

# Adaptive Clustering: Obtaining Better Clusters Using Feedback and Past Experience

Abraham Bagherjeiran, Christoph F. Eick, Chun-Sheng Chen, Ricardo Vilalta

abagherj@cs.uh.edu, ceick@cs.uh.edu, lyons19@cs.uh.edu, vilalta@cs.uh.edu

University of Houston

Department of Computer Science

Houston, Texas 77204-3010

## Abstract

*Adaptive clustering uses external feedback to improve cluster quality; past experience serves to speed up execution time. An adaptive clustering environment is proposed that uses  $Q$ -learning to learn the reward values of successive data clusterings. Adaptive clustering supports the reuse of clusterings by memorizing what worked well in the past. It has the capability of exploring multiple paths in parallel when searching for good clusters. In a case study, we apply adaptive clustering to instance-based learning relying on a distance function modification approach. A distance function adaptation scheme that uses external feedback is proposed and compared with other distance function learning approaches. Experimental results indicate that the use of adaptive clustering leads to significant improvements of instance-based learning techniques, such as  $k$ -nearest neighbor classifiers. Moreover, as a by-product a new instance-based learning technique is introduced that classifies examples by solely using cluster representatives; this technique shows high promise in our experimental evaluation.*

## 1. Introduction

A clustering algorithm finds groups of objects in a pre-defined attribute space. Since the objects have no known prior class membership, clustering is an unsupervised learning process that optimizes some explicit or implicit criterion inherent to the data such as the squared summed error [9]. The main criticism of the fixed-criterion approach is that it is excessively simplistic and does not accurately capture the user's understanding of the true nature of the data.

This paper introduces a novel data mining technique we term adaptive clustering. The central idea is to use reinforcement learning (RL) algorithms that can incorporate feedback and past experience to guide the search toward better clusters. We propose an adaptive clustering environment that modifies the weights of a distance function based on feedback. Our approach employs a traditional RL framework in which states consist of a set of cluster representa-

tives and distance function weights and actions modify the weights.

Since the application of a RL algorithm to clustering is unusual, it is worthwhile to discuss our reasons for this fusion of ideas. The goal of weight-based distance function learning is to find the weights that induce a clustering that best meets externally defined objectives. This is clearly an optimization problem, but we can extend the objective of adaptive clustering to support relearning. We envision the adaptive clustering environment as a continuously running process that can perform clustering tasks with dynamic data and potentially interactive evaluations. This is similar to the goal of RL algorithms, which are often applied in dynamic environments subject to changing external rewards [7, 10, 11, 13].

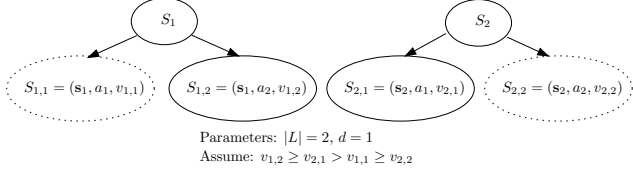
The paper is organized as follows. Section 2 introduces the idea of an environment for adaptive clustering. Section 3 introduces a framework that applies adaptive clustering to instance-based classification. Section 4 provides the results of an experimental evaluation of the proposed framework. Section 5 concludes the paper.

## 2. Adaptive Clustering

The adaptive clustering environment is a non-deterministic RL environment with states, actions, and rewards. A clustering algorithm forms a clustering on the data using its current distance function. The objective function evaluates the clustering and returns a reward. Given the current cluster representatives and reward, the RL algorithm selects a weight-changing action to apply to the distance function. The clustering algorithm forms a new clustering using the new distance function. This process repeats for a fixed number of iterations.

The state space consists of pairs of cluster representatives and distance function weights. The state vector contains the  $k$   $d$ -dimensional cluster representatives and the  $d$  attribute weights:

$$\begin{aligned} \mathbf{s} &= (\bar{r}_1, \dots, \bar{r}_k; \mathbf{w}) \\ &= (\bar{r}_{1_1}, \dots, \bar{r}_{1_d}; \dots; \bar{r}_{k_1}, \dots, \bar{r}_{k_d}; w_1, \dots, w_d) \end{aligned}$$



**Figure 1. Adaptive clustering search strategy.**

where  $r_{ij}$  is the  $j$ th coordinate value of the  $i$ th centroid.

Given a state, the learner can take an action that either increases or decreases the weight of a single attribute. Given the old weight vector  $w$  of length  $d$ , the action generates a new weight vector  $w'$ :

$$w' = \begin{cases} w_i \pm \Delta w_i & i = i^* \\ w_i & i \neq i^* \end{cases}$$

where  $i^*$  is the index of the attribute whose weight will be changed, the constant  $\Delta \in [.25, .5]$  is a randomly chosen percent change of the target weight, and the last equation is the renormalization of the weights. Given  $d$  attributes, the learner chooses one of  $2d$  actions that increases or decreases the weight of an attribute by adding or subtracting  $\Delta w_i$ . The distance function then becomes:

$$d(o_i, o_j) = \sum_{l=1}^d w_l |o_{il} - o_{jl}|$$

where  $o_i$  and  $o_j$  are  $d$ -dimensional objects and  $w = (w_1, \dots, w_d)$  is the new, normalized weight vector whose components are non-negative and  $\sum_{l=1}^d w_l = 1$ .

### 2.1. Search Strategy

The search process uses the Q-learning algorithm to estimate the value of applying a particular weight change action to a particular state. It maintains an open list  $L = \{S_1, \dots, S_{|L|}\}$  of search states where each state  $S_i = (s, a, v)$  consists of the state vector  $s$  from the environment, the action  $a$  to execute, and Q-value  $v$  of the state-action pair according to value iteration. Like RL algorithms, the search process executes the actions in the states with the highest Q-values. Like a local beam search algorithm and unlike most RL algorithms, it keeps the top  $|L| \geq 1$  most valuable states for execution in parallel [12].

As an example, Figure 1 further illustrates the search strategy. The figure assumes that the open list is of size 2. We assume that the open-list currently contains states  $S_1$  and  $S_2$  corresponding to state vectors  $s_1$  and  $s_2$ , respectively. Since the example has only one attribute, we either increase or decrease its weight. In search step 1, the algorithm creates a new open list that contains all single-action successors of the search states in its current open list. The bottom-row search states contain the current state vector and the action to execute. The algorithm looks up the Q-value of each of the  $2d$  bottom-row states and then keeps

the  $|L| = 2$  states having the largest values (shown in bold). The RL agent then applies the selected action in each of the bold search states. The prioritized sweeping algorithm next performs value iteration of the new reward values for all states in the new open list. The search continues until a fixed-number of steps have elapsed and returns the state with the highest reward value as its solution.

### 2.2. Complexity Concerns

From the previous discussion, it is obvious that the state space is very large. Adaptive clustering performs well in such large state spaces because it uses a variant of the Q-learning algorithm called prioritized sweeping [10]. This algorithm presumes that in a large state space, maintaining a value estimate for individual states makes learning unnecessarily slow. Its solution is to apply Q-learning to states based on a priority related to their value and recent frequency of activation.

The priority heuristic of the prioritized sweeping algorithm complements our beam search strategy well. The priority of a state's update depends on the absolute difference between the change in the expectation of reward given the obtained reward. For example, if the obtained reward causes the expectation to change from 5 to either 3 or 7, the algorithm will update states with the same priority. Having updated value information on paths that have significantly changed allows us to quickly concentrate on paths that suddenly appear better rather than worse. Thus, adaptive clustering can better concentrate on paths that suddenly appear more attractive without having to continually generate and evaluate novel states.

### 3. Experimental Evaluation

The experiments use  $k$ -means as the underlying clustering algorithm. It uses centroids as cluster representatives and creates clusters by assigning objects in a dataset to the nearest centroid [9]. The objective function  $R$  determines the worth of a particular clustering by computing the information gain of the objects in the clusters with respect to the original unclustered data [12]:

$$R(X) = R(\{c_1, \dots, c_k\}) = \frac{H(CL, O) - \sum_{i=1}^k \frac{|c_i|}{|O|} * H(CL, c_i)}{H(CL, O)}$$

$$\text{where } O = \bigcup_{i=1}^k c_i \text{ and } c_i \cap c_j \neq \emptyset$$

$$H(CL, V) = - \sum_{cl \in CL} p(cl, V) \log_2 p(cl, V)$$

where

$$p(cl, V) = \frac{|\{o \mid o \in V \wedge o \text{ is of class } cl \in CL\}|}{|\{o \mid o \in V\}|}$$

where  $O$  is the original dataset,  $X$  is a clustering on  $O$ ,  $c_i$  is one of the  $k$  mutually exclusive and exhaustive clusters of  $O$ , and  $H(CL, V)$  is the entropy with respect to the distribution of classes  $cl \in CL$  in  $V$ .  $H(CL, V)$  is computed by iterating over the individual classes in the data set. In summary, information gain computes the weighted average of the entropy with respect to the class distribution in each cluster.

The justification for the information gain objective function is obvious. A distance function that results in a clustering with high information gain places examples of the same class closer together than in the original dataset. In conjunction with a  $k$ -nearest neighbor classifier, we obtain high predictive accuracy simply because the learned distance function isolates local regions in the dataset with high class purity.

### 3.1. Procedure

The experiments compare classification accuracy with learned weights from the adaptive clustering algorithm and two related distance function learning algorithms<sup>1</sup>. The first related distance function learning algorithm employs a weight-updating heuristic that changes weights based on class density information in clusters [5]. The second uses a randomized hill climbing algorithm to search for good distance function weights [3]. It selects a randomly created weight vector as the current weight vector, creates  $m$  ( $m = 30$  was used in the experiments) new weight vectors in the neighborhood of the current weight vector, and selects the weight vector with the highest value for the objective function  $R$  as the new current weight vector.

After applying each algorithm to learn weights, we apply one of two instance-based classifiers: 1-NN and NCC. The 1-NN classifier is the one nearest-neighbor classifier [12]. The Nearest Centroid Classifier (NCC) clusters the training data, retains the cluster centroids, and assigns to the centroid the most common class in the cluster. At each incoming point, the NCC finds the nearest centroid and assigns its class to the point.

The experimental procedure is to apply adaptive clustering to several datasets from the UCI machine learning repository [2]. We apply adaptive clustering with parameters  $k = C$  with  $|L| \in \{1, 30, 60\}$  and  $k = 5C$  with  $|L| = 60$  where  $C$  is the number of classes. The experiment performs 10 runs of 10-fold cross-validation on each of the datasets. Each run of cross-validation consists of 20 iterations each of which is run for a randomly created initial set of clusters. The weight vector with the highest reward is then sent as a parameter to the 1-NN and NCC classifiers to classify the examples in the test set. After an iteration is completed, the open list is cleared. Each iteration consists of 50 steps in the learning process. Each step expands state-action pairs based on value and follows the search process described in Section 2.1.

<sup>1</sup>More details about the experiments are available from the authors [1].

## 4. Experimental Results

Table 1 shows the results for experiments that compare adaptive clustering to 3 variants of the 1-NN classifier for a benchmark consisting of UCI datasets [2]. The  $\pm$  columns indicate statistical significance using the paired  $t$ -test of significance with significance level 0.10 for the 10 runs. For example, the adaptive clustering algorithm with parameters  $k = C, |L| = 1$  is significantly better than all three comparison algorithms for the autos dataset and worse only compared to algorithm 3 for the ionosphere dataset.

Experiment 1 in Table 1 compares the 1-NN classifier with an equally weighted distance function to 1-NN with learned weights. Adaptive clustering achieved a consistent improvement in accuracy for open list sizes greater than one. The diabetes dataset, which was significantly worse with open list size 1, became not significantly worse with a larger open list. On average, the results improve over all the data sets; furthermore, the results tend to improve with the size of the open list. In summary, the capability to explore multiple paths in parallel when searching for good distance functions leads to better accuracies for the tested classifiers. Moreover, increasing the number of clusters from  $C$  to  $5C$  increases the average accuracy for all datasets further.

Experiments 2 and 3 compare the traditional 1-NN classifier with learned weights from a density-based weight-updating heuristic [5], a randomized hill-climbing algorithm (RHC) [3], and the adaptive clustering algorithm. The results show a substantial improvement in several of the tested datasets over the two distance function learning algorithms. For the breast-cancer and glass datasets, the improvement is most obvious.

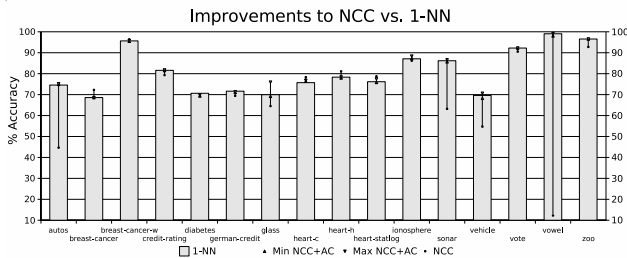
Figure 2 compares the NCC classifier using equal weights with the traditional 1-NN classifier and the NCC after learning weights. Interestingly, using adaptive clustering leads to a quite dramatic increase in accuracy when using NCC. Since this classifier makes its decision only using the cluster centroids, even a small change in the distance between points and centroids has a significant impact. For several of the datasets, a large difference in accuracy can be observed. Notably, the accuracy for the vowel dataset improved from 12% to over 99%. In addition, the NCC with learned weights achieved a higher average accuracy than the traditional 1-NN classifier. This interesting result suggests that adaptive clustering sufficiently improves cluster quality so that by just using cluster representatives instead of all objects in the dataset, one can achieve a better accuracy than the 1-NN classifier.

## 5. Conclusion

Users of clustering algorithms often have some idea of a good clustering of a dataset but often cannot quantify their ideas. Adaptive clustering uses external feedback to learn attribute weights for clusterings that better meet these objectives. Recent work in semi-supervised and supervised clustering demonstrates that the alteration of attribute weights in a distance metric can help to address user needs

Dataset	1: 1-NN		2: RHC [3]		3: IOWA [5]		$k = C,  L  = 1$			$k = C,  L  = 30$			$k = C,  L  = 60$			$k = 5C,  L  = 60$		
	$\mu$	$\sigma$	$\mu$	$\sigma$	$\mu$	$\sigma$	$\mu$	$\sigma$	$\pm$	$\mu$	$\sigma$	$\pm$	$\mu$	$\sigma$	$\pm$	$\mu$	$\sigma$	$\pm$
autos	74.55	1.32	55.30	4.70	59.76	4.18	75.53	1.44	<b>123</b>	75.13	1.78	<b>23</b>	75.28	1.22	<b>23</b>	75.38	1.33	<b>123</b>
b. c.	68.58	1.82	55.07	6.11	61.80	5.30	68.58	1.82	<b>23</b>	68.58	1.82	<b>23</b>	68.58	1.82	<b>23</b>	68.58	1.82	<b>23</b>
b. c. w.	95.65	0.34	96.42	0.96	97.04	0.27	95.38	0.57	23	95.49	0.39	23	95.28	0.32	23	95.64	0.44	3
credit-rating	81.58	0.65	81.59	2.20	75.52	5.27	81.86	0.83	<b>3</b>	81.74	0.57	<b>3</b>	81.77	0.72	<b>3</b>	81.84	0.87	<b>3</b>
diabetes	70.62	0.84	73.69	2.03	72.88	2.85	69.29	0.97	123	69.75	1.18	123	70.12	1.11	23	69.91	1.28	23
german-credit	71.63	0.68	70.54	2.35	68.61	3.76	71.28	0.73	1	70.92	0.80	1	71.20	0.64	1	71.42	0.60	
glass	69.95	0.93	68.60	3.04	66.77	3.15	68.89	2.00		72.20	1.89	<b>123</b>	72.01	2.56	<b>123</b>	76.26	2.18	<b>123</b>
heart-c	75.70	0.84	77.67	1.92	77.04	3.03	76.62	1.03	1	76.59	1.21		76.65	0.88	1	76.72	1.07	1
heart-h	78.33	1.06	75.90	2.18	76.48	3.77	78.32	1.34	2	78.40	1.11	2	77.86	1.62		78.43	1.67	2
heart-statlog	76.15	0.88	79.07	3.65	79.67	5.08	76.63	1.59		77.41	1.34	1	77.26	1.11		77.37	0.86	1
ionosphere	87.10	0.49	85.19	4.53	90.15	0.91	87.24	0.88	3	88.24	0.88	13	87.21	0.91		88.52	1.12	13
sonar	86.17	0.84	73.87	2.97	76.02	1.46	85.79	1.93	<b>23</b>	86.12	1.85	<b>23</b>	86.07	1.48	<b>23</b>	86.22	1.05	<b>23</b>
vehicle	69.59	0.67	N/A		63.96	1.50	69.14	0.89	<b>3</b>	68.83	1.09	<b>13</b>	68.59	1.25	<b>3</b>	70.55	1.12	<b>13</b>
vote	92.23	0.50	95.22	1.91	94.25	1.36	92.23	0.50	23	92.23	0.50	23	92.23	0.50	23	92.23	0.50	23
vowel	99.05	0.14	N/A		N/A		98.22	0.51	1	99.15	0.19	1	99.27	0.26	1	99.05	0.24	
zoo	96.55	0.50	92.85	1.96	92.25	3.06	96.55	0.50	<b>23</b>	96.55	0.50	<b>23</b>	96.55	0.50	<b>23</b>	96.55	0.50	<b>23</b>
Average	80.84		77.21		76.81		80.72			81.08			81.00			81.54		

**Table 1. Average and standard deviations for improvements to the 1-NN classifier. The  $\pm$  columns show improvement (bold) and degradation.**



**Figure 2. Results comparing the 1NN classifier against the NCC classifier and its best and worst improvements with adaptive clustering.**

[4, 6, 8, 14]. Most of those techniques, however, require the user to assess cluster quality at a very low level of granularity. In adaptive clustering, it is sufficient to assign a reward to the clustering as a whole.

We discussed the results of a case study that applied adaptive clustering to learn distance function weights to maximize externally defined objectives. In particular, adaptive clustering was used to enhance existing instance-based classifiers using information gain as the objective function. Using the learned weights, the 1-NN classifier achieves a statistically significant improvement in accuracy for several datasets. Furthermore, a classifier that only uses the cluster representatives to classify incoming instances uses the learned weights to achieve, on average, the same or better accuracy than the more costly 1-NN classifier. Based on its improved accuracy, the technique can find more informative clusters and representatives in this domain.

## References

[1] A. Bagherjeiran, C. F. Eick, and R. Vilalta. Adaptive clustering: Better representatives with reinforcement learn-

ing. Technical Report UH-CS-05-06, University of Houston Computer Science Department, 2005.

- [2] C. Blake and C. Merz. UCI repository of machine learning databases, 1998.
- [3] C.-S. Chen. Using weight updating heuristics and randomized hill climbing distance function learning. Master's thesis, University of Houston, 2005.
- [4] D. Cohn, R. Caruana, and A. McCallum. Semi-supervised clustering with user feedback. Technical Report 2003-1892, Cornell University, 2003.
- [5] C. F. Eick, A. Rouhana, A. Bagherjeiran, and R. Vilalta. Using clustering to learn distance functions for supervised similarity assessment. In *Proc. Int'l Conf. on Machine Learning and Data Mining*, 2005.
- [6] C. F. Eick and N. Zeidat. Using supervised clustering to enhance classifiers. In *Proc. 15th Int'l Symp. on Methodologies for Intelligent Systems*, Saratoga Springs, NY, 2005.
- [7] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *J. Artif. Intell. Res.*, 4:237–285, 1996.
- [8] T. Kohonen, J. Hynninen, J. Kangas, J. Laaksonen, and K. Torkkola. LVQ PAK: The learning vector quantization program package. Technical report, Helsinki University of Technology, 1996.
- [9] J. M. McQueen. Some methods of classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symp. on Mathematical Statistics and Probability*, pages 281–297, 1967.
- [10] A. W. Moore and C. G. Atkeson. Prioritized sweeping–reinforcement learning with less data and less time. *Machine Learning*, 13:103–130, 1993.
- [11] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, New York, 1994.
- [12] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition, 2003.
- [13] C. J. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.
- [14] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems 15*. MIT Press, 2003.