Neurocomputing 000 (2018) 1–13

Contents lists available at ScienceDirect

Neurocomputing

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

Pruning strategies for nearest neighbor competence preservation learners☆

Fabrizio Angiulli[a,*], Estela Narvaez[b]

[a] DIMES Department University of Calabria, 87036 Rende, CS, Italy
[b] Faculty of Enginnering, National University of Chimborazo, 060150 Riobamba, Ecuador

ARTICLE INFO

Article history:
Received 9 August 2017
Revised 9 December 2017
Accepted 22 April 2018
Available online xxx

Communicated by Dr Xiaofeng Zhu

Keywords:
Classification
Nearest neighbor rule
Training-set consistent subset
Overfitting
Pessimistic error estimate

ABSTRACT

In order to alleviate both the spatial and temporal cost of the nearest neighbor classification rule, competence preservation techniques aim at substituting the training set with a selected subset, known as consistent subset. In order to improve generalization and to prevent induction of overly complex models, in this study the application of the Pessimistic Error Estimate (PEE) principle in the context of the nearest neighbor rule is investigated. Generalization is estimated as a trade-off between training set accuracy and model complexity. As major results, it is shown that PEE-like selection strategies guarantee to preserve the accuracy of the consistent subset with a far larger reduction factor and, moreover, that sensible generalization improvements can be obtained by using a reduced subset. Moreover, comparison with state-of-the-art hybrid prototype selection methods highlight that the here introduced FCNN-PAC strategy is able to obtain a model of size comparable to that obtained by the best prototype selection methods, with far smaller time requirements, corresponding to four orders of magnitude on medium-sized datasets.

1. Introduction

The nearest neighbor decision rule [1] (nearest neighbor rule for short) assigns to an unclassified sample point the classification of the nearest of a set of previously classified points. A strong point of the nearest neighbor rule is that, for all distributions, its probability of error is bounded above by twice the Bayes probability of error [1–3]. That is, it may be said that half the classification information in an infinite size sample set is contained in the nearest neighbor.

Naive implementation of the nearest neighbor rule requires to store all the previously classified data points, and then to compare each sample point to be classified to each stored point.

In order to reduce both space and time requirements the concept of *training set consistent subset*, that is a subset of the original training set that correctly classifies all the training samples, was introduced by Hart [4] together with an algorithm, called the CNN rule (for Condensed nearest neighbor rule), to determine a consistent subset of the original sample set. Since then different techniques have been introduced [5–8], referred to as training set reduction, training set condensation, reference set thinning, and prototype selection algorithms.

Using a training set consistent subset, instead of the entire training set, to implement the nearest neighbor rule has the additional advantage that it may guarantee better classification accuracy. Indeed, [6] showed that the VC dimension of an nearest neighbor classifier is given by the number of reference points in the training set.

Thus, in order to achieve a classification rule with controlled generalization, it is better to replace the training set with a small consistent subset.

In this study, motivated by approaches used in the context of other classification algorithms in order to improve generalization and to prevent induction of overly complex models, such as in the case of decision trees, we investigate the application of the Pessimistic Error Estimate (PEE) principle [9] in the context of the nearest neighbor rule.

With this aim, we relax the notion of consistency of a subset by introducing the notion of α-consistent subset ($\alpha \in [0, 1]$), that is a subset that correctly classifies at least the α fraction of the training set. Then we describe a variant of the FCNN algorithm [8], called α-FCNN rule, that computes an α-consistent subset.

☆ A preliminary version of this work appears under the title "Pruning Nearest Neighbor Competence Preservation Learners" in the proceedings of the 27th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2015, Vietri sul Mare, Italy, November 9–11, 2015, pp. 943–949.

* Corresponding author.
E-mail addresses: fabrizio.angiulli@unical.it (F. Angiulli), e.narvaez@dimes.unical.it (E. Narvaez).

https://doi.org/10.1016/j.neucom.2018.04.017
0925-2312/

We then introduce some subset selection strategies, namely PACOPT, MAXOPT, and TRNOPT, intended to select the most promising subset according to different ways of estimating expected accuracy. Among them, the PACOPT strategy is based on PEE principle and estimates generalization as a trade-off between training set accuracy and model complexity.

Moreover, a variant of the PACOPT strategy, called aPACOPT for approximate-PACOPT, is described. The technique attempts to reduce time complexity by early terminating the learning phase on the basis of the current trend of the pessimistic accuracy estimate curve.

Before going into the details, next we summarize the contributions of this research:

- As the first major result, we show that the PACOPT selection strategy guarantees to preserve the accuracy of the consistent subset with a larger reduction factor, since on the average the subset selected by PACOPT contains the 30% of the training set consistent subset objects.
- As the second major result, we show that a sensible, on the average of the 2%, generalization improvement can be obtained by using a reduced subset of intermediate size, consisting on the average of the 63% of the training set consistent subset objects.
- Moreover, the aPACOPT strategy allows to compute approximatively the same model determined by the PACOPT strategy, but with sensibly smaller time requirements (the execution time of aPACOPT corresponds to the 25% – 50% of the time required by PACOPT).
- Comparison with state-of-the-art prototype selection methods highlight that the aPACOPT strategy is able to obtain a model of size comparable to that obtained by the best prototype selection methods in terms of reduction ratio, with far smaller time requirements, corresponding to 4 orders of magnitude on medium-sized datasets.

The rest of the work is organized as follows. Section 2 discusses scenario and related works. Section 3 describes the α-FCNN algorithm for computing a soft consistent training set subset. Section 4 describes the four selection strategies for reducing training set complexity, namely TRNOPT, PACOPT, MAXOPT, and aPACOPT. Section 5 illustrates experimental results, including accuracy, scalability, and comparison with state-of-the-art related methods. Section 6 concludes the work.

2. Related work

Errors committed by classification models [10] are generally divided into two types: *training errors* and *generalization errors*. The training error, also known as resubstitution error, is the number of misclassification errors committed on training records, whereas the generalization error is the expected error of the model on previously unseen records. A good model must have low training error as well as low generalization error. This is important because a model that fits the training data too well can have a poorer generalization error than a model with a higher training error, this phenomenon is known as model *overfitting*. The model overfitting problem has been investigated by several authors including [11–13].

The chance for model overfitting increases as the model becomes more complex. For this reason, might be preferred simpler models, a strategy that agrees with a well-known principle known as Occam's razor: given two models with the same generalization error, the simpler model is preferred over the more complex model [9].

Thus, overfitting happens when a model is more flexible than it needs to be and incorporates noise in the training data to the extent that it negatively impacts the performance on the model on new data [14]. There are several possible causes why overfitting happens: the presence of noise, a model too complex, a small training set, a very rich hypothesis space, a domain with many features [9].

A way to help learning algorithms to select the most appropriate model, is to be able to estimate the generalization error. The right complexity is that of a model that produces the lowest generalization error. The problem is that the learning algorithm has access only to the training set during model building. It has no knowledge of the test set, and thus, does not know how well the model will perform on records it has never seen.

Decision trees are one of the most common classification techniques used in the practice. A decision tree is a hierarchical structure consisting of nodes and directed edges. A problem common when a decision tree is built is that many of the branches will reflect anomalies in the training data due to noise or outliers. As the number of nodes in the decision tree increases, the tree will have fewer training errors. Up to a certain size also the test error will decrease. However, once the tree becomes too large, its test error rate begins to increase even though its training error rate continues to decrease, due to overfitting.

Two well-known techniques for incorporating model complexity into the evaluation of classification models are PEE and MDL, which are described next in the context of decision trees.

The *Pessimistic error estimate* (PEE) approach explicitly computes generalization error as the sum of training error and a penalty term for model complexity. The resulting generalization error can be considered its pessimistic error estimate. Let $n(t)$ be the number of training records classified by node t and $e(t)$ be the number of misclassified records. The pessimistic error estimate of a decision tree T, $e_g(T)$, can be computed as follows:

$$e_g(T) = \frac{\sum_{i=1}^{k}[e(t_i) + \Omega(t_i)]}{\sum_{i=1}^{k} n(t_i)} = \frac{e(T) + \Omega(t)}{N(t)},$$

where k is the number of leaf nodes, $e(T)$ is the overall training error of the decision tree, N_t is the number of training records, and $\Omega(t_i)$ is the penalty term associated with each node t_i.

The *Minimum Description Length* (MDL) principle is a general method for inductive inference, based on the idea that the more we are able to compress a set of data, the more regularities we have found in it [15]. Is based on an information-theoretic approach, when two models fit the data equally well, MDL will choose the one that is the simplest in the sense that it allows for a shorter description of the data [9]. The MDL principle involves adding to the error function an extra term that is designed to penalize mappings that are not smooth [16,17]. MDL principle use encoding techniques to define the best decision tree as the one that requires the fewest number of bits to both (1) encode the tree and (2) encode the exceptions to the tree.

There are two common approaches to tree pruning that exploit the above mentioned principles, that are *pre-pruning* and *post-pruning* [18–25]. The first approach stops growing the tree earlier, before it perfectly classifies the training set. Upon halting, the node becomes a leaf. The leaf may hold the most frequent class among the subset tuples or the probability distribution of those tuples. The second approach, post-pruning, removes subtrees from a fully grown tree. A subtree at a given node is pruned by removing its branches and replacing it with a leaf. The leaf is labeled with the most frequent class among the subtree being replaced.

Reducing model complexity is transversal to many other learning scenarios, as in the case of ensemble pruning [26], instance selection for time-series prediction [27], resampling for improving classifier performance [28], and unsupervised feature selection [29].

Please cite this article as: F. Angiulli, E. Narvaez, Pruning strategies for nearest neighbor competence preservation learners, Neurocomputing (2018), https://doi.org/10.1016/j.neucom.2018.04.017

In order to reduce the complexity of nearest neighbor classifiers, three categories of techniques exist for the selective storage of instances, i.e. *competence enhancement, competence preservation*, and *hybrid approaches*. Competence enhancement methods remove noisy or corrupted instances such that classification accuracy is improved. Competence preservation techniques remove superfluous instances such that performance is improved while classification accuracy of the training set is preserved. Finally hybrid approaches perform both competence enhancement and competence preservation [30].

Among the competence enhancement techniques there are the Edited Nearest Neighbor (ENN) algorithm [31] and its extension Repeated-ENN (RENN) [32], which remove instances considered to be noise from the training set: an instance is considered as noise if it does not agree with its k nearest neighbors.

The seminal work in competence preservation is the Condensed Nearest Neighbor (CNN) [4]. The basic idea is to remove the data points that do not add additional information and show similarity with other training points. A consistent subset is a subset of the training set having the property of allowing correct classification of the whole training set objects through the use of the nearest neighbor rule. One of the shortcomings of the CNN rule is that it may select points far from the decision boundary.

Gates [33] introduced the Reduced Nearest Neighbor Rule (RNN) as an improvement over CNN. This algorithm is able to eliminate noisy instances and internal instances while retaining border points. From the perspective of temporal cost, it is more expensive than the CNN rule. It will always produce a subset of the result of CNN algorithm.

The Selective Nearest Neighbor Rule (SNN) devised by Ritter et al. [34] improves on the CNN and RNN. This algorithm produces a selective subset of the original data so that (1) the subset is consistent, (2) the distance between any sample and its nearest selective neighbor is less than the distance from the sample to any sample of the other class, and (3) the subset is the smallest possible.

The Modified Condensed Nearest-Neighbor rule [7] (MCNN) builds a training set in an incremental manner. This algorithm is based on the misclassified samples: a representative prototype of each class is determined and added to the set of basic prototypes to classify these patterns correctly, until there are no misclassified points in training set. The MCNN algorithm may work well in the case of patterns with a Gaussian distribution with an equal and diagonal covariance matrix. In general, it might require a lot of iterations to converge.

Aha et al. [35], introduced a series of instance-based learning algorithms to reduce storage requirements and tolerate noisy instances. Other researchers have provided variations on these algorithms [36,37]. Wilson and Martinez [38] introduced three new instance reduction techniques that are intuitive and provide good storage reduction called RT1, RT2 and RT3. RT3, has shown higher generalization accuracy and lower storage requirements. Moreover, in [5] a series of six algorithms for training set reduction based on the nearest neighbor algorithm, called DROP1, DROP2, DROP3, DROP4, DROP5 and DEL, are described.

Accoding to the extensive taxonomy proposed in [39], CCIS, SSMA, and RMHC are remarkable methods belonging to the hybrid family.

The Class Conditional Instance Selection (CCIS) [40] is a large margin-based algorithm for instance selection, where the effect of eliminating one instance on the hypothesis margin of other instances is measured by the concept of class conditional nearest neighbor. The hypothesis margin is defined as the distance between the hypothesis and the closest hypothesis that assigns an alternative label to the given instance. The proposed instance selection method can be interpreted as a novel large-margin-based procedure for training Voronoi networks.

The Steady-State MA (SSMA) [41] is a model of memetic algorithm for instance selection, that incorporates an ad-hoc local search specifically designed for optimizing the properties of the prototype selection problem with the aim of tackling the scaling up problem. Memetic algorithms combine evolutionary algorithms and local search within the evolutionary cycle.

The Random Mutation Hill Climbing Algorithm (RMHC) [42] technique uses the wrapper approach for feature selection and for decreasing the number of prototypes stored by instance-based methods. The general approach is to use a bit string to represent a set of prototypes, and in some experiments, a collection of features. The intuitive search mechanism is that the mutation of the bit vector changes the selection of instances in the prototype set or toggles the inclusion or exclusion of a feature from the nearest neighbor computation.

Other recent condensation algorithms that will be compared with the approach here proposed are a prototype selection method, TRKNN [43], a cluster-based learning method, PSC [44], and a prototype invention method, BNNT [45].

The Template Reduction for KNN (TRKNN) prototype reduction algorithm [43] defines the concept of chain of the nearest neighbors and selects the condensed set on the basis of a cut-off value for the distances among alternating adjacent examples of the chain. The Prototype Selection by Clustering (PSC) algorithm [44] belongs to the category of cluster-based learning algorithms. Specifically, after splitting the training set into c clusters, PSC selects border prototypes and some interior prototypes. The Binary NN Tree (BNNT) algorithm [45] belongs to the class of prototype invention algorithms, in which prototypes can be obtained as, e.g., weighted averages of training set examples. BNNT builds nearest-neighbors trees involving training set examples and then selects or generates prototypes from each tree on the basis of the labels associated with the examples occurring therein. The parameter *tree_level* is required to stop tree growing if the number of nodes reaches this threshold.

3. The α-FCNN algorithm

The α-FCNN algorithm is here introduced to provide a procedure to compute a soft consistent training set subset for the nearest neighbor rule. The basic idea is to relax the notion of consistency, in order to allow the selected subset to score an error rate at most α on the training set objects with, the aim to avoid overfitting. This is obtained by early stopping subset growing, thus disregarding the objects that will be selected in the last part of the selection procedure.

Selection strategies introduced in the subsequent section serve then the purpose to determining the right value of $1-\alpha$ guaranteeing the best trade-off between model size and the associated generalization error (Algorithm 1).

We start by giving some preliminary definitions.

In the following we denote by T a labeled training set from a metric space with distance metrics d.

We denote by $nn(p, T)$ the nearest neighbor of p in T according to the distance d. If $p \in T$, then p itself is its nearest neighbor. We denote by $l(p)$ the label associated with p.

Given a point q, the NN rule $NN(q, T)$ assigns to q the label of the nearest neighbor of q in T, i.e. $NN(q, T) = l(nn(q, T))$.

A subset S of T is said to be a *training set consistent subset of* T if, for each $p \in (T - S)$, $l(p) = NN(p, S)$.

Now we relax the notion of training set consistent subset.

Definition 1. Let α be a real value in [0,1], also called *consistency fraction*. A subset S of T is said to be *training set α-consistent subset*

Algorithm 1: The α-FCNN rule.

Input: A training set T, a consistency threshold α;
Output: A training set α-consistent subset S of T;
Method:
$i = 0$;
$S_0 = \emptyset$;
$\ell^* = \arg\max_\ell |\{p : l(p) = \ell\}|$;
$\Delta S_0 = Centroids(\{p : l(p) = \ell^*\})$;
$\alpha_0 = |\{p : l(p) = \ell^*\}|/|T|$;
while $(\alpha_i < \alpha)$ **do**
  $i = i + 1$;
  $S_i = S_{i-1} \cup \Delta S_{i-1}$;
  $\Delta S_i = \emptyset$;
  $e_i = 0$;
  $err_{\max} = 0$;
  foreach $(p \in S)$ **do**
    $err = |Voren(p, S, T)|$;
    if $(err > err_{\max})$ **then**
      $p^* = p$;
      $err_{\max} = err$;
    $e_i = e_i + err$;
  $\Delta S_i = \{rep(p^*, Voren(p^*, S, T))\}$;
  $\alpha_i = 1 - e_i/|T|$;
return(S_i);

of T if it correctly classifies at least the α fraction of the objects in T, that is to say if

$$\frac{|\{p \in T : l(p) = NN(p, S)\}|}{|T|} \geq \alpha$$

□

Thus, a training set consistent subset corresponds to a training set 1-consistent subset, and vice versa.

The α-FCNN algorithm (for α-consistent Fast Condensed Nearest Neighbor rule) is a specialization of the FCNN algorithm [8,46] for computing an α-consistent subset.

Let S be a subset of T and let p be an element of S. We denote by $Vor(p, S, T)$ the set $\{q \in T \mid p = nn(q, S)\}$, that is the set of the elements of T that are closer to p than to any other element p' of S. Furthermore, we denote by $Voren(p, S, T)$ the set of the *Voronoi enemies* of p in T w.r.t. S, defined as $\{q \in Vor(p, S, T) | l(q) \neq l(p)\}$.

We denote by $Centroids(X)$ the set containing the centroids[1] of of the objects in X.

The following Theorem states the property exploited by the FCNN rule in order to compute a training set consistent subset.

Theorem 1. *S is a training set α-consistent subset of T for the nearest neighbor rule if and only if*

$$\left(1 - \frac{1}{|T|} \sum_{p \in S} |Voren(p, S, T)|\right) \geq \alpha$$

Proof. The elements of $Voren(p, S, T)$ are precisely the misclassified objects belonging to the Voronoi cell induced by the object p. Thus, the summation above corresponds to the number of training examples that are misclassified by using S as reference set for the nearest neighbor rule, while the left hand side represents the accuracy of the subset S. The result then follows by noticing that the accuracy is not smaller than α. □

The algorithm α-FCNN starts by selecting the centroid of the most populated class.

Then it works in an incremental manner: during each iteration the set S is augmented with ΔS until the stop condition, given by Theorem 1, is reached.

The set ΔS is composed of one single object per iteration. Specifically, ΔS is built by selecting $rep(p^*, Voren(p^*, S, T))$, that is the representative of the Voronoi enemies of one of the elements of S. Such an element can be defined as the object p^* of S such that $|Voren(p^*, S, T)|$ is maximum, that is, the object inducing the Voronoi cell containing the maximum number of misclassified points.

The representative $rep(p, X)$ of X w.r.t. p is defined as the class centroid in X closest to p, that is $rep(p, X) = nn(p, Centroids(X))$ (hence α-FCNN selects $nn(p^*, Centroids(Voren(p^*, S, T)))$).

The learning curve of the method corresponds to the curve of the training set accuracy versus the current subset size. To illustrate the advantages of α-FCNN, we note that the learning curve of FCNN grows rapidly during the first iterations reaching high accuracy in a short time. Subsequently, the growth rate decreases and accuracy varies much less rapidly. Fig. 1 shows a binary classification example. The red class consists of the points inside the circle centered in the origin having radius $r = (2\pi)^{-1/2}$, while the blue class consists of the points in $[-0.5, +0.5]^2$ that are outside the circle (see Fig. 1a). After only 10 iterations (see Fig. 1b) the method has achieved good accuracy, equal to about 90%. After another 40 iterations (see Fig. 1c) the accuracy became more satisfactory (about 96%), but growth was slower. Approximately 300 iterations are needed to obtain a (consistent) subset (see Fig. 1c) achieving 100% accuracy. By allowing to select an α-consistent subset, the α-FCNN method can return one of these intermediate subsets, disregarding the prototypes contributing less to the accuracy of the model and avoiding overfitting. The α-FCNN exploits different selection strategies to determine the value of α providing the best trade-off between model size and generalization. The description of the strategies is precisely the topic of the next section.

4. Selection strategies

In this section, we describe the strategies we have designed in order to select the FCNN condensed set associated with the best generalization power.

4.1. The TRNOPT selection strategy

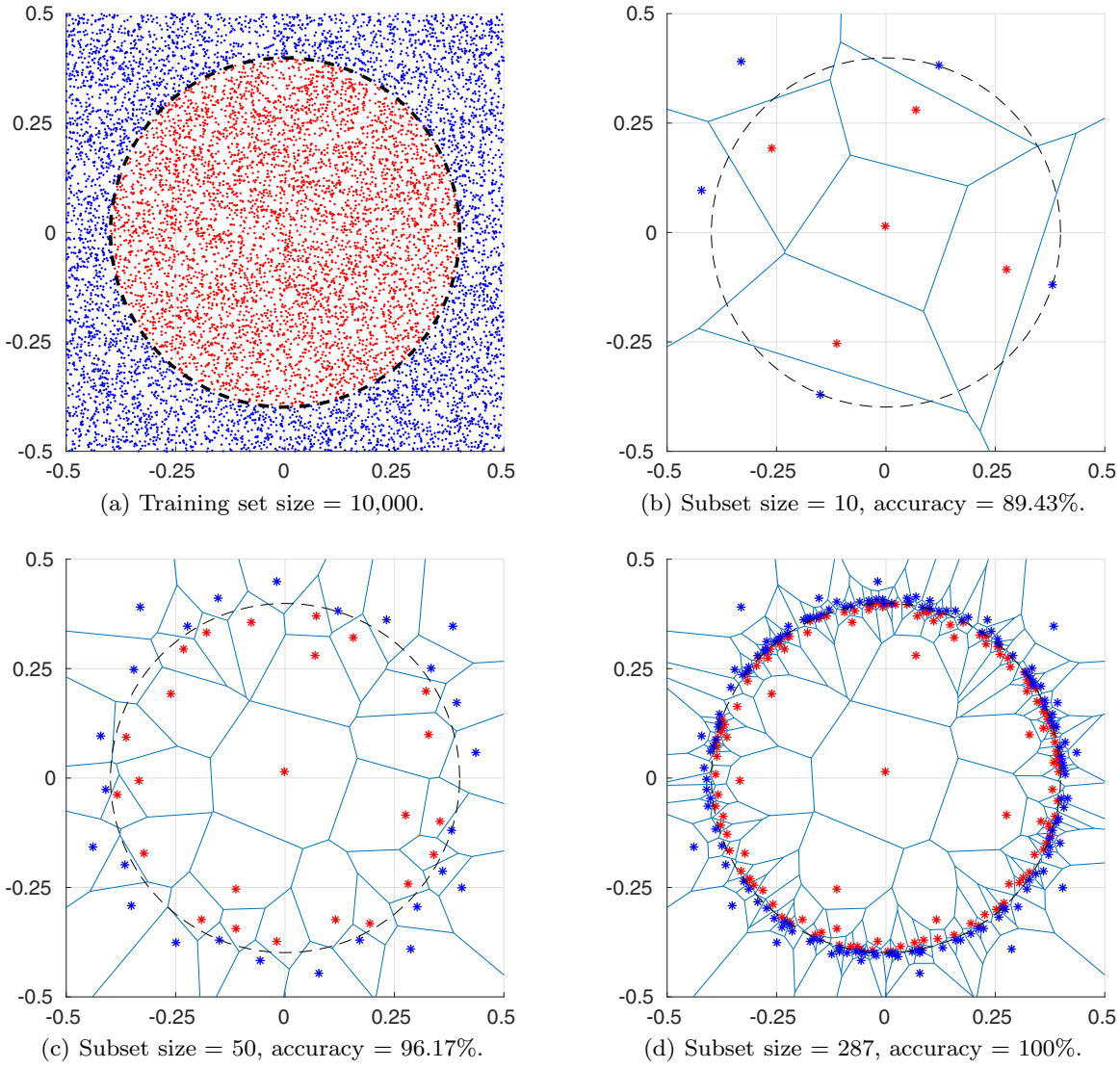
According to the first technique, named TRNOPT (for *Optimal Training Accuracy* estimate), the accuracy estimate corresponds to the training accuracy. This means that the method selects as the optimal subset the (1−)consistent subset S of the training set. With this aim we run the α-FCNN algorithm with $\alpha = 1$ and then use the computed subset as the classification model.

4.2. The PACOPT selection strategy

Prepruning is a standard technique for handling noise and preventing overfitting in decision tree learning. Prepruning heuristics are used to reduce (not entirely eliminate) the amount of overfitting, so that both learning and model building will be more efficient. The idea of the second strategy is similar to the approach of prepruning used during decision trees growth.

The *pessimistic error estimate* approach [9] computes generalization error of a classifier C as the sum of the associated training

[1] In the Euclidean space, the centroid of a class is the object of the class closest to the geometric center of the class. In a metric space, it can be defined as the object of the class minimizing the sum of distances to any other object of the same class.



(a) Training set size = 10,000. (b) Subset size = 10, accuracy = 89.43%.

(c) Subset size = 50, accuracy = 96.17%. (d) Subset size = 287, accuracy = 100%.

Fig. 1. [Best viewed in color] Classification example (training set in fig. (a)) showing the difference between the training set consistent subset (fig. (d)) and the α-consistent subsets (fig. (b) for $\alpha \approx 0.9$ and fig. (c) for $\alpha \approx 0.95$).

error and of a *penalty* term. The latter term intends to take into account model complexity. The resulting error $e_{pess}(C)$, which can be considered the pessimistic error estimate of the generalization error of C, is hence computed as follows:

$$e_{pess}(C) = \frac{e(C) + \Omega(C)}{N_{train}},$$

where $e(C)$ is the number of misclassified training examples (or absolute training error) of the classifier C, N_{train} is the number of training examples, and $\Omega(C)$ is the penalty term associated with C.

In the case of decision trees, $\Omega(C)$ can be computed as $N_{leafs}(C) \cdot p$, where $N_{leafs}(C)$ is the number of leafs of the tree C and p is a penalty factor, that is to say a constant accounting for the complexity cost associated with a single leaf node. Hence, $p = 0.5$ means that a node should always be expanded into its two child nodes as long as it improves the classification of at least one training record, because expanding a node, which is equivalent to adding 0.5 to the overall error, is less costly than committing one training error. For example, if $e(C) = 4$, $N_{train} = 24$, $N_{leafs}(C) = 7$, and $p = 0.5$, we obtain

$$e_{pess}(C) = \frac{4 + 7 \cdot 0.5}{24} = 0.3125$$

Suppose now that by expanding a leaf node of C we obtain the novel tree C_{new} such that $e(C_{new}) = 2$, then

$$e_{pess}(C_{new}) = \frac{2 + 8 \cdot 0.5}{24} = 0.25$$

Since $e_{pess}(C_{new}) < e_{pess}(C)$ we can conclude that C_{new} is preferable to C.

The PACOPT (for *Optimal Pessimistic Accuracy* estimate) strategy incorporates model complexity into the evaluation of the training set α-consistent subset by exploiting pessimistic error estimate. Specifically, let α_i the training accuracy of the training set consistent subset S_i determined at the beginning of the i-th iteration of the α-FCNN algorithm. According to the PACOPT criterion, the selected subset S^* is the one maximizing the pessimistic accuracy estimate determined as follows:

$$\begin{aligned} &1 - pessimistic\_error\_estimate \\ &= 1 - \frac{misclassified\_examples + penalty}{total\_examples} \\ &= 1 - \frac{(1-\alpha_i)|T| + p \cdot |S_i|}{|T|} \end{aligned}$$

Please cite this article as: F. Angiulli, E. Narvaez, Pruning strategies for nearest neighbor competence preservation learners, Neurocomputing (2018), https://doi.org/10.1016/j.neucom.2018.04.017

$$= \alpha_i - p \cdot \frac{|S_i|}{|T|},$$

where the penalty term associated with the subset S_i is $p \cdot |S_i|$.

Differently from decision trees, here we assume that the penalty term is given by $p \cdot |S|$, the product of the penalty factor by the number of elements the α-consistent subset, since for the nearest neighbor decision rule all the reference objects play the same role (loosely speaking, they can be regarded as the leafs of a "flat" tree).

Thus,

$$S^* = \arg\max_{S_i} \left\{ \alpha_i - p \cdot \frac{|S_i|}{|T|} \right\}$$

The penalty factor p intends to model the cost of including another object in the subset S_i. This corresponds to say that the subset S_j (having training accuracy α_j) is preferable to S_i (having training accuracy $\alpha_i \leq \alpha_j$) provided that:

$$\begin{aligned}
&\alpha_i - p \cdot \frac{|S_i|}{|T|} < \alpha_j - p \cdot \frac{|S_j|}{|T|} \\
&\iff \alpha_i - p \cdot \frac{|S_i|}{|T|} < \alpha_j - p \cdot \frac{|S_i| + |S_j - S_i|}{|T|} \\
&\iff \alpha_i < \alpha_j - p \cdot \frac{|S_j - S_i|}{|T|} \\
&\iff (\alpha_j - \alpha_i) \cdot |T| > p \cdot |S_j - S_i|
\end{aligned}$$

Intuitively, this means that the methods prefers a larger subset provided that the number of further correctly classified examples overcomes by a factor p the number of additional examples to be included in the model.

Applying the PACOPT criterion does not require additional asymptotic operations and, hence, its cost is the same of the TRNOPT criterion.

4.3. The MAXOPT selection strategy

The MAXOPT strategy (for *Optimal Cross Validation Accuracy* estimate) exploits cross validation in order to select the model showing the best generalization performance.

With this aim, the parameter α is varied within a suitable interval $[\alpha_{\min}, 1]$ by considering values $\alpha_1 = \alpha_{\min}, \alpha_2, \ldots, \alpha_m = 1$ ($m > 1$). For each α_j ($1 \leq j \leq m$) a stratified ten-fold cross validation is performed in order to determine the average accuracy associated with the training set α_j-consistent subset. The best consistency fraction α^* is the one associated with the maximum average cross validation accuracy. The optimal model S^* for MAXOPT is eventually obtained as the α^*-consistent subset extracted from the whole training set.

4.4. The aPACOPT selection strategy

The PACOPT strategy aims at improving model size, while preserving or even improving accuracy, over the TRNOPT strategy, but presents the same temporal cost due the need of computing the whole pessimistic accuracy estimate curve. aPACOPT, for approximate-PACOPT, is a variant of the PACOPT strategy. The technique attempts to reduce time complexity by early terminating the learning phase on the basis of the current trend of the pessimistic accuracy estimate curve.

The termination condition of aPACOPT assumes that the above curve is monotone non-decreasing before the global maximum and monotone non-increasing after the global maximum. However, the design of the termination condition is made difficult by the fact that the learning curve is not regular (i.e. monotone), but instead presents a lot of local fluctuations. In order to make the termination condition robust to such a fluctuations, the termination will be based on the behavior of the curve within a window of size w.

Specifically, let $t = 1, 2, \ldots, |S|$ denote the current iteration of the algorithm, also referred to as time t in the following. The window $W^{(t)}$ at time $t \geq w$ is the pair $\langle X^{(t)}, Y^{(t)} \rangle$, where $X^{(t)}$ is a vector of w elements (or timestamps):

$$X^{(t)} = (X_1^{(t)} = t - w + 1,\ X_2^{(t)} = t - w + 2, \ldots, X_w^{(t)} = t),$$

representing the time interval $[t - w + 1, t]$ and $Y^{(t)} = (Y_1, \ldots, Y_w)$ is a vector of w elements representing the pessimistic accuracy estimate:

$$Y_i^{(t)} = \alpha_{t-w+i} - p \cdot \frac{|S_{t-w+i}|}{|T|},$$

within time interval $[t - w + 1, t]$. The vector $\hat{Y}^{(t)}$ consists of the $t - w$ accuracy estimates

$$\hat{Y}^{(t)} = \left(\alpha_1 - p \cdot \frac{|S_1|}{|T|},\ \alpha_2 - p \cdot \frac{|S_2|}{|T|}, \ldots, \alpha_{t-w} - p \cdot \frac{|S_{t-w}|}{|T|} \right)$$

pertaining to all the windows that precede the current one.

The current window can be exploited to approximate the slope tangent line to the the pessimistic accuracy curve at time t by computing the intercept β_t of the regression line for the data points $\{(X_i^{(t)}, Y_i^t), i = 1, \ldots, w\}$, as follows:

$$\beta_t = \frac{\sum_{i=1}^{w} \left[(X_i^{(t)} - \overline{X}^{(t)})(Y_i^{(t)} - \overline{Y}^{(t)}) \right]}{\sum_{i=1}^{w} (X_i^{(t)} - \overline{X}^{(t)})},$$

where $\overline{X}^{(t)}$ ($\overline{Y}^{(t)}$, resp.) denotes the mean of the elements in $X^{(t)}$ ($Y^{(t)}$, resp.).

Let us define two boolean variables, that are

$$\varphi_t = \begin{cases} \textbf{true}, & \text{if } \beta_t < \epsilon \\ \textbf{false}, & \text{otherwise} \end{cases}$$

representing the fact that the slope of curve tangent is non-increasing (where ϵ is a small positive value, e.g. $\epsilon = 10^{-3}$, intended to capture errors associated with the estimation procedure), and

$$\psi_t = \begin{cases} \textbf{true}, & \text{if } \max(Y^{(t)}) < \max(\hat{Y}^{(t)}) \\ \textbf{false}, & \text{otherwise} \end{cases}$$

representing the fact that the global maximum has not been observed in the current window.

Now we are in the position of providing the termination condition of the aPACOPT strategy. The methods stops if the following condition holds:

$$STOP_t \equiv \left(\bigwedge_{i=1}^{w} \varphi_{t-w+i} \right) \wedge \left(\bigwedge_{i=1}^{w} \psi_{t-w+i} \right),$$

which can be interpreted as the fact that the slope of curve has been non-increasing and no new local maximum has been observed during an entire window.

At the time t at which aPACOPT decides to stop, the best pessimistic accuracy estimate $\tilde{\alpha}^*$ so far encountered is determined and the associate subset is returned as the solution. It can be concluded that the subset returned by aPACOPT cannot be greater than those computed by PACOPT, since two cases are possible: (1) $\tilde{\alpha}^* < \alpha^*$, and this means that the algorithm stopped before reaching the maximum α^*, which is associated with a larger subset; or (2) $\tilde{\alpha}^* = \alpha^*$, and this means that the algorithm stopped after reaching the maximum and, hence, selected the subset associated with α^*.

Please cite this article as: F. Angiulli, E. Narvaez, Pruning strategies for nearest neighbor competence preservation learners, Neurocomputing (2018), https://doi.org/10.1016/j.neucom.2018.04.017

Table 1
Dataset characteristics.

Data set name	*Abbrev.*	*Size*	*Features*	*Classes*
Bupa	BUP	345	6	2
Coil 2000	COI	4,992	85	2
Colon Tumor	COL	62	2000	2
Echocardiogram	ECH	61	11	2
Ionosphere	ION	351	34	2
Iris	IRI	150	4	3
Image Segmentation	IMA	2310	19	7
Pen Digits	PEN	7494	16	10
Pima Indians Diabetes	PIM	768	8	2
Satellite Image	SAT	6435	36	6
Spam Database	SPA	4207	57	2
SPECT Heart Data	SPE	349	44	2
Vehicle	VEH	846	18	4
Wisconsin Breast Cancer	WBC	683	9	2
Wine	WIN	178	13	3
Wisc. Progn. Breast Cancer	WPB	198	33	2

5. Experimental results

In this section, we present experimental results involving the introduced techniques.

A number of training sets are from the UCI Machine Learning Repository,[2] whose characteristics are summarized in Table 1 where, for each data set, the name, abbreviation, size, features, and number of classes are given. Moreover, we employed also the following datasets.

The *MIST* dataset consists of binary images of handwritten digits (60,000 objects, 784 features, and 10 classes). This dataset is commonly used for training various image processing systems. All digit images have been size-normalized and centered in a fixed size image of 28×28 pixels. In the original dataset each pixel of the image is represented by a value between 0 and 255, where 0 is black, 255 is white and anything in between is a different shade of grey.

The *Forest Cover Type* dataset (*Covtype*, for short, 581052 objects, 54 features, and 7 classes). contains tree observations from four wilderness areas of the Roosevelt National Forest in Colorado. All observations are cartographic variables (no remote sensing) from 30×30 meter sections.

The *Gisette* dataset ($n = 6000$, $d = 5000$, and 2 classes) was constructed from the*MIST* data. The original data were modified for the purpose of the feature selection challenge. Pixels were sampled at random in the middle top part of the feature containing the information necessary to disambiguate digit 4 from digit 9 and higher order features were created as products of these pixels to plunge the problem in a higher dimensional feature space. Also a number of distractor features, called probes, having no predictive power were added. There are 2500 real features and 2500 probes. The order of the features and patterns were randomized.

The *Madelon* dataset[3] ($n = 2000$, $d = 500$, and 2 classes) is an artificial dataset containing data points grouped in 32 clusters placed on the vertices of a five dimensional hypercube and randomly labeled $+1$ or -1. The five dimensions constitute 5 informative features. 15 linear combinations of those features were added to form a set of 20 (redundant) informative features. Based on those 20 features one must separate the examples into the 2 classes. Also a number of distractor features, called probes, having no predictive power were added. The order of the features and patterns were randomized.

The *Census House* dataset[4] ($n = 22784$ and $d = 139$) has been constructed from the 1990 US Census. Attributes are mostly counts cumulated at different survey levels. These are all concerned with predicting the median price of the house in the region based on demographic composition and a state of housing market in the region.

The *MIT Face* dataset[5] ($n = 31022$, $d = 362$, and 2 classes) from the Center for Biological and Computational Learning at MIT (CBCL) consists of images (19×19 grayscale pixels) of faces and non-faces.

If not otherwise stated, the default value of the penalty parameter is $p = 1$ and the default value of the windows size is $w = 100$. All the experiments were executed on a personal computer based on an Intel Core i7-6700 3.40GHz processor, equipped with 16GB of RAM, and under Linux Operating System.

5.1. Accuracy of the strategies

Here, we study the accuracy and model size of the PACOPT, TRNOPT, and MAXOPT strategies by exploiting the dataset described in Table 1. As for the aPACOPT strategy, since it is tailored on large datasets, we defer its analysis to the following section devoted to the scalability analysis and to the comparison with competitors methods.

In order to assess performances of the strategies and to measure generalization, for PACOPT and TRNOPT ten fold cross validation has been accomplished and the average values over the ten folds are reported, while MAXOPT has been executed on the whole training set by considering the same folds. Thus, in these experiments accuracy associated with MAXOPT represents the best generalization achievable by an α-consistent subset for the nearest neighbor rule.

5.1.1. Accuracy curves

First of all, in order to make clear the behavior of the subset selection strategies, we show in Fig. 2 the accuracy estimated by the three methods on the subsets S_i associated with the various accuracy levels α_i.

As far as TRNOPT is concerned, the estimated accuracy is equal to the training one and, hence, the associated curve is always increasing and reaches its optimum at the $\alpha_{\max} = 1$ accuracy level, which corresponds to the largest subset (the 1-consistent one).

As far as PACOPT is concerned, since its estimated accuracy depends on the trade-off between the training accuracy and the penalty term taking into account subset size, the associated curve is firstly increasing and then can be decreasing if the peak is reached before 100% training accuracy. For this strategy, the optimal subset is determined in correspondence of the estimated accuracy peak. For example, on *Spam* the PACOPT estimate increases up to $|S| = 6\%$ and then decreases, while TRNOPT has its maximum at $|S| \approx 21\%$.

As far as MAXOPT is concerned, in general its curve depends on cross-validation accuracy and no specific trend is guaranteed. Since in the experiments presented we are using cross validation accuracy as a measure of generalization, in this case the optimal subset is that in correspondence of the maximum of the MAXOPT accuracy estimate curve.

5.1.2. Accuracy and model size

Table 2 summarizes the accuracy and model size achieved by the strategies.

The *Size* columns report the relative size of the subset computed by each method (values are expressed in percentage of the

[2] http://archive.ics.uci.edu/ml/.
[3] http://archive.ics.uci.edu/ml/datasets/madelon.
[4] https://www.cs.toronto.edu/~delve/data/census-house/desc.html.
[5] http://cbcl.mit.edu/software-datasets/FaceData2.html.

Please cite this article as: F. Angiulli, E. Narvaez, Pruning strategies for nearest neighbor competence preservation learners, Neurocomputing (2018), https://doi.org/10.1016/j.neucom.2018.04.017

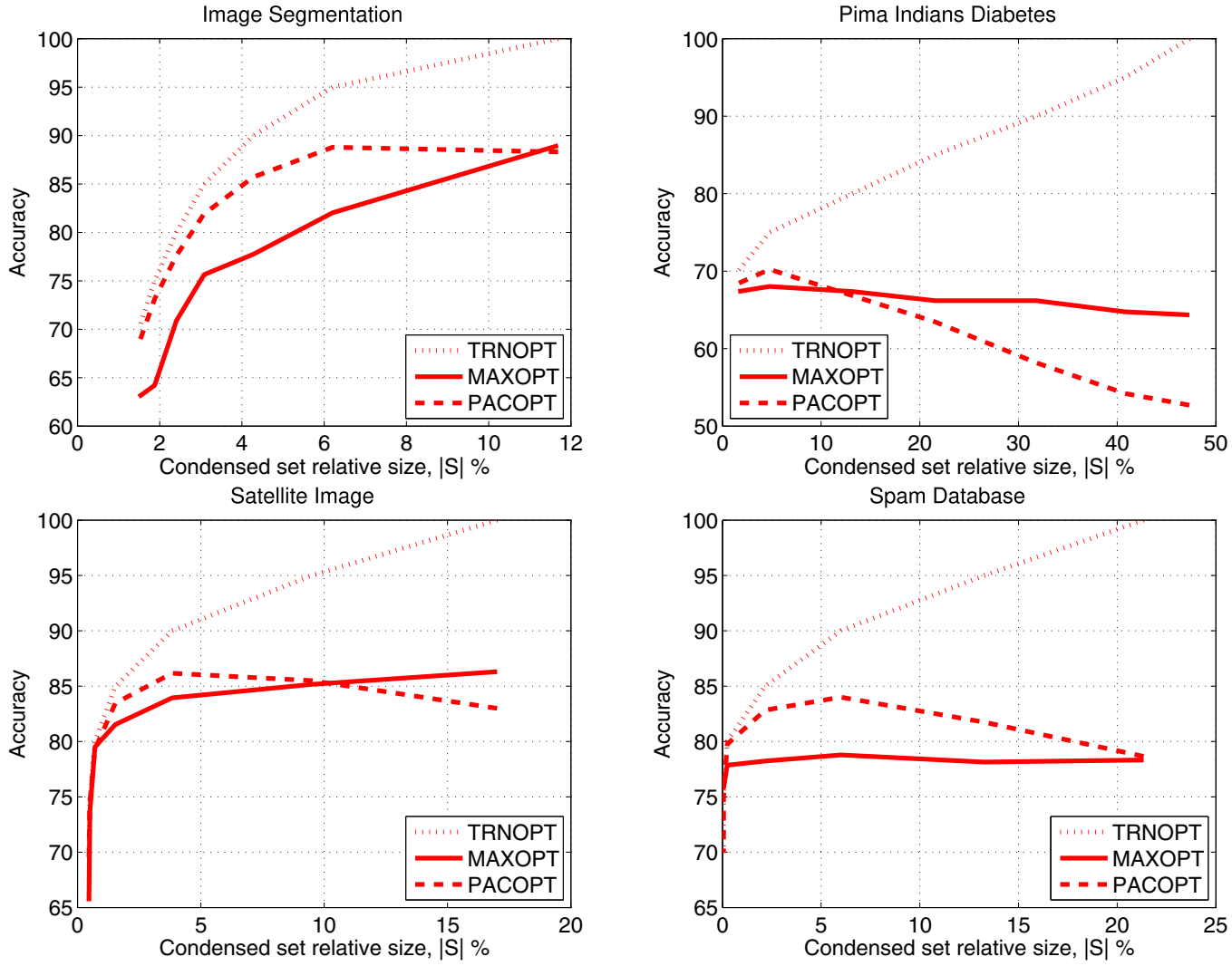


Fig. 2. TRNOPT, PACOPT, and MAXOPT accuracy curves.

Table 2
Accuracy and model size.

	PACOPT				MAXOPT				TRNOPT	
	Acc	*ΔAcc*	*Size*	*ΔSize%*	*Acc*	*ΔAcc*	*Size*	*ΔSize%*	*Acc*	*Size*
BUP	**63.53**	+4.71	6.44	−88.23	**63.53**	+4.71	6.44	−88.23	58.82	54.75
COI	**94.25**	+10.16	0.02	−99.90	**94.25**	+10.16	0.02	−99.90	84.09	20.47
COL	**80.00**	0.00	3.58	−90.01	**81.67**	+1.67	17.92	−50.00	80.00	35.84
ECH	**93.33**	0.00	5.46	−50.04	93.33	0.00	10.93	0.00	93.33	10.93
ION	85.14	−0.29	5.70	−69.98	**87.71**	+2.28	12.03	−36.65	85.43	18.99
IRI	92.00	−2.00	2.96	−71.46	94.00	0.00	10.37	0.00	94.00	10.37
IMA	82.03	−6.93	6.20	−46.96	88.96	0.00	11.69	0.00	88.96	11.69
PEN	**98.42**	0.00	3.71	0.00	98.42	0.00	3.71	0.00	98.42	3.71
PIM	**68.03**	+3.69	4.77	−88.45	**68.03**	+3.69	4.77	−88.45	64.34	41.31
SAT	83.95	−2.36	3.83	−77.48	86.31	0.00	17.01	0.00	86.31	17.01
SPA	**78.79**	+0.46	6.00	−71.88	**78.79**	+0.46	6.00	−71.88	78.33	21.34
SPE	70.29	−16.18	10.19	−65.95	86.47	0.00	29.93	0.00	86.47	29.93
VEH	59.40	−3.76	12.74	−74.80	63.10	0.00	50.56	0.00	63.10	50.56
WBC	**95.74**	+1.92	0.33	−96.17	**95.74**	+1.92	0.33	−96.17	93.82	8.62
WIN	**68.24**	0.00	17.48	−55.56	68.24	0.00	39.33	0.00	68.24	39.33
WPB	**66.32**	+2.11	3.37	−92.67	**74.32**	+10.11	0.56	−98.78	64.21	46.02
μ	*80.70*	*−0.61*	*5.66*	*−70.76*	*83.37*	*+2.06*	*13.59*	*−37.06*	*81.31*	*25.31*
σ	*±12.66*	*±5.47*	*±4.34*	*±24.16*	*±11.92*	*±3.36*	*±14.10*	*±43.48*	*±12.89*	*±16.35*

Please cite this article as: F. Angiulli, E. Narvaez, Pruning strategies for nearest neighbor competence preservation learners, Neurocomputing (2018), https://doi.org/10.1016/j.neucom.2018.04.017

Table 3
p-values of the Wilcoxon test for zero mean of the difference of accuracy of the three strategies.

	PACOPT	TRNOPT
MAXOPT	0.004	0.008
PACOPT	–	0.787

size of the whole dataset; thus, e.g., $Size = 6.44$ for PACOPT on dataset BUP means that the subset is composed of the 6.44% of the dataset objects).

The $\Delta Size$ columns report the variation of the size of the subset computed by PACOPT and MAXOPT methods with respect to the size computed by TRNOPT method. Specifically, values are obtained as follows:

$$\Delta Size\%_{\text{METHOD}} = 100\left(1 - \frac{Size_{\text{METHOD}}}{Size_{\text{TRNOPT}}}\right)$$

The *Acc* columns report the Optimal Pessimistic Accuracy Estimate (PACOPT), Optimal Cross Validation Accuracy (MAXOPT), Optimal Training accuracy (TRNOPT).

As for PACOPT is concerned, accuracies are highlighted in bold when their value is greater or equal than that obtained by TRNOPT.

As for MAXOPT is concerned, due to the experimental design this method scores always the maximum accuracy. Hence, accuracy values are highlighted in bold when they are strictly greater than that obtained by TRNOPT.

The ΔAcc columns report the variation of accuracy of PACOPT and MAXOPT with respect to TRNOPT:

$$\Delta Acc = Acc_{\text{METHOD}} - Acc_{\text{TRNOPT}}$$

5.1.3. Assessing performances

In order to assess performances of the strategies, we performed the Wilcoxon signed rank test for zero mean. It is a test of the hypothesis that the difference $x_i - y_i$ between the pairs of samples in the vectors x and y comes from a distribution whose mean is zero.

Recall that the *p*-value is the probability of obtaining a test statistic value at least as extreme as the one that was actually observed, assuming that the null hypothesis (in our case, that the observed distribution complies with the theoretical one) holds. Before the test is performed, a threshold value is chosen, called the significance level λ of the test, traditionally 5% ($\lambda = 0.05$) or 1% ($\lambda = 0.01$).

In the specific case, the null hypothesis "mean is zero" can be rejected at the λ level provided that the *p*-value is smaller than λ.

Table 3 reports the *p*-values of the Wilcoxon test for zero mean of the difference of accuracy by considering accuracy values reported in Table 2.

As far as MAXOPT is concerned, *p*-values are reported in order to confirm that the size of the sample and the number of wins are large enough to reach a robust significance level. And, indeed, it can be seen that for MAXOPT the null hypothesis can be always rejected at significance level 1%.

As a valuable result, the *p*-value of the test for the PACOPT and TRNOPT strategies is far larger than an acceptable rejecting level. Hence, from the point of view of the guaranteed accuracy the two methods are comparable.

Putting all things together, it can be concluded that *PACOPT guarantees to preserve the accuracy of the training set consistent subset, but with a far smaller subset*. On the average, the reduction factor over the 1-consistent subset is of the 70%. As for MAXOPT, the experiments highlights that by selecting a reduced condensed subset, sensible accuracy improvements can be obtained, on the average the 2%, with an appreciable reduction of the subset, on the average the reduction factor associated with the optimal subset is of the 37%, which corresponds about to half of the reduction factor of PACOPT for the parameter setting here considered.

Table 4
Execution times (seconds).

	PACOPT	MAXOPT
BUP	0.012	3.00
COI	0.480	76.64
COL	0.008	5.32
ECH	0.004	1.32
ION	0.016	2.24
IRI	0.004	1.28
IMA	0.036	11.52
PEN	0.088	31.48
PIM	0.020	4.76
SAT	0.332	88.88
SPA	0.444	98.88
SPE	0.016	3.8
VEH	0.024	9.08
WBC	0.012	1.64
WIN	0.008	1.76
WPB	0.008	2.40

5.1.4. Execution time

Table 4 presents the average time elapsed (in seconds) to complete a run of the method on the datasets above considered. Results highlight that MAXOPT is more demanding in terms of CPU usage than PACOPT, due to the need to process all the different folds.

5.1.5. Comparison with other methods

In this section we present comparison of α-FCNN with three condensation algorithms for the nearest neighbor rule. They are, a prototype selection method, TRKNN [43], a cluster-based learning method, PSC [44], and a prototype invention method, BNNT [45].

To perform the comparison, we exploited the results reported in [45], where TRKNN, PSC, and BNNT have been compared on most of the datasets considered in the previous sections. Table 5 reports the details of the comparison. Results concern the accuracy (columns Acc), the relative size of the subset (columns *Size*), and the execution time (columns *Time*).

According to the discussion reported in [45], the best values of the parameters of TRKNN, PSC, and BNNT are: either 1.2 or 1.4 for the cutoff value of TRKNN; $c \in \{6m, 8m\}$, with m the number of classes, for the clusters number c of PSC; and $tree\_level \in \{n/20, \sqrt{n}\}$ for the tree stop growing threshold of BNNT. Thus, the first row associated with each dataset reports the best value pertaining to the best parameters of each method, while the second row reports the average value associated with the best parameters. Some configurations are missing, since they have not been considered in [45]. As for the α-FCNN algorithm, the first row of each dataset concerns the MAXOPT strategy, while the second row concerns the PACOPT strategy.

In Table 5, the values associated with the best performances are highlighted in bold. From these values it can be concluded that in all cases the α-FCNN pruning strategies guarantee optimal subset size and execution time, while being able to achieve optimal or near-optimal accuracy. The two bottom rows of the table report the average values. Notably, MAXOPT and PACOPT outperform the other algorithms for all the performance parameters. To illustrate, the average relative subset size of PACOPT is about one fifth of that of the best among TRKNN, PSC and BNNT, and that the run time of PACOPT is at least two orders of magnitude smaller than that of the other methods.

Please cite this article as: F. Angiulli, E. Narvaez, Pruning strategies for nearest neighbor competence preservation learners, Neurocomputing (2018), https://doi.org/10.1016/j.neucom.2018.04.017

Table 5
Comparison with the PSC, TRKNN, and BNNT algorithms.

	TRKNN		PSC			BNNT			α-FCNN		
	Acc	*Size*	*Acc*	*Size*	*Time*	*Acc*	*Size*	*Time*	*Acc*	*Size*	*Time*
IRI	–	–	93.05	29.18	0.94	**94.20**	22.50	0.83	94.00	**10.37**	**1.28**
	92.08	58.83	92.72	31.67	0.95	**93.77**	22.08	0.88	92.00	**2.96**	**0.01**
IMA	–	–	82.90	51.19	17.62	87.58	19.58	21.39	**88.96**	**11.69**	**11.52**
	58.49	39.73	81.71	50.09	18.06	**86.87**	19.67	21.75	82.03	**6.20**	**0.04**
PEN	–	–	83.13	28.35	135.16	95.57	16.21	126.34	**98.42**	**3.71**	**31.48**
	80.05	39.53	82.67	28.75	137.54	95.39	17.12	127.39	**98.42**	**3.71**	**0.09**
PIM	66.29	40.64	–	–	–	**70.81**	32.89	–	68.03	**4.77**	**4.76**
	–	–	–	–	–	**70.28**	36.69	–	68.03	**4.77**	**0.02**
SPA	69.14	36.38	69.21	40.11	180.11	69.14	36.38	175.45	**78.79**	**6.00**	**98.88**
	76.54	35.79	68.63	42.21	182.71	68.76	35.36	180.20	**78.79**	**6.00**	**0.44**
WBC	87.27	38.24	–	–	–	90.85	33.35	–	**95.74**	**0.33**	**1.64**
	–	–	–	–	–	89.92	33.12	–	**95.74**	**0.33**	**0.01**
WIN	–	–	52.51	40.73	2.19	55.57	**22.47**	2.32	**68.24**	39.33	**1.76**
	63.23	57.94	51.93	36.52	2.23	55.56	23.17	2.34	**68.24**	**17.48**	**0.01**
μ	74.23	38.42	76.16	37.91	67.20	80.53	26.20	65.27	**84.60**	**10.89**	**24.94**
	74.08	46.36	75.53	37.85	68.30	80.08	26.74	66.51	**83.32**	**5.92**	**0.10**

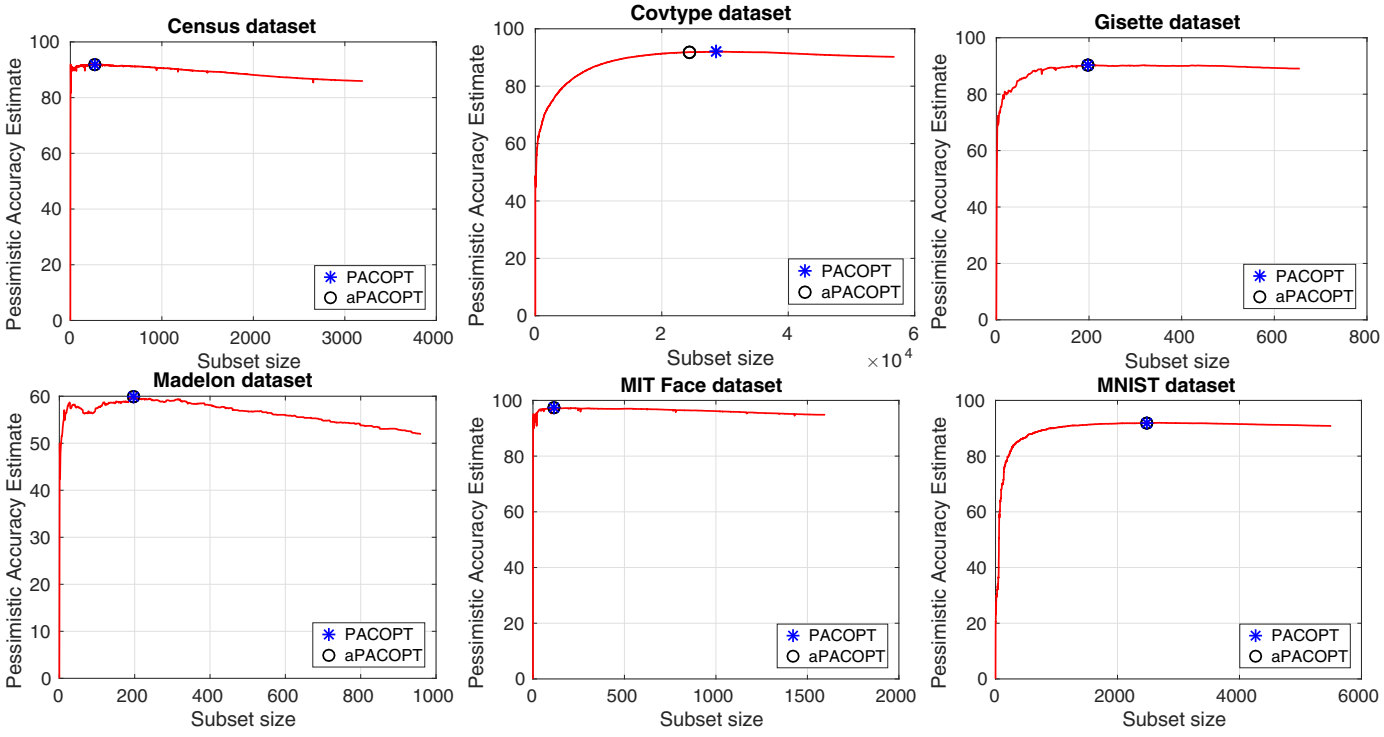


Fig. 3. Pessimistic accuracy estimate curves.

5.2. Analysis of the aPACOPT strategy

The rationale of the aPACOPT strategy is to compute a model comparable to that determined by PACOPT, but with sensibly smaller time requirements. This section is devoted to experimentally substantiating the above claim in the context of large and high-dimensional datasets.

Fig. 3 shows the trend of the Pessimistic Accuracy Estimate of the aPACOPT strategy for the *Census, Covtype, Gisette, Madelon, MIT Face*, and *MNIST* datasets.

It can be noticed that the trend of the curves agrees with the assumption the termination condition of the aPACOPT strategy is designed on, since they resemble concave functions.

As for the subset selected by aPACOPT, it can be seen that in all cases it is of remarkable quality if compared to that selected by PACOPT. For example, on*MIST* the model size of PACOPT and aPACOPT coincide (model size 2485, 4.1% of the dataset size, pessimistic accuracy estimate 91.97%), while on*Covtype* the model of aPACOPT (model size 24327 corresponding to the 4.2% of the dataset size and scoring a pessimistic accuracy estimate of 91.87%) is slightly smaller than that of PACOPT (model size 28651 corresponding to the 4.9% of the dataset size and scoring a pessimistic accuracy estimate of 92.05%).

Differences are justified by the fact that aPACOPT could get stuck in a local maximum since, although from a macroscopic point of view the learning curve appears to be monotone non-decreasing before the global maximum and monotone non-increasing after, from a microscopic point of view it presents local fluctuations.

Notice that the size of the model selected by TRNOPT on the two largest datasets is 56,802 (9.8% of the dataset size) for *Covtype* and 5503 (9.2% of the dataset size) for *MNIST*. Hence, in both cases the size of the model selected by aPACOPT is more than halved with respect to the basic strategy.

Please cite this article as: F. Angiulli, E. Narvaez, Pruning strategies for nearest neighbor competence preservation learners, Neurocomputing (2018), https://doi.org/10.1016/j.neucom.2018.04.017

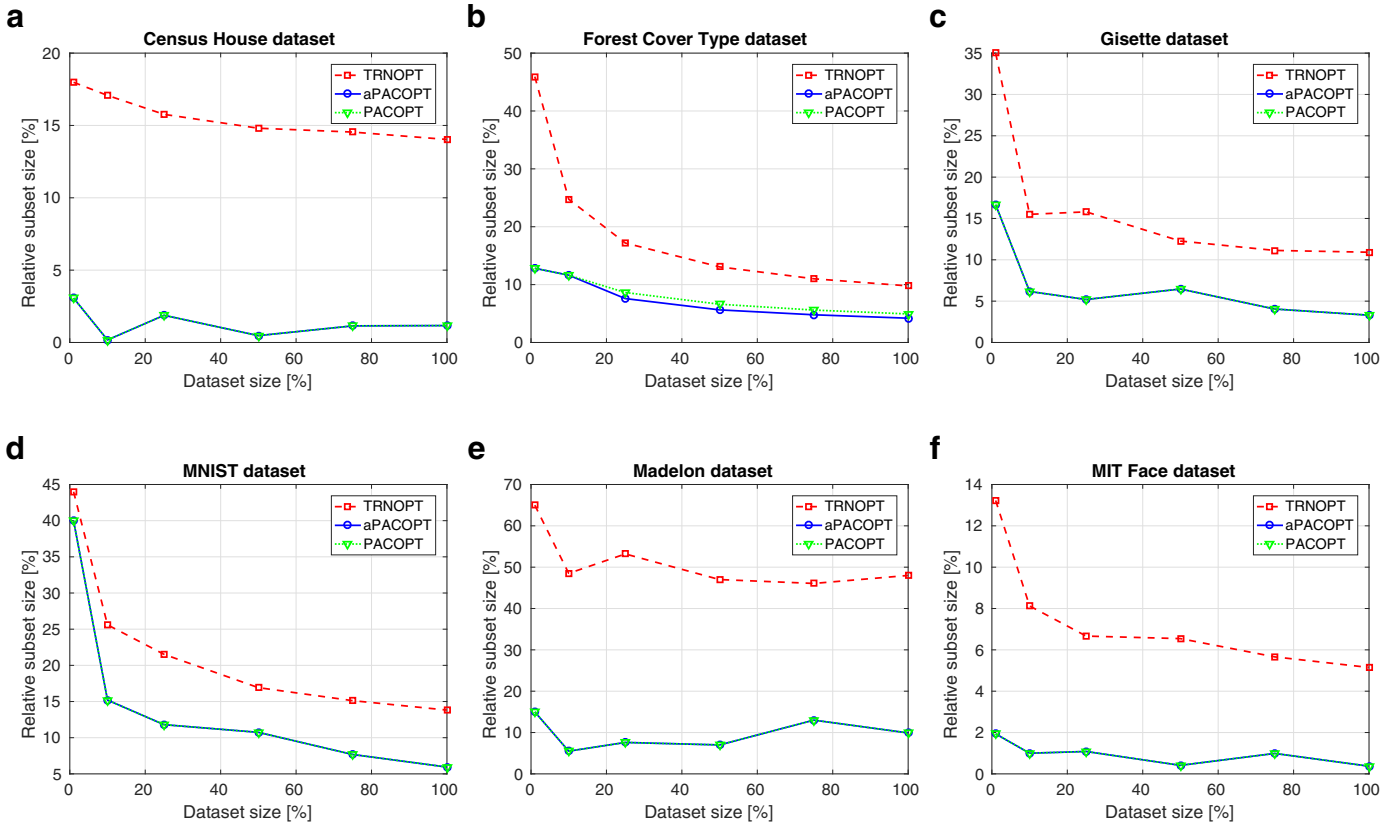


Fig. 4. TRNOPT, PACOPT, and aPACOPT relative model size on large datasets.

As for the other datasets, the size of the subsets is (aPACOPT vs PACOPT): 1.16% vs 14.03% for *Census*; 3.28% vs 10.92% for *Gisette*; 9.85% vs 48% for *Madelon*; and 0.37% vs 5.15% for *MIT Face*.

Fig. 4 accounts for the growth of the model size as a function of the dataset size. Specifically, the plots report the size of the model as a percentage of the dataset size that it is induced on. It can be seen that all the strategies are particularly suitable for large data, since in both cases the percentage of the selected objects is decreasing with the number of objects composing the training set. Moreover, the reduction ratio is also remarkable for high-dimensional datasets.

Finally, Fig. 5 shows the execution time of the strategies versus the size of the dataset. We recall that the execution time of TRNOPT and PACOPT is the same. The plots highlight that, despite the fact that aPACOPT is able to determine a model of quality comparable to that of PACOPT, by using the aPACOPT strategy great time savings are obtained with respect to the non-approximate PACOPT strategy.

5.3. Comparison of aPACOPT with hybrid methods

Garcia et al. [39] propose an extensive taxonomy based on the main characteristics presented by prototype selection methods for the nearest neighbor classification rule. The taxonomy exploits a number of criteria that can be used to evaluate the relative strengths and weaknesses of each algorithm. These include storage reduction, noise tolerance, generalization accuracy, and time requirements.

According to the analysis there accomplished, DROP3, CCIS, SSMA, and RMHC are remarkable methods belonging to the hybrid family. The best methods, considering the tradeoff reduction/accuracy rate, are RMHC, and SSMA, over medium data sets. However, these methods achieve a significant improvement in the accuracy rate at the expense of a high computational cost. The execution time of these techniques could be prohibitive when the data scales up. The methods that harm the accuracy at the expense of a great reduction of time complexity are DROP3 and CCIS.

Thus, we selected these remarkable prototype selection techniques in order to accomplish a scalability comparison taking into account the novel PAC-based FCNN pruning strategy. As for the competitor methods we used the implementation within the KEEL software platform. KEEL[6] (Knowledge Extraction based on Evolutionary Learning) tool is an open source (GPLv3) software based on data flow to design experiments with different datasets and computational intelligence algorithms.

Fig. 6 illustrates the scalability analysis of CCIS, DROP3, RMHC, SSMA, FCNN (TRNOPT strategy) and FCNN-PAC (aPACOPT strategy) on the *MNIST* and *Covtype* datasets.

Datasets up to 10000 objects for *Covtype* and up to 5000 objects for *MNIST* have been considered, due to the heavy time requirements of competitor methods. As for the execution time of FCNN and FCNN-PAC on larger dataset, the reader is referred to Fig. 5.

Consider the model size. Although FCNN-PAC, together with CCIS, presents the smaller reduction ratio, the FCNN-PAC strategy is able to obtain a model of size comparable to that of competitors that are notable for their reduction ratio, as RMHC and DROP3. As for SSMA, it appears to be able to reach the best reduction ratio.

As for as the exection time is concerned, however, SSMA exhibits very large time requirements. On *Covtype* it requires, together with RMHS, more than 5000s on the dataset of size 10000. Compare this time with the 0.55 s required by FCNN-PAC.

[6] http://keel.es/.

Please cite this article as: F. Angiulli, E. Narvaez, Pruning strategies for nearest neighbor competence preservation learners, Neurocomputing (2018), https://doi.org/10.1016/j.neucom.2018.04.017

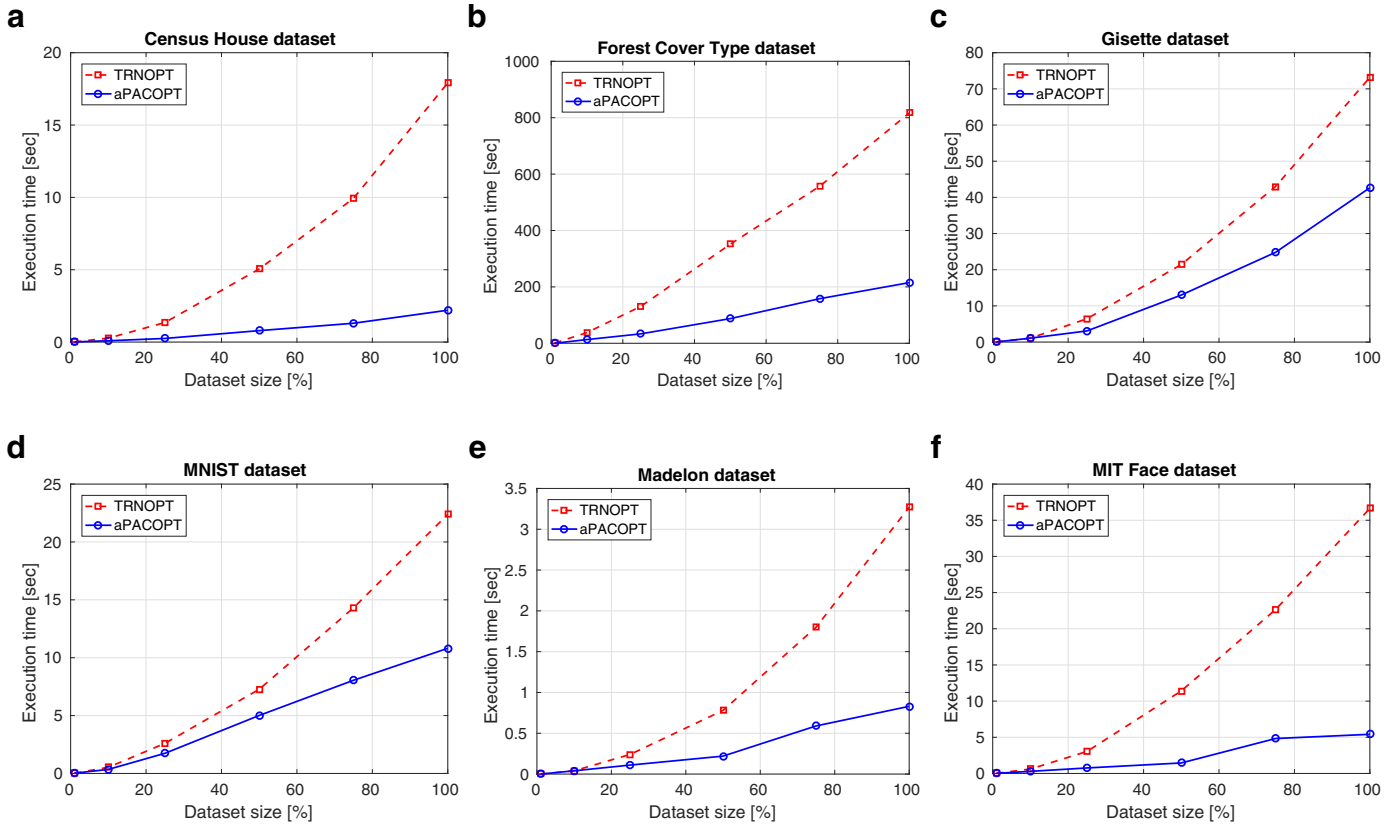


Fig. 5. TRNOPT, PACOPT, and aPACOPT execution time on large datasets.

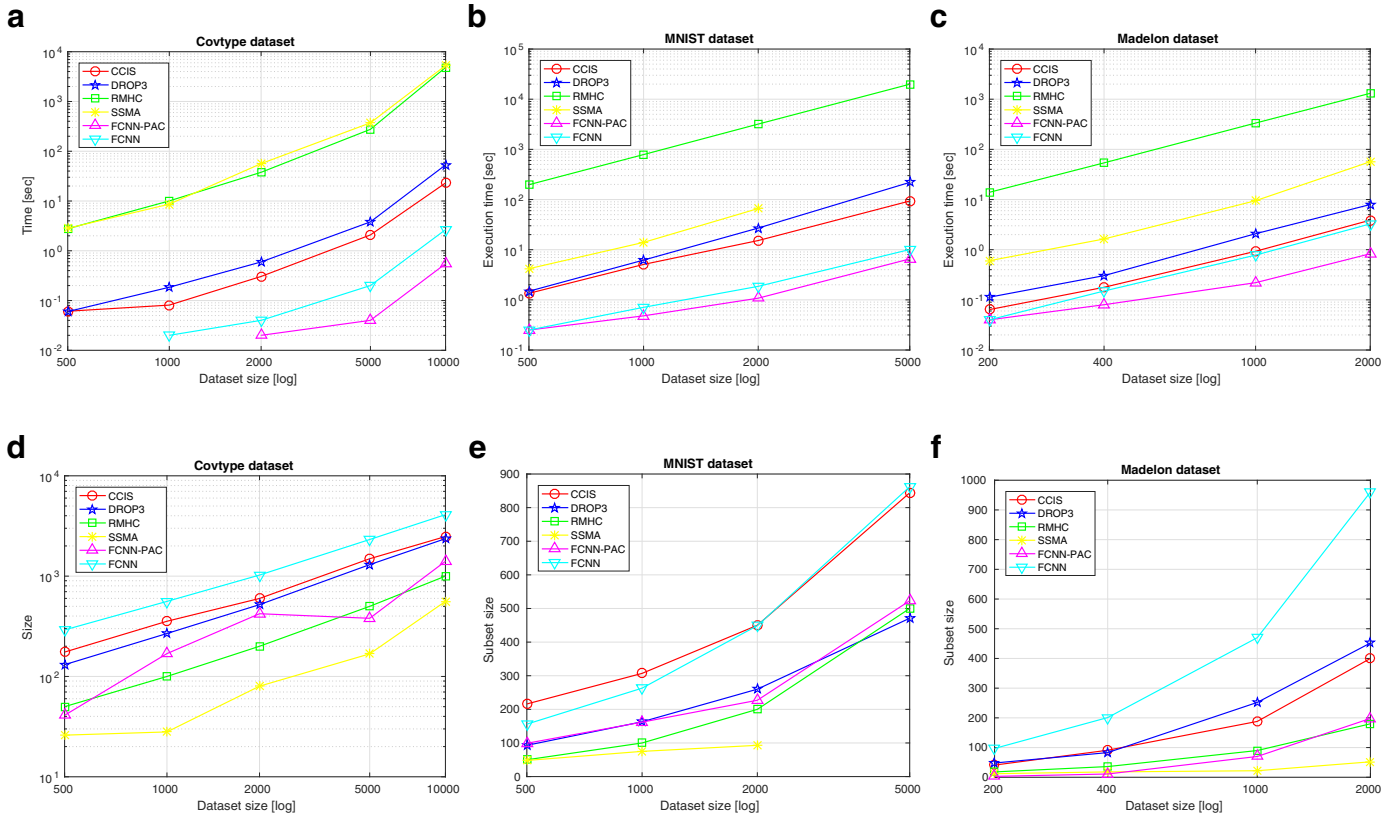


Fig. 6. [Best viewed in color] Comparison with hybrid methods.

Please cite this article as: F. Angiulli, E. Narvaez, Pruning strategies for nearest neighbor competence preservation learners, Neurocomputing (2018), https://doi.org/10.1016/j.neucom.2018.04.017

On *MIST*, SSMA was not able to run on datasets of size larger than 2000 due to memory requirements (the algorithm reported an "out of memory" error of these instances). On the same dataset, FCNN-PAC required 6.55 s on the larger instance, while RMHC employed about 20000 s to process the same input.

6. Conclusion

Motivated by approaches designed to improve generalization and to prevent induction of overly complex models, in this study we investigated the application of the Pessimistic Error Estimate (PEE) principle in the context of nearest neighbor rule competence preservation techniques. As major results, we showed that the here introduced selection strategies guarantee to preserve the accuracy of a nearest neighbor consistent subset with a far larger reduction factor, that sensible generalization improvements can be obtained by using a reduced subset of intermediate size, and that these strategies are able to obtain a model of size comparable to that obtained by the best prototype selection methods, but being different orders of magnitude faster.

References

[1] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Trans. Inf. Theory 13 (1) (1967) 21–27.
[2] C.J. Stone, Consistent nonparametric regression, Annals Stat. 5 (4) (1977) 595–620.
[3] L. Devroye, On the inequality of cover and hart in nearest neighbor discrimination, IEEE Trans. Pattern Anal. Mach. Intell. (1) (1981) 75–78.
[4] P. Hart, The condensed nearest neighbor rule, IEEE Trans. Inf. Theory 14 (3) (1968) 515–516.
[5] D.R. Wilson, T.R. Martinez, Reduction techniques for instance-based learning algorithms, Mach. Learn. 38 (3) (2000) 257–286.
[6] B. Karacali, H. Krim, Fast minimization of structural risk by nearest neighbor rule, IEEE Trans. Neural Netw. 14 (1) (2003) 127–137.
[7] V.S. Devi, M.N. Murty, An incremental prototype set building technique, Pattern Recognit. 35 (2) (2002) 505–513.
[8] F. Angiulli, Fast nearest neighbor condensation for large data sets classification, IEEE Trans. Knowl. Data Eng. 19 (11) (2007) 1450–1464.
[9] P.-N. Tan, M. Steinbach, V. Kumar, Classification: basic concepts, decision trees, and model evaluation, Introd. Data Mining 1 (2006) 145–205.
[10] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann, 2005.
[11] C. Schaffer, Overfitting avoidance as bias, Mach. Learn. 10 (2) (1993) 153–178.
[12] T. Dietterich, Overfitting and undercomputing in machine learning, ACM Comput. Surv. (CSUR) 27 (3) (1995) 326–327.
[13] D.D. Jensen, P.R. Cohen, Multiple comparisons in induction algorithms, Mach. Learn. 38 (3) (2000) 309–338.
[14] D.M. Hawkins, The problem of overfitting, J. Chem. Inf. Comput. Sci. 44 (1) (2004) 1–12.
[15] P. Grunwald, A tutorial introduction to the minimum description length principle, arXiv:0406077 (2004).
[16] J. Rissanen, Minimum Description Length Principle, Wiley Online Library, 1985.
[17] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.
[18] J. Han, M. Kamber, J. Pei, Data Mining: Concepts and Techniques, Elsevier, 2011.
[19] J.R. Quinlan, R.L. Rivest, Inferring decision trees using the minimum description length principle, Inf. Comput. 80 (3) (1989) 227–248.
[20] J.R. Quinlan, Simplifying decision trees, Int. J. Human-Comput. Stud. 51 (2) (1999) 497–510.
[21] T. Niblett, I. Bratko, Learning decision rules in noisy domains, in: Proceedings of the Expert Systems' 86, The 6Th Annual Technical Conference on Research and Development in Expert Systems III, Cambridge University Press, 1987, pp. 25–34.
[22] L. Breiman, J. Friedman, C.J. Stone, R.A. Olshen, Classification and Regression Trees, CRC press, 1984.
[23] Y. Mansour, Pessimistic decision tree pruning based on tree size, in: Proceedings of the Machine Learning-International Workshop then Conference, Citeseer, 1997, pp. 195–201.
[24] J.R. Quinlan, C4. 5: Programs for Machine Learning, Elsevier, 2014.
[25] F. Esposito, D. Malerba, G. Semeraro, J. Kay, A comparative analysis of methods for pruning decision trees, IEEE Trans. Pattern Anal. Mach. Intell. 19 (5) (1997) 476–491.
[26] Q. Dai, A novel ensemble pruning algorithm based on randomized greedy selective strategy and ballot, Neurocomputing 122 (Supplement C) (2013) 258–265.
[27] M.B. Stojanovi, M.M. Boi, M.M. Stankovi, Z.P. Staji, A methodology for training set instance selection using mutual information in time series prediction, Neurocomputing 141 (Supplement C) (2014) 236–245.
[28] O. Loyola-Gonzlez, J.F. Martnez-Trinidad, J.A. Carrasco-Ochoa, M. Garca-Borroto, Study of the impact of resampling methods for contrast pattern based classifiers in imbalanced databases, Neurocomputing 175 (Part B) (2016) 935–947.
[29] R. Hu, X. Zhu, D. Cheng, W. He, Y. Yan, J. Song, S. Zhang, Graph self-representation method for unsupervised feature selection, Neurocomputing 220 (Supplement C) (2017) 130–137.
[30] H. Brighton, C. Mellish, Advances in instance selection for instance-based learning algorithms, Data Min. Knowl. Discov. 6 (2) (2002) 153–172.
[31] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Trans. Syst. Man Cybern. (3) (1972) 408–421.
[32] I. Tomek, An experiment with the edited nearest-neighbor rule, IEEE Trans. Syst. Man Cybern. (6) (1976) 448–452.
[33] W. Gates, The reduced nearest neighbor rule(1972).
[34] G. Ritter, H. Woodruff, S. Lowry, T. Isenhour, An algorithm for a selective nearest neighbor decision rule, IEEE Trans. Inf. Theory 21 (6) (1975) 665–669.
[35] D.W. Aha, D. Kibler, M.K. Albert, Instance-based learning algorithms, Mach. Learn. 6 (1) (1991) 37–66.
[36] J. Zhang, Selecting typical instances in instance-based learning, in: Proceedings of the Ninth International Conference on Machine Learning, 1992, pp. 470–479.
[37] C.E. Brodley, Addressing the selective superiority problem: automatic algorithm/model class selection, in: Proceedings of the Tenth International Conference on Machine Learning, 1993, pp. 17–24.
[38] D.R. Wilson, T.R. Martinez, Instance pruning techniques, in: Proceedings of the ICML, vol. 97, 1997, pp. 403–411.
[39] S. Garcia, J. Derrac, J. Cano, F. Herrera, Prototype selection for nearest neighbor classification: taxonomy and empirical study, IEEE Trans. Pattern Anal. Mach. Intell. 34 (3) (2012) 417–435.
[40] E. Marchiori, Class conditional nearest neighbor for large margin instance selection, IEEE Trans. Pattern Anal. Mach. Intell. 32 (2) (2010) 364–370.
[41] S. García, J.R. Cano, F. Herrera, A memetic algorithm for evolutionary prototype selection: a scaling up approach, Pattern Recognit. 41 (8) (2008) 2693–2709.
[42] D.B. Skalak, Prototype and feature selection by sampling and random mutation hill climbing algorithms, in: Proceedings of the Eleventh International Conference on Machine Learning, 1994, pp. 293–301.
[43] H.A. Fayed, A.F. Atiya, A novel template reduction approach for the K -nearest neighbor method, IEEE Trans. Neural Netw. 20 (5) (2009) 890–896.
[44] J.A. Olvera-López, J.A. Carrasco-Ochoa, J.F. Martínez-Trinidad, A new fast prototype selection method based on clustering, Pattern Anal. Appl. 13 (2) (2010) 131–141.
[45] J. Li, Y. Wang, A new fast reduction technique based on binary nearest neighbor tree, Neurocomputing 149 (Part C) (2015) 1647–1657.
[46] F. Angiulli, Fast condensed nearest neighbor rule, in: Proceedings of the 22nd International Conference on Machine Learning, ACM, 2005, pp. 25–32.

Fabrizio Angiulli is an associate professor of computer engineering at DIMES, University of Calabria, Italy. Previously, he held a research and development position at ICAR of the National Research Council of Italy and, after that, a tenured assistant professor position at DEIS, University of Calabria. His main research interests are in the area of data mining, notably outlier detection and classification techniques, knowledge representation and reasoning and database management and theory. He has authored more than one hundred papers appearing in premier journals and conference proceedings. He regularly serves on the program committee of several conferences and, as an associate editor, on the editorial board of AI Communications. He is a senior member of the IEEE and a member of ACM and CVPL.

Estela Narvaez was born in Guano, Ecuador, in 1982. Her undergraduate studies took place at National University of Chimborazo (UNACH), Ecuador, from which she graduated with a Master Degree in Computing Engineering in 2012. Since then, she is a member of the Systems Engineering college in Riobamba-Ecuador. She received her Ph.D. in Information and Communication Engineering for Pervasive Intelligent Environments from the DIMES Department of University of Calabria, Italy, in November 2017. Currently, she is a Professor at the Faculty of Engineering, UNACH, Ecuador. Her research interests include data mining and knowledge discovery, with a focus on nearest neighbors methods and network structured data.

Please cite this article as: F. Angiulli, E. Narvaez, Pruning strategies for nearest neighbor competence preservation learners, Neurocomputing (2018), https://doi.org/10.1016/j.neucom.2018.04.017