/0	8/0/0	37/3/0
Min-Max	4/3/1	6/2/0	8/0/0	8/0/0	8/0/0	34/5/1
Huang	2/5/1	6/2/0	8/0/0	8/0/0	8/0/0	30/9/1
In All	11/11/2	20/4/0	22/2/0	24/0/0	24/0/0	101/17/2

TABLE 4
Number of Queries to Discover all c Neighborhoods

Data (# of Classes)	Methods	
	Explore	NPU
Breast (2)	1.42	2.68
Digits-389 (3)	14.22	5.50
Ecoli (5)	76.12	50.34
Glass (6)	98.90	73.94
Heart (2)	2.76	2.08
Parkinsons (2)	1.82	4.22
Segment (7)	102.98	36.92
Wine (3)	9.40	6.14

in this comparison we use the same Explore phase to initialize all three methods, thus they should behave exactly the same for the early stages. For higher performance areas (presumably when using more queries), we observe that more points lie above the diagonal line. This suggests that after eliminating the compounding factor of the Explore phase, our NPU method remains superior to the Min-Max and Huang's method.

Point-based versus pairwise uncertainty. Huang's method is our closest competitor. A key difference between the approach we propose and Huang's method is that our method considers point-based uncertainty whereas Huang's method is based on pairwise uncertainty. In Section 3, we identify a potential issue with Huang's method. That is, it only considers the information content of the first query, even though multiple queries may be needed to identify the neighborhood of a selected instance. Here, we would like to make a direct comparison between these two approaches. As such, we consider our method without the normalization step and compare it with Huang's method in Fig. 6. The x -axis shows the performance of Huang's method, and the y -axis shows the performance of our method without normalization, i.e., directly using (2) as the selection criterion (referred to as unnormalized point-based uncertainty), both measured in NMI. Each point in the figure corresponds to a particular data set with a particular query size. As we can see from the figure, UPU generally outperforms Huang's method,

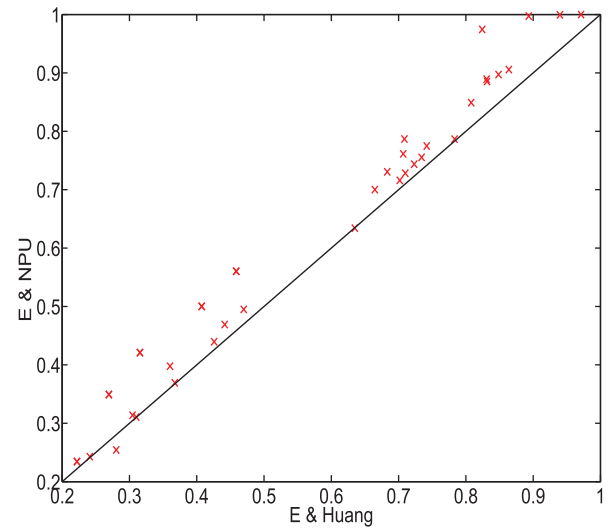


Fig. 5. Comparing performance (NMI) of E & Huang with E & NPU.

providing strong evidence that point-based uncertainty leads to better and more informative queries.

Effect of normalization. Finally, we would like to examine the effect of normalization. In Fig. 7, we show the performance comparison between the two variants of our method with or without the normalization step, namely NPU and UPU. The x -axis shows the performance of the unnormalized version (UPU), whereas the y -axis shows the performance of the normalized version (NPU). From the figure, we can see that most of the points lie either above or close to the diagonal line, suggesting that the normalization step leads to some mild improvement in performance. To understand the significance of the differences, we performed paired student t -tests at 95 percent significance level. The results show that for four of the eight data sets (Digits, Ecoli, Parkinsons, and Segment), the normalization step was able to improve the performance in a statistically significant way. On the other hand, it never hurts the performance significantly. This suggests that it is advisable to include the normalization

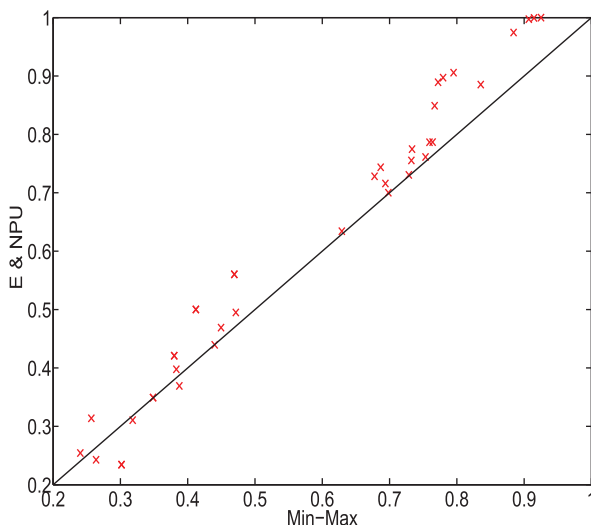


Fig. 4. Comparing performance (NMI) of Min-Max with E & NPU.

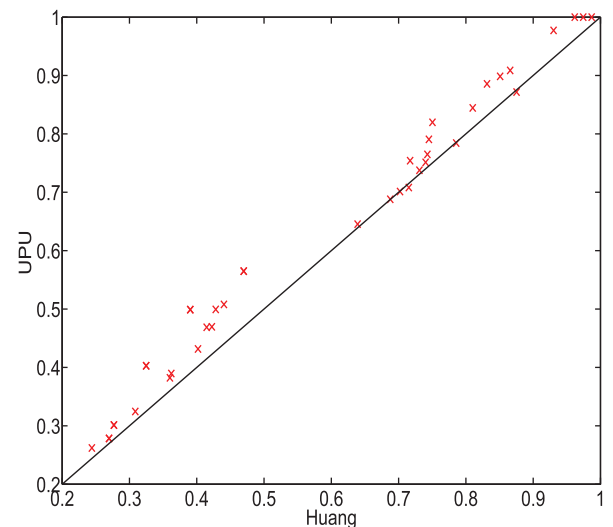


Fig. 6. Comparing performance (NMI) of Huang with unnormalized point-based uncertainty (UPU).

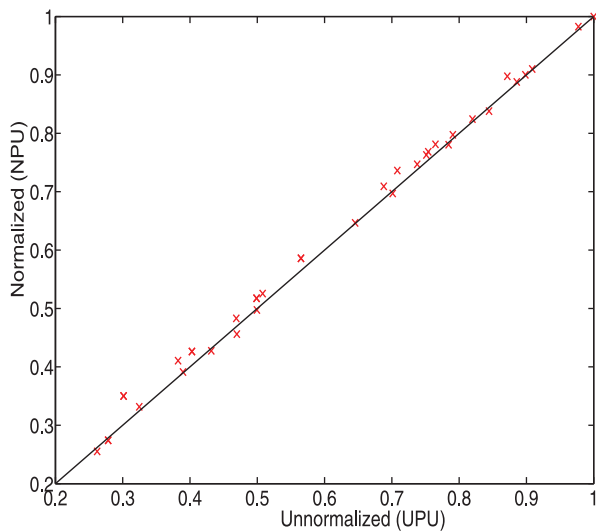


Fig. 7. Comparing performance (NMI) of the point-based uncertainty method with and without normalization. The y -axis shows the performance of the normalized version (NPU) whereas the x -axis shows the performance of the unnormalized version (UPU).

step since it may provide additional benefit for some data sets and generally does not hurt the performance.

5 CONCLUSION AND FUTURE WORK

In this paper, we study an iterative active learning framework to select pairwise constraints for semi-supervised clustering and propose a novel method for selecting the most informative queries.

Our method takes a neighborhood-based approach, and incrementally expands the neighborhoods by posing pairwise queries. We devise an instance-based selection criterion that identifies in each iteration the best instance to include into the existing neighborhoods. The selection criterion trades off two factors, the information content of the instance, which is measured by the uncertainty about which neighborhood the instance belongs to; and the cost of acquiring this information, which is measured by the expected number of queries required to determine its neighborhood.

We empirically evaluate the proposed method on the eight benchmark data sets against a number of competing methods. The evaluation results indicate that our method achieves consistent and substantial improvements over its competitors.

There are a number of interesting directions to extend our work. The iterative framework requires repeated reclustering of the data with an incrementally growing constraint set. This can be computationally demanding for large data sets. To address this problem, it would be interesting to consider an incremental semi-supervised clustering method that updates the existing clustering solution based on the neighborhood assignment for the new point. An alternative way to lower the computational cost is to reduce the number of iterations by applying a batch approach that selects a set of points to query in each iteration. A naive batch active learning approach would be to select the top k points that have the highest normalized uncertainty to query their neighborhoods. However, such a

strategy will typically select highly redundant points. Designing a successful batch method requires carefully trading off the value (normalized uncertainty) of the selected points and the diversity among them—a direction that we plan to pursue for future work.

REFERENCES

- [1] S. Basu, A. Banerjee, and R. Mooney, "Active Semi-Supervision for Pairwise Constrained Clustering," *Proc. SIAM Int'l Conf. Data Mining*, pp. 333-344, 2004.
- [2] S. Basu, I. Davidson, and K. Wagstaff, *Constrained Clustering: Advances in Algorithms, Theory, and Applications*. Chapman & Hall, 2008.
- [3] M. Bilenko, S. Basu, and R. Mooney, "Integrating Constraints and Metric Learning in Semi-Supervised Clustering," *Proc. Int'l Conf. Machine Learning*, pp. 11-18, 2004.
- [4] I. Davidson, K. Wagstaff, and S. Basu, "Measuring Constraint-Set Utility for Partitional Clustering Algorithms," *Proc. 10th European Conf. Principle and Practice of Knowledge Discovery in Databases*, pp. 115-126, 2006.
- [5] D. Greene and P. Cunningham, "Constraint Selection by Committee: An Ensemble Approach to Identifying Informative Constraints for Semi-Supervised Clustering," *Proc. 18th European Conf. Machine Learning*, pp. 140-151, 2007.
- [6] D. Cohn, Z. Ghahramani, and M. Jordan, "Active Learning with Statistical Models," *J. Artificial Intelligence Research*, vol. 4, pp. 129-145, 1996.
- [7] Y. Guo and D. Schuurmans, "Discriminative Batch Mode Active Learning," *Proc. Advances in Neural Information Processing Systems*, pp. 593-600, 2008.
- [8] S. Hoi, R. Jin, J. Zhu, and M. Lyu, "Batch Mode Active Learning and Its Application to Medical Image Classification," *Proc. 23rd Int'l Conf. Machine learning*, pp. 417-424, 2006.
- [9] S. Hoi, R. Jin, J. Zhu, and M. Lyu, "Semi-Supervised SVM Batch Mode Active Learning for Image Retrieval," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 1-7, 2008.
- [10] S. Huang, R. Jin, and Z. Zhou, "Active Learning by Querying Informative and Representative Examples," *Proc. Advances in Neural Information Processing Systems*, pp. 892-900, 2010.
- [11] B. Settles, "Active Learning Literature Survey," technical report, 2010.
- [12] R. Huang and W. Lam, "Semi-Supervised Document Clustering via Active Learning with Pairwise Constraints," *Proc. Int'l Conf. Data Mining*, pp. 517-522, 2007.
- [13] P. Mallapragada, R. Jin, and A. Jain, "Active Query Selection for Semi-Supervised Clustering," *Proc. Int'l Conf. Pattern Recognition*, pp. 1-4, 2008.
- [14] Q. Xu, M. Desjardins, and K. Wagstaff, "Active Constrained Clustering by Examining Spectral Eigenvectors," *Proc. Eighth Int'l Conf. Discovery Science*, pp. 294-307, 2005.
- [15] L. Breiman, "Random Forests," *Machine learning*, vol. 45, no. 1, pp. 5-32, 2001.
- [16] M. Al-Razgan and C. Domeniconi, "Clustering Ensembles with Active Constraints," *Applications of Supervised and Unsupervised Ensemble Methods*, pp. 175-189, Springer, 2009.
- [17] O. Shamir and N. Tishby, "Spectral Clustering on a Budget," *J. Machine Learning Research - Proc. Track*, vol. 15, pp. 661-669, 2011.
- [18] K. Voevodski, M. Balcan, H. Röglin, S. Teng, and Y. Xia, "Active Clustering of Biological Sequences," *J. Machine Learning Research*, vol. 13, pp. 203-225, 2012.
- [19] L. Breiman, "RF/Tools: A Class of Two-Eyed Algorithms," *Proc. SIAM Workshop, Statistics Dept.*, 2003.
- [20] T. Shi and S. Horvath, "Unsupervised Learning with Random Forest Predictors," *J. Computational and Graphical Statistics*, vol. 15, pp. 118-138, 2006.
- [21] A. Frank and A. Asuncion, "UCI Machine Learning Repository," <http://archive.ics.uci.edu/ml>, 2010.
- [22] O. Mangasarian, W. Street, and W. Wolberg, "Breast Cancer Diagnosis and Prognosis via Linear Programming," *Operations Research*, vol. 43, no. 4, pp. 570-577, 1995.
- [23] M. Little, P. McSharry, S. Roberts, D. Costello, and I. Moroz, "Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection," *BioMedical Eng. OnLine*, vol. 6, no. 1, p. 23, 2007.

- [24] B. Sugato, "Wekaut, A Modified Version of Weka." <http://www.cs.utexas.edu/users/ml/risc/code/>, 2011.
- [25] L. Kuncheva and S. Hadjitodorov, "Using Diversity in Cluster Ensembles," *Proc. Int'l Conf. Systems, Man and Cybernetics*, vol. 2, pp. 1214-1219, 2004.
- [26] I. Davidson, S. Ravi, and L. Shamis, "A SAT-Based Framework for Efficient Constrained Clustering," *Proc. SIAM Int'l Conf. Data Mining*, 2010.



Sicheng Xiong received the BEng degree in computer science from the Beijing University of Posts and Telecommunications, Beijing, China, in 2009. He is currently working toward the graduate degree in computer science at Oregon State University, Corvallis, Oregon. His research interests are in machine learning, focusing on active learning, semi-supervised clustering, and distance metric learning.



Javad Azimi received the BSc degree in computer engineering from Shahed University, Tehran, Iran, in 2004, and the MSc degree in computer engineering (artificial intelligence) from the Iran University of Science and Technology, Tehran, Iran, in 2007. He received the PhD degree in computer science (machine learning) from Oregon State University, Corvallis, Oregon, in 2012. Since then, he has been with InsightsOne Inc! as a data scientist, where he develops algorithms to learn from big data in different applications including online advertising and health industry. His research interests include machine learning with special focus on Bayesian optimization, data clustering, active learning, and anomaly detection.



Xiaoli Z. Fern received the MS degree from Shanghai Jiao Tong university (SJTU), Shanghai, China, in 2000. She received the PhD degree in computer engineering from Purdue University, West Lafayette, IN, in 2005. She co-organized the first International Workshop on Discovering, Utilizing and Summarizing Multiple Clustering (MultiClust KDD 2010), and served as the publicity chair for International Conference on Machine Learning in 2007. She is currently an editorial board member of the *Machine Learning Journal* and serves regularly on the program committee for a number of top tier international conferences on machine learning and data mining. She is an assistant professor at the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR, since 2005. Her research interest is in the area of machine learning and data mining. She received a US National science Foundation (NSF) Career Award in 2011.

► **For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.**