Expert Systems with Applications 40 (2013) 6894–6900

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

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

ATISA: Adaptive Threshold-based Instance Selection Algorithm

George D.C. Cavalcanti [a,*], Tsang Ing Ren [a,b], Cesar Lima Pereira [a]

[a] Centro de Informática, Universidade Federal de Pernambuco, Brazil
[b] iMinds – Vision Lab, Department of Physics, University of Antwerp, Universiteitsplein 1, B-2610 Wilrijk, Belgium

ARTICLE INFO

Keywords:
Instance selection
Instance-based learning algorithms

ABSTRACT

Instance reduction techniques can improve generalization, reduce storage requirements and execution time of instance-based learning algorithms. This paper presents an instance reduction algorithm called Adaptive Threshold-based Instance Selection Algorithm (ATISA). ATISA aims to preserve important instances based on a selection criterion that uses the distance of each instance to its nearest enemy as a threshold. This threshold defines the coverage area of each instance that is given by a hyper-sphere centered at it. The experimental results show the effectiveness, in terms of accuracy, reduction rate, and computational time, of the ATISA algorithm when compared with state-of-the-art reduction algorithms.

1. Introduction

In supervised learning, a training set of instances is given as input to a machine learning algorithm. Each instance is formed by a descriptor vector and a label that represent its properties and class respectively. After the learning process, unknown instances are presented to the algorithm that must infer their classes through generalizations (Mitchell, 1990).

Instance-based learning algorithms use the whole set of instances from the training set to construct inference structures. The Nearest Neighbor Rule (Cover & Hart, 1967) is a well-known example of this kind of algorithm. It requires a large memory space since the whole training instance set has to be stored. Each query pattern is compared with every pattern in the training set. An unknown instance is assigned to the class of the most similar instance in the training set. Instead of using only the nearest instance, this rule can be generalized (in the sense of proximity) by using the classes of the k most similar neighbors; this algorithm is called k-Nearest Neighbors (kNN). There are other algorithms that use this neighborhood concept, for example: the Center-based Nearest Neighbor classifier (Gao & Wang, 2006). In spite of the efficiency of instance-based learning algorithms, they suffer from problems such as: large storage requirements, low prediction performance and sensitivity to noisy and redundant data (David, Aha, & Albert, 1991).

Instance reduction techniques are used as a pre-processing step on the data set. One of its objectives is to overcome the limitations faced by instance-based learning algorithms. This is obtained by reducing the size of the training set by removing "irrelevant" instances. Reduction techniques can be classified in three types: condensation, edition and hybrid (Garcia, Derrac, Cano, & Herrera, 2012; García, Marqués, & Sánchez, 2012; Nanni & Lumini, 2011). Edition methods aim to remove noisy instances and Condensation methods search for a consistent subset of the training set. A consistent subset is formed by eliminating instances in the training set that do not affect the classification accuracy of the whole training set. Hybrid methods compute a subset of the training set combining the characteristics of Edition and Condensation methods; so, noisy and superfluous instances are eliminated.

The Condensed Nearest Neighbor Rule (CNN) (Hart, 1968) is one of the first method that attempts to reduce the number of instances in a training set. It removes any well classified instance using the nearest neighbor rule. The CNN algorithm maintains border instances and eliminates redundant ones. Another example of a reduction technique is the Edited Nearest Neighbor Rule (ENN) (Wilson, 1972) that keeps central instances by smoothing the decision boundaries. CNN is a condensation algorithm while ENN is a noise filter. There are many algorithms that can not be classified neither as condensation nor noise filter; they are hybrid selection approaches. An example is the Iterative Case Filtering algorithm (ICF) (Brighton & Mellish, 1999) that iteratively removes instances according to a filtering rule. Wilson and Martinez (2000) proposed a series of algorithms called Decremental Reduction Optimization Procedure. Marchiori (2008) proposed a network-based representation of a training set, called Hit Miss Network (HMN) that provides a compact description of the nearest neighbor relation over pairs of instances from each pair of classes.

* Corresponding author. Address: Universidade Federal de Pernambuco (UFPE), Centro de Informática (CIn), Av. Jornalista Anibal Fernandes, s/n. Cidade Universitária 50740-560, Recife, PE, Brazil. Tel.: +55 81 2126 8430x4346; fax: +55 81 2126 8438.

E-mail addresses: gdcc@cin.ufpe.br (G.D.C. Cavalcanti), tir@cin.ufpe.br (T.I. Ren), clp@cin.ufpe.br (C.L. Pereira).

URL: http://www.cin.ufpe.br/~viisar (G.D.C. Cavalcanti).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.06.053

This paper introduces an instance reduction algorithm called Adaptive Threshold-based Instance Selection Algorithm (ATISA, for short). It aims to find the best set of instances in order to reduce the size of the stored set of instances while maintaining or even improving the generalization accuracy. ATISA does not eliminate an instance based on its spacial location - central or border points. It selects important points based on a threshold that defines a coverage area per instance.

This work is organized as follows. The Adaptive Threshold-based Instance Selection Algorithm or ATISA is described in Section 2. The experimental results are presented in Section 3. Finally, Section 4 shows the conclusions.

2. Adaptive Threshold-based Instance Selection Algorithm (ATISA)

The Adaptive Threshold-based Instance Selection Algorithm (ATISA) is an incremental instance selection technique. Based on a threshold, the algorithm decides whether or not instances are included in the final subset. ATISA has three different approaches: ATISA1, ATISA2 and ATISA3. These three versions are described in the next subsections.

Algorithm 1 : Adaptive Threshold-based Instance Selection Algorithm 1 (ATISA1)

```
Require: $T$ {The training set}
1: $T_f = T$
2: for $x_i \in T$ do
3:    if $class(x_i) \neq class_{max}(NN_k(x_i,T))$ then
4:       $T_f = T_f \setminus \{x_i\}$
5:    end if
6: end for
7: for $x_i \in T_f$ do
8:    $\theta(x_i) = d(x_i, NE(x_i,T_f))$
9: end for
10: $S = \emptyset$
11: for $x_i \in T_f$ do
12:    $N = NN_k(x_i,S)$
13:    if $class(x_i) \neq class_{max}(N)$ or $d'(x_i,N) > \theta'(N)$ then
14:       $S = S \cup \{x_i\}$
15:    end if
16: end for
17: return $S$ {The subset of the selected instances from the training set}
```

2.1. Adaptive Threshold-based Instance Selection Algorithm 1 (ATISA1)

In general terms, the algorithm applies a noise filter to the set and afterwards it calculates one threshold for each instance. Each remaining instance is classified based on its nearest neighbor. In the case of an erroneous classification, the instance is inserted into the final set. Otherwise, it is added only if the distance to its nearest neighbor is greater than the threshold of that neighbor. Algorithm 1 shows the pseudocode of ATISA1.

$NN_k(a,A)$ represents the set of the k nearest neighbors of the instance a in the training set A, while $class(a)$ represents the class of the instance a. $class_{max}(B)$ receives the set of instances B and returns its predominant class. $NE(a,A)$ represents the nearest enemy of a in the set A. That nearest enemy is the nearest neighbor of a that has a different class. The distance between instances a and b is given by $d(a,b)$. The threshold of each instance is represented by the function $\theta(a)$, where a corresponds to an instance. To use this algorithm with $k > 1$, the distance and threshold functions have the following variation: $d'(a,A)$ and $\theta'(A)$. The former returns the average distance between instance a and all instances in the set A, and the latter returns the average threshold calculated over the set of instances A.

The selection solution of ATISA1 has three steps: ***pre-processing*** (filter), ***thresholds assignment*** and ***reduced set generation***.

The ***pre-processing*** is a filtering process that aims to remove noisy data from the training set (lines 1 to 6 in Algorithm 1). The *Edited Nearest Neighbor Rule* (ENN) (Wilson, 1972) is the algorithm used for this purpose. ENN removes any instance that is not correctly classified by its nearest neighbor. In other words, given an instance x_i and its nearest neighbor $NN_k(x_i)$, $k = 1$, this instance is removed if its class $class(x_i)$ is different from the class of its nearest enemy $class(NN_k(x_i))$. The algorithm ENN enlarges the spaces between classes, smoothing the decision boundaries.

Threshold assignment: The threshold of an instance x_i is defined by the distance to its nearest enemy $NE(x_i)$ (lines 7 to 9 in Algorithm 1). Fig. 1 illustrates the threshold θ of the instance x. One threshold is assigned for each instance, it defines an area that is covered by the instance. All the instances laying inside this delimited area have the same class of the instance x.

The ***reduced set generation*** (lines 10 to 16 in Algorithm 1) works as follows; after pre-processing, all remaining instances are randomly selected. One selected instance x_i is inserted in the set S if it is correctly classified by its nearest neighbors in S. In a particular case, a misclassified instance can be inserted in S. However, this only occurs if its distance to the nearest neighbor is greater than the threshold of this neighbor (for $k > 1$, average distances and thresholds are used, regardless their classes).

Each threshold can be interpreted as a radius that forms a hyper-sphere centered at the instance position. If one instance is correctly classified, even standing outside this hyper-sphere, it must be added to the final set, otherwise, it can be misclassified in the upcoming steps of the process.

Fig. 2 shows an example of the ATISA1 third step: the procedure to obtain the reduced set. In Fig. 2(a), random indexes were assigned to each instance aiming to create an order. In Fig. 2(b), the first two instances were inserted, because they were not correctly classified and, at that point of the algorithm, the set S is empty. After, the instance with index 3 is not inserted in S. This happens because it is correctly classified, besides, it is inside the circle created by the threshold of its nearest neighbor (Fig. 2(c)). In this toy sample, the threshold of instance 1 is the distance between instances of indexes 1 and 9. In the next verification, instance 5 is checked. Although correctly classified, it is out of

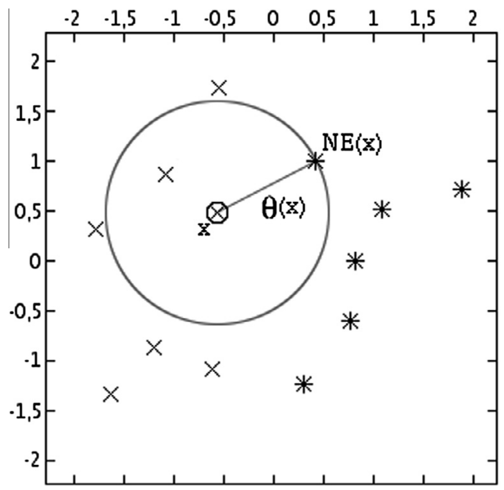
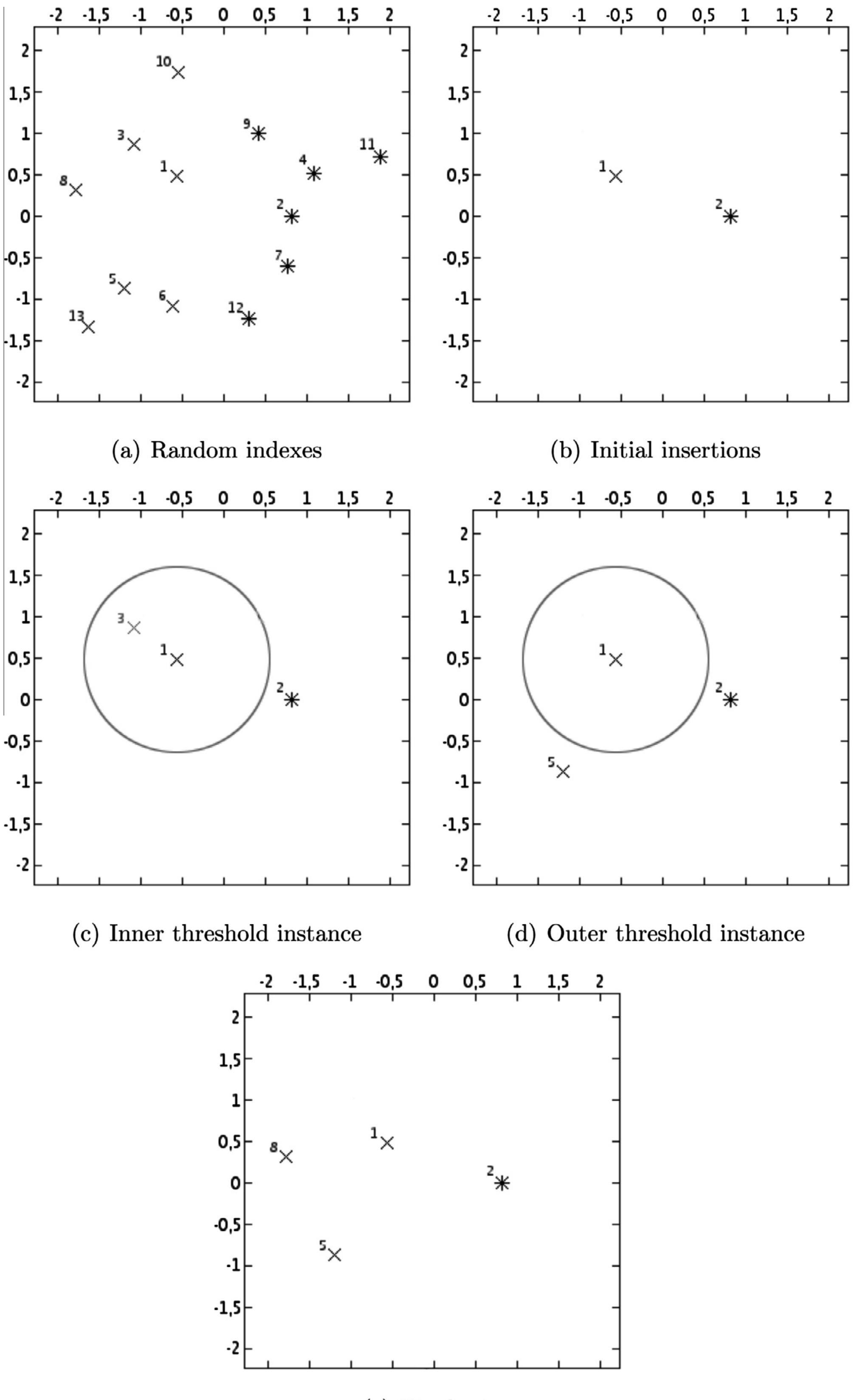


Fig. 1. Definition of the threshold θ for the instance x. NE (x) represents the nearest enemy of the instance x.



(a) Random indexes (b) Initial insertions

(c) Inner threshold instance (d) Outer threshold instance

(e) Final set

Fig. 2. An example of how ATISA1 obtain the reduced set.

the area covered by its nearest neighbor (Fig. 2(d)); so, it is added to the set *S*. At the end, the final set *S* is shown in Fig. 2(e).

Algorithm 2: Adaptive Threshold-based Instance Selection Algorithm 2 (ATISA2)

```
Require: $T$ {The training set}
1: $T_f = T$
2: for $x_i \in T$ do
3:     if $class(x_i) \neq class_{max}(NN_k(x_i,T))$ then
4:         $T_f = T_f \setminus \{x_i\}$
5:     end if
6: end for
7: for $x_i \in T_f$ do
8:     $\theta(x_i) = d(x_i, NE(x_i,T_f))$
9: end for
10: $T_f = sort(T_f)$
11: $S = \emptyset$
12: for $x_i \in T_f$ do
13:     $N = NN_k(x_i,S)$
14:     if $class(x_i) \neq class_{max}(N)$ or $d'(x_i,N) > \theta'(N)$ then
15:         $S = S \cup \{x_i\}$
16:     end if
17: end for
18: return $S$ {The subset of the selected instances from the training set}
```

2.2. Adaptive Threshold-based Instance Selection Algorithm 2 (ATISA2)

In the second version of the ATISA algorithm, a step before the process of creating the final reduced set is added. Instead of randomly selecting the instances, as done by ATISA1, an ordering procedure is required by ATISA2. So, the distance between each instance and its nearest enemy (threshold value) is calculated. The first selected instance is the one having the largest distance; the second largest distance corresponds to the second selected instance; and so on.

Initially, instances with the largest thresholds are inserted in *S*. This means that these instances cover a representative area of the feature space. Based on that, it is probable that the last inserted instances (the ones with small thresholds) will be placed inside their neighbors coverage areas. So, they will not be inserted in *S*. In comparison with ATISA1, this procedure reduces the size of the final instance set.

Algorithm 2 shows the pseudocode of ATISA2. The difference between ATISA1 and ATISA2 is in line 10. There is a *sort* function in this line and it is responsible for ordering the instances from the largest to the smallest threshold value.

Fig. 3 presents the behavior of ATISA2 for the same toy problem shown in Section 2.1. In Fig. 3(a), the instances are labeled following the ordering obtained by ATISA2. The first visited instances are the ones more distant from their nearest enemy. In this example, the first two inserted instances are the ones with labels 1 and 2. These instances can be viewed in Fig. 3(b), each with its respective coverage area. They are the instances with greater threshold in the instance set, and, in this problem, all other instances are placed inside the coverage area of these two instances. As they are correctly classified by instances 1 and 2, they are not included in *S*. There is only one exception: the instance with index 12 that has instance 1 as its nearest neighbor. Instance 12 is added to *S* because the distance between this instance and instance 1 is greater than the threshold of the instance 1. The final set of instances generated by ATISA2 is presented in Fig. 3(b).

Algorithm 3: Adaptive Threshold-based Instance Selection Algorithm 3 (ATISA3)

```
Require: $T$ {The training set}
1: $T_f = T$
2: for $x_i \in T$ do
3:     if $class(x_i) \neq class_{max}(NN_k(x_i,T))$ then
4:         $T_f = T_f \setminus \{x_i\}$
5:     end if
6: end for
7: for $x_i \in T_f$ do
8:     $\theta(x_i) = d(x_i, NE(x_i,T_f))$
9: end for
10: $T_f = sort(T_f)$
11: $S = one\_by\_class(T_f)$
12: $update\_thresholds(S)$
13: for $x_i \in T_f$ do
14:     $N = NN_k(x_i,S)$
15:     if $class(x_i) \neq class_{max}(N)$ or $d'(x_i,N) > \theta'(N)$ then
16:         $S = S \cup \{x_i\}$
17:         $update\_thresholds(S)$
18:     end if
19: end for
20: return $S$ {The subset of the selected instances from the training set}
```

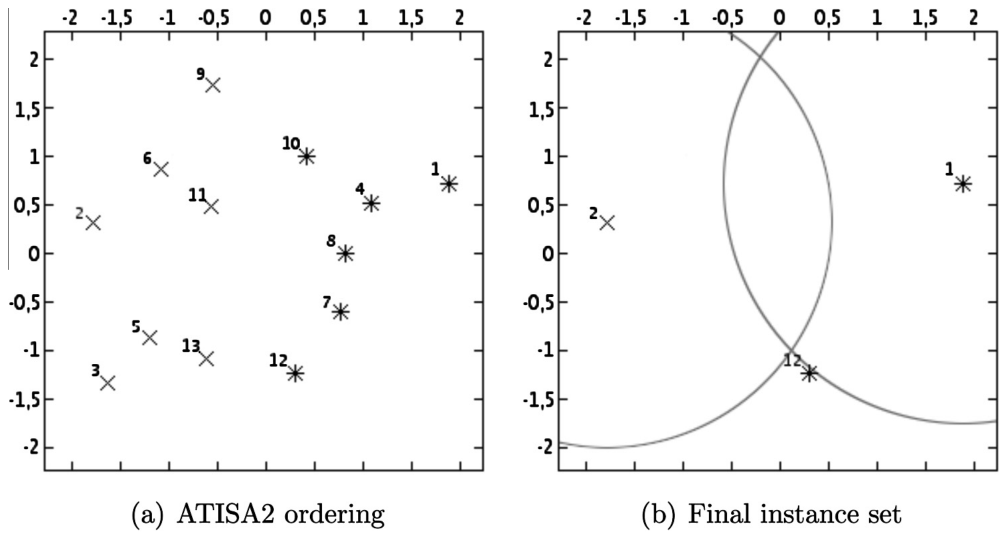


Fig. 3. (a) Show the instance labeled; (b) The reduced instance set obtained after the ATISA2 procedure.

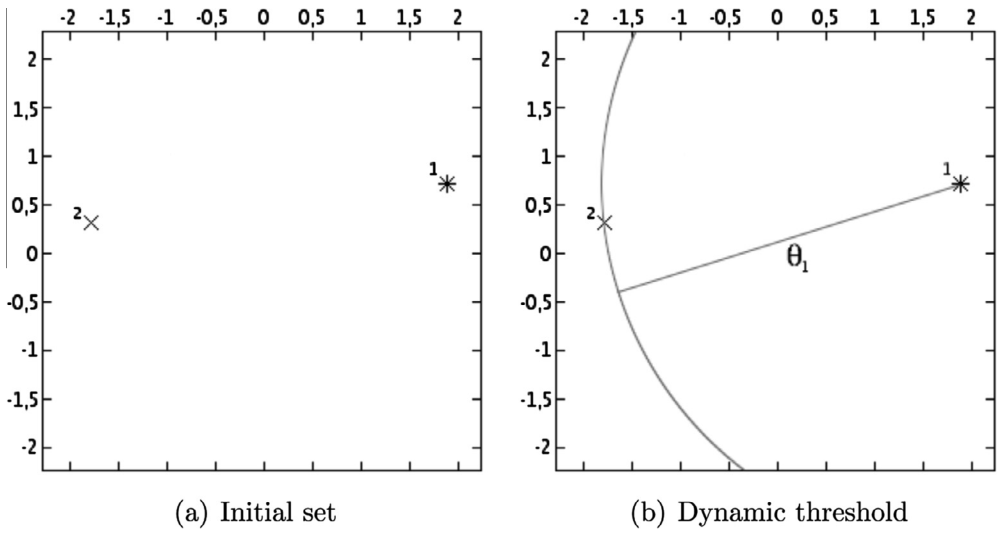


Fig. 4. An example of the ATISA3 procedure to updated the threshold dynamically.

Table 1
Accuracy rates of experiments on the datasets using the 3-Nearest Neighbor classifier. *R* represents the reduction percentage.

Databases	3-NN	ATISA1	R	ATISA2	R	ATISA3	R
annealing	92.48	92.61	82.01	91.85	87.18	88.47	91.20
australian	85.94	85.80	79.82	85.36	85.27	85.80	91.01
breast cancer	95.42	95.42	90.53	94.71	94.91	93.71	96.92
crx	85.51	86.09	77.91	82.90	83.20	82.32	89.97
hepatitis	83.87	82.58	81.21	80.00	85.59	81.94	90.46
image segmentation	96.23	94.20	86.25	93.07	88.82	91.52	91.90
ionosphere	85.19	86.61	79.61	84.62	82.46	81.48	94.49
iris	94.67	98.00	81.56	94.00	84.00	92.00	88.74
liver	65.22	62.32	65.67	60.58	73.94	62.03	83.09
pima diabetes	74.35	72.53	79.25	72.01	84.46	71.48	90.39
promoters	84.91	83.02	61.64	83.02	72.96	72.64	78.09
sonar	83.65	79.81	56.46	79.81	65.81	73.08	76.66
soybean	89.58	81.76	66.77	78.18	77.24	79.15	77.20
tic tac toe	88.83	89.14	78.84	91.44	83.59	89.87	83.30
voting	92.64	93.79	87.28	93.79	92.03	91.95	95.76
wine	96.63	94.94	77.59	96.07	78.40	90.45	87.58
Average	87.19	86.16	77.03	85.09	82.49	82.99	87.92
Median	87.39	86.35	79.43	84.99	83.79	84.06	90.18
Wilcoxon	~	n/a	n/a	+	–	+	–

2.3. Adaptive Threshold-based Instance Selection Algorithm 3 (ATISA3)

ATISA3 is an evolution of ATISA2 that calculates the threshold dynamically. In other words, when the set *S* is modified, a procedure is started to update all the instances thresholds. ATISA3 reaches even higher reduction rates than ATISA2. Initially, thresholds are more likely to have higher values. This happens due to the ordering inherited from ATISA2 combined with the fact that thresholds are dynamically calculated. This dynamic threshold update increases the chance of an instance to be within the coverage area of another instance.

The pseudocode of ATISA3 is presented in Algorithm 3. Its main difference in comparison to ATISA2 is in lines 11 to 19. In line 11, the function *S* = *one_by_class*(*A*) receives a set of instance *A* as input and returns a set having one instance of each class. Each class is represented by the instance with the largest threshold. After, the thresholds of the instances in *S* are updated and this is performed by the function *update_thresholds*(*S*) – line 12.

ATISA3 starts with one instance per class, as shown in Fig. 4(a). In this case, the instances labeled 1 and 2 have the same threshold value (Fig. 4(b)) that corresponds to the distance between them. This toy problem has only two different classes. It is possible to observe that all other instances are correctly classified and they are inside the boundaries of their respective neighbor threshold. So, the reduced set constructed by ATISA3 has only two instances: 1 and 2.

3. Experimental results

Experiments were carried out using 16 databases from the UCI Machine Learning Repository (Frank & Asuncion, 2010): *annealing*, *australian*, *breast cancer*, *crx*, *hepatitis*, *image segmentation*, *ionosphere*, *iris*, *liver*, *pima diabetes*, *promoters*, *sonar*, *soybean*, *tic tac toe*, *voting* and *wine*. All numerical values were normalized to a number between 0 and 1. These databases have different number of classes, attributes (categorical or numerical) and instances; also, some of them have instances with missing values.

ATISA was compared with the following techniques: *Decremental Reduction Optimization Procedure* 3 (DROP3) (Wilson & Martinez, 2000), *Hit Miss Network for Editing Iterated* (HMN-EI) (Marchiori, 2008) and *Iteractive Case Filtering* (ICF) (Brighton & Mellish, 1999). Garcia et al. (2012) showed that DROP3 and ICF are between the methods most used for comparison purposes and that HMN-EI has found very competitive accuracy rates. It is also important to

Table 2
Results of experiments on the data sets. R is the percentage of training points removed. The symbols +, – and ~ show the number of times ATISA1 average accuracy (storage reduction) is significantly better (+), significantly worse (–) or similar (~) than the other algorithm, according to a paired t-test at 0.05 significance level. In the "Wilcoxon" row, + indicates ATISA1 was significantly better than the other algorithm according to a Wilcoxon test for paired samples, – indicates it was significantly worse and ~ indicates no significant difference.

Databases	DROP3	R	HMN-EI	R	ICF	R	ATISA1	R
annealing	$92.73^{\sim}$	88.36^{-}	89.47^{+}	38.64^{+}	86.34^{+}	84.18^{-}	92.61	82.01
australian	$84.49^{\sim}$	85.04^{-}	83.33^{+}	48.41^{+}	$84.64^{\sim}$	82.00^{-}	85.80	79.82
breast cancer	$94.56^{\sim}$	92.99^{-}	$96.14^{\sim}$	52.03^{+}	$94.42^{\sim}$	$90.00^{\sim}$	95.42	90.53
crx	$85.07^{\sim}$	86.33^{-}	$84.49^{\sim}$	50.68^{+}	82.90^{+}	82.69^{-}	86.09	77.91
hepatitis	$83.23^{\sim}$	89.18^{-}	$84.52^{\sim}$	54.48^{+}	$83.87^{\sim}$	87.24^{-}	82.58	81.21
image segmentation	94.94^{-}	90.35^{-}	92.03^{+}	37.88^{+}	90.65^{+}	79.86^{+}	94.20	86.25
ionosphere	$85.76^{\sim}$	95.28^{-}	$86.61^{\sim}$	59.54^{+}	72.36^{+}	94.90^{-}	86.61	79.61
iris	94.00^{+}	87.70^{-}	$96.00^{\sim}$	44.44^{+}	$96.00^{\sim}$	71.19^{+}	98.00	81.56
liver	$63.77^{\sim}$	74.56^{-}	$57.68^{\sim}$	57.01^{+}	$56.52^{\sim}$	77.55^{-}	62.32	65.67
pima diabetes	$72.27^{\sim}$	83.44^{-}	$74.22^{\sim}$	50.30^{+}	$70.44^{\sim}$	85.63^{-}	72.53	79.25
promoters	$84.91^{\sim}$	71.58^{-}	$84.91^{\sim}$	$59.85^{\sim}$	$74.53^{\sim}$	$64.98^{\sim}$	83.02	61.64
sonar	$77.88^{\sim}$	70.67^{-}	$75.48^{\sim}$	57.96^{-}	73.56^{+}	67.89^{-}	79.81	56.46
soybean	$82.41^{\sim}$	65.58^{+}	$80.78^{\sim}$	37.79^{+}	$79.48^{\sim}$	55.77^{+}	81.76	66.77
tic tac toe	83.61^{+}	83.59^{-}	81.00^{+}	82.73^{-}	86.53^{+}	31.53^{+}	89.14	78.84
voting	$92.87^{\sim}$	94.35^{-}	90.11^{+}	55.40^{+}	90.80^{+}	$87.77^{\sim}$	93.79	87.28
wine	$96.07^{\sim}$	84.83^{-}	$96.63^{\sim}$	66.72^{+}	$94.38^{\sim}$	87.64^{-}	94.94	77.59
Average	85.54	83.99	84.59	53.37	82.34	76.93	86.16	77.03
Median	84.99	85.68	84.71	53.25	84.25	82.34	86.35	79.43
Wilcoxon	~	–	+	+	+	~	n/a	n/a

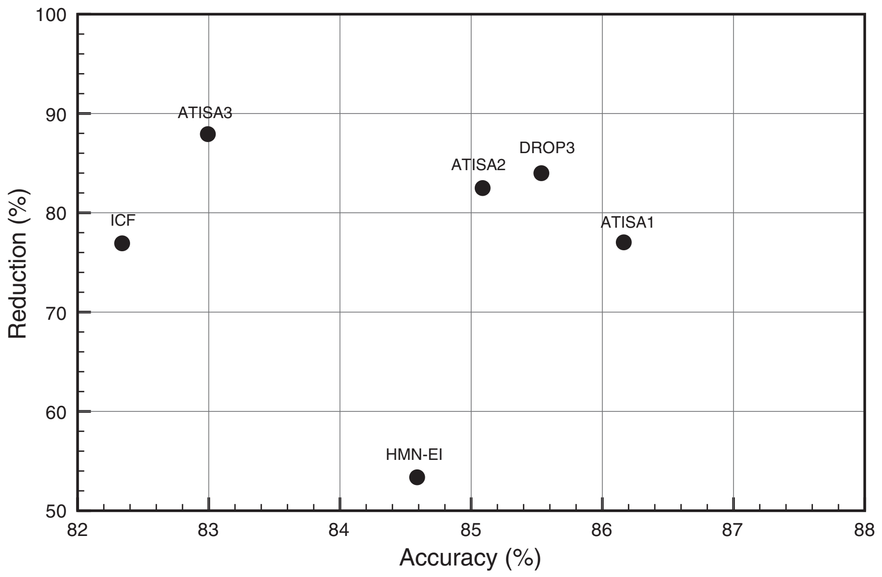


Fig. 5. Accuracy x Reduction chart.

emphasize that there are in the literature previous methods that construct a rank in order to find the best instances, for example: the InstanceRank method (Vallejo, Troyano, & Ortega, 2010). However, this method did not perform as well as the DROP3, ICF and HMN-EI methods. So, it was not used for comparison purposes.

The *Heterogeneous Euclidean Overlap Metric* (HEOM) (Wilson & Martinez, 1997) was used in every required situation, i.e., in the presence of hybrid attributes (numerical and categorical). HEOM uses the Euclidean distance in numeric attributes and the Hamming distance in categorical ones. Every distance between attributes is calculated and the value returned by HEOM corresponds to the square root of the sum of these distances, as it is described in Eq. (1).

$$HEOM(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{a=1}^{m} d_a(x_a, y_a)} \qquad (1)$$

where $\mathbf{x}$ and $\mathbf{y}$ are instances, m is the number of attributes and x_a and y_a represent the attributes of the instances $\mathbf{x}$ and $\mathbf{y}$ respectively.

$$d_a(x,y) = \begin{cases} (x-y)^2, & \text{numerical} \\ 0, & \text{categorical } (x=y) \\ Max, & \text{categorical } (x \neq y) \text{ or missing} \end{cases} \qquad (2)$$

When the attribute values are numeric, the squared difference is calculated as defined in Eq. (2). If they are categorical and have the same value, the attribute distance is minimal (zero). But, if they are categorical and do not have the same value, the attribute distance is maximal ($Max = 1$, because the whole dataset was normalized to a number between 0 and 1).

All experiments reported use the *k-Nearest Neighbor* classifier ($k = 3$) and the 10-fold cross-validation method. After preliminary experiments, the number of nearest neighbor of the function $NN_k(\cdot)$ was also set to 3. Tables 1 and 2 show the experimental results. For each technique, classification accuracy and reduction rate are presented (*R* denotes reduction). In order to assess whether differences in accuracy and storage reduction are significant, a non-parametric Wilcoxon signed ranks test for zero median at a 0.05 significance level was used. The symbols +, − and ∼, in the row labeled "Wilcoxon", indicate that ATISA1 is significantly better, worse or equivalent compared with the other algorithms, respectively.

Table 1 presents a comparison between the three versions of ATISA. In this table, the 3-NN classifier is used as a reference result. It shows that ATISA3 obtains better reductions rates than the other ones. Observing the Wilcoxon test, ATISA1 is significantly better than ATISA2 and ATISA3 and similar to 3-NN in terms of accuracy rate. On the other hand, ATISA1 is significantly worse than ATISA2 and ATISA3, in terms of reduction.

Table 2 presents a comparison between ATISA1 (the best of the three proposed methods) and three reduction techniques: DROP3, HMN-EI and ICF. ATISA1 is significantly better than ICF and HMN-EI in classification performance. Observing the reduction, ATISA1 is better the HMN-EI and similar to ICF. In terms of accuracy rates, ATISA1 and DROP3 show similar results.

Fig. 5 shows a two dimensional plot: accuracy versus reduction. Each point in this figure represents the average value showed in Tables 1 and 2. The top right corner is the best possible solution. ATISA1 is ahead in terms of accuracy and ATISA3 reaches the best reduction.

The average running time in seconds shows that the ATISA algorithms are faster than the other ones. The reduction process of the DROP3 algorithm took, in average, 4.29 s, while ATISA1 was almost three times faster, 1.56 s. ICF and HMN-EI took 3.16 and 2.44 respectively. The values showed above represent the average time of the reduction process per algorithm using all the datasets. The algorithms were implemented in C++ using the GCC 4.1.2 compiler. All the experiments were executed in a Linux (Kernel 2.6.18) controlled environment using an Intel Xeon 3 GHz with 4 GB of RAM.

4. Conclusions

This paper proposed three algorithms to perform instance selection: Adaptive Threshold-based Instance Selection Algorithm 1,2,3 (ATISA, for short). ATISA calculates a hypersphere centered at each instance and this hypersphere defines the coverage area of the instance. The radius of the hypersphere is calculated based on the distance between the instance and its nearest enemy. Instances that belong to other instances area of coverage are removed. The proposed algorithms maintain important points to define each class, it does not matter if these points belong to the border or are inner points of the class.

The three versions of the algorithm were evaluated using UCI databases. In general terms, ATISA1 obtains the best classification performance, ATISA3 obtains the best rates in the reduction task and ATISA2 is a balanced choice between accuracy and reduction rates. ATISA algorithms were compared with the state-of-the-art instance selection methods. This empirical analysis shows the effectiveness of the proposed techniques in terms of accuracy and reduction rates. Moreover, when the computational cost was evaluated, ATISA was the faster algorithm when compared with DROP3, ICF and HMN-EI.

Acknowledgment

This research was partially supported by CNPq, Capes and Facepe.

References

Brighton, H., & Mellish, C. (1999). On the consistency of information filters for lazy learning algorithms. In *European conference on principles of data mining and knowledge discovery* (pp. 283–288).

Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory, 13*, 21–27.

David, W., Aha, D. K., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning, 6*, 37–66.

Frank, A., & Asuncion, A. (2010). UCI machine learning repository.

Gao, Q.-B., & Wang, Z.-Z. (2006). Center-based nearest neighbor classifier. *Pattern Recognition, 40*, 346–349.

Garcia, S., Derrac, J., Cano, J., & Herrera, F. (2012). Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 34*, 417–435.

García, V., Marqués, A. I., & Sánchez, J. S. (2012). On the use of data filtering techniques for credit risk prediction with instance-based models. *Expert Systems with Applications, 39*, 13267–13276.

Hart, P. E. (1968). The condensed nearest neighbor rule. *IEEE Transactions on Information Theory, 14*, 515–516.

Marchiori, E. (2008). Hit miss networks with applications to instance selection. *Journal of Machine Learning Research, 9*, 997–1017.

Mitchell, T. M. (1990). The need for biases in learning generalizations. In J. Shavlik & T. Dietterich (Eds.), *Readings in machine learning* (pp. 184–191). Morgan Kaufman.

Nanni, L., & Lumini, A. (2011). Prototype reduction techniques: A comparison among different approaches. *Expert Systems with Applications, 38*, 11820–11828.

Vallejo, C. G., Troyano, J. A., & Ortega, F. J. (2010). Instancerank: Bringing order to datasets. *Pattern Recognition Letters, 31*, 133–142.

Wilson, D. L. (1972). Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man and Cybernetics, 2*, 408–421.

Wilson, D. R., & Martinez, T. R. (1997). Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research, 6*, 1–34.

Wilson, D. R., & Martinez, T. R. (2000). Reduction techniques for instance-based learning algorithms. *Machine Learning, 38*, 257–286.