Short Papers

Class Conditional Nearest Neighbor for Large Margin Instance Selection

Elena Marchiori

Abstract—This paper presents a relational framework for studying properties of labeled data points related to proximity and labeling information in order to improve the performance of the 1NN rule. Specifically, the class conditional nearest neighbor (*ccnn*) relation over pairs of points in a labeled training set is introduced. For a given class label c, this relation associates to each point a its nearest neighbor computed among only those points with class label c (excluded a). A characterization of *ccnn* in terms of two graphs is given. These graphs are used for defining a novel scoring function over instances by means of an information-theoretic divergence measure applied to the degree distributions of these graphs. The scoring function is employed to develop an effective large margin instance selection method, which is empirically demonstrated to improve storage and accuracy performance of the 1NN rule on artificial and real-life data sets.

Index Terms—Computing methodologies, artificial intelligence, learning, heuristics design, machine learning.

1 Introduction

In a typical classification problem, we are given a training set consisting of sample points and their class labels. The training set is used for predicting the class of new sample points. In particular, the one nearest neighbor (1NN) rule classifies an unknown point into the class of the nearest of the training set points. 1NN is used in many applications because of its intuitive interpretation, flexibility, and simple implementation. Moreover, for all distributions, the 1NN rule's probability of error is bounded above by twice the Bayes' probability of error [12]. However, 1NN requires to memorize the entire training set (that is, it is a memory-based classifier), and its performance can be negatively affected by the presence of many input variables (see, for instance, [18], [21], [32]) or noisy instances (see, for instance, [8], [40]). In order to tackle these problems, various algorithms have been developed, such as those for instance/prototype selection [1], [2], [3], [8], [20], [24], [33], for feature selection [21], [27], [38], and for distance learning [31], [42], [43].

The 1NN rule does not rely on knowledge of the underlying data distribution (nonparametric classification), but uses directly proximity followed by class labeling information for classifying new points. Therefore, in this paper, we analyze proximity *conditioned* to class labeling information in order to improve accuracy and storage performance of the 1NN rule. We introduce a relation called class conditional nearest neighbor (*ccnn*), defined on pairs of points from a labeled training set as follows: For a given class c, *ccnn* associates to instance a its *nearest neighbor* computed among only those instances (excluded a) *in the class* c. Thus, this relation describes proximity information *conditioned* to a class label, for each class of the training set.

• *The author is with the Institute for Computing and Information Sciences (ICIS), Faculty of Science, Radboud University, Toernooiveld 1, NL 6525 ED Nijmegen, The Netherlands. E-mail: elenam@cs.ru.nl.*

Manuscript received 24 Sept. 2008; revised 19 June 2009; accepted 14 Aug. 2009; published online 19 Aug. 2009.
Recommended for acceptance by B. Scholkopf.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2008-09-0642.
Digital Object Identifier no. 10.1109/TPAMI.2009.56.

The *ccnn* relation is characterized by means of two graphs: the between-class and within-class nearest neighbor graphs. These graphs are used to define a new instance scoring function by means of a directed information-theoretic measure (the K-divergence) applied to the in-degree distributions of these graphs. The scoring function is used to develop an effective large margin instance selection method, called *Class Conditional* selection (CC). Points of the training set with negative or zero score are discarded, since it is shown that their removal increases the hypothesis margin of the resulting 1NN rule. The instance selection method selects iteratively instances, where instances with higher score are selected first. The process terminates when the empirical error of the resulting 1NN rule increases. Results of extensive experiments with artificial and real-life data sets show that the method improves significantly the 1NN rule's storage and test accuracy performance. Moreover, the test accuracy results of CC are similar to those of the KNN classifier that uses the entire training set and selects the number K of neighbors by leave-one-out cross validation.

In order to further improve the storage performance of the method, we develop a postprocessing algorithm, called *Thin-out* selection (THIN) that selects points close to the decision boundary of the 1NN rule. This is achieved by selecting instances having positive in-degree in the between-class graph of the actual training set. The process is repeated on the remaining instances until the empirical error increases. Application of CC followed by THIN is called *Class Conditional Instance Selection* (CCIS). Experimental comparison with two state-of-the-art instance selection algorithms, ICF and DROP3, described in Section 4.1, indicate similar storage reduction of the methods. Test accuracy results of CCIS are significantly better than those of ICF and similar to those of DROP3. Finally, results show that CC has test accuracy significantly better than that of these instance selection algorithms but worse storage reduction.

These results show the usefulness of *ccnn* for defining properties of training set instances to be used for improving the performance of the 1NN rule.

1.1 Related Work

The *ccnn* relation is related to Hit Miss networks (HMNs) introduced in [29]. In that paper, it was shown that structural properties of HMNs correspond to properties of training points related to the decision boundary of the 1NN rule, such as being border or central point. This observation was used to introduce an instance selection heuristic algorithm for the 1NN rule based on HMNs. Here, we use two components of HMNs for defining a new information-theoretic instance scoring function used to perform large margin instance selection.

Graph-based representations of training sets in the context of 1NN-based classification mainly use proximity graphs. Proximity graphs are defined as graphs in which points close to each other by some definition of closeness are connected [4]. The nearest neighbor graph (NNG) is a typical example of proximity graph, where each vertex is a data point that is joined by an edge to its nearest neighbor. Representations of a data set based on proximity graphs have been used with success to define algorithms for improving storage and accuracy of the nearest neighbor rule. For a thorough survey of graph-based methods for nearest neighbor classification, the reader is referred to [40].

A popular relation involving both proximity and class labeling information is the nearest unlike neighbor (NUN) [14], which links one point with its nearest neighbor among those points with different class label. NUN has been used in [16] to provide a measure of confidence in the decisions made by the 1NN-based

decision systems, and employed in [15] for defining hybrid condensing algorithms. We show that *ccnn* incorporates both types of label-independent (nearest neighbor) and label-dependent (nearest unlike neighbor) information.

The heuristic algorithms CC and THIN are motivated by works on large margin analysis of prototype selection [6], [13] and feature selection [21]. In [21], hypothesis margin is used to define a loss function for performing feature weighting, and introduce a large margin bound of the generalization error for the 1NN rule, which uses a set of selected features. In [13], the notion of hypothesis margin is introduced and used to provide a large margin bound for the generalization error of a family of prototype selection algorithms. In particular, they show that 1NN generalizes well if a prototype selection algorithm selects a small number of prototypes with large hypothesis margin and small training error. These three objectives are directly used in the heuristic algorithms for instance selection that we propose.

The main differences between CCIS and prototype selection algorithms, such as those analyzed in [13], are that, in CCIS, prototypes are members of the training set and are automatically computed. Indeed, CCIS belongs to the family of instance selection algorithms. Instance selection algorithms, and in particular CCIS, can be interpreted as procedures for training Voronoi networks (Vnets) [26]. Voronoi networks discretize the feature space into Voronoi regions and assign the samples in each region to a class. In [26], it is shown that Vnets asymptotically converge to the Bayes classifier with arbitrary high probability provided the number of representative samples grows slower than the square root of the number of training samples.

The heuristic for instance selection here proposed differs from previous instance selection algorithms, such as those mentioned in Section 1, mainly because it employs a new instance scoring function that is used to directly enlarge the hypothesis margin while selecting instances.

The rest of the paper is organized as follows: After presenting the notation used throughout the paper, Section 2 introduces the *ccnn* relation, its graph-based representation, and comparison with NNG and NUN. In Section 3, we develop a large margin instance selection algorithm based on *ccnn*, whose performance is comparatively analyzed experimentally in Section 4 on a large collection of data sets. Finally, we conclude the paper in Section 5 with a summary of the contributions and point to future work.

1.2 Background

In this paper, we use A to denote a data set of n instances $A = \{a_1, \ldots, a_n\}$, where a_i is a real-valued vector of dimension m. Let C denote the set of class labels of A and let $l : A \to C$ the function mapping each instance a_i to its class label $l(a_i)$.

A graph $G = (V, E)$ consists of a finite set V and a subset $E \subset V \times V$. The elements of V are the *vertices* of the graph and those of E are the edges of the graph. In this work, we consider directed graphs, that is, such that each edge $(u, v) \in E$ is oriented (from u to v). We say that u and v are *adjacent* vertices, denoted by $u \sim v$, if $(u, v) \in E$. The degree function deg is defined by $deg(u) = |\{v \mid u \sim v \text{ or } v \sim u\}|$. The *in-degree* function in_deg is defined by $in\_deg(u) = |\{v \mid v \sim u\}|$.

We denote by ϵ^S the *leave-one-out* error of A calculated using the 1NN rule with S as training set; that is, if $a \in A$ is also in S, then it is classified by the 1NN rule using as training set S without a [21]. We denote by ϵ^S the training or empirical error of S.

2 Class Conditional Nearest Neighbor

With the aim to analyze class and proximity information contained in a training set in an integrated fashion, the following relation is introduced:

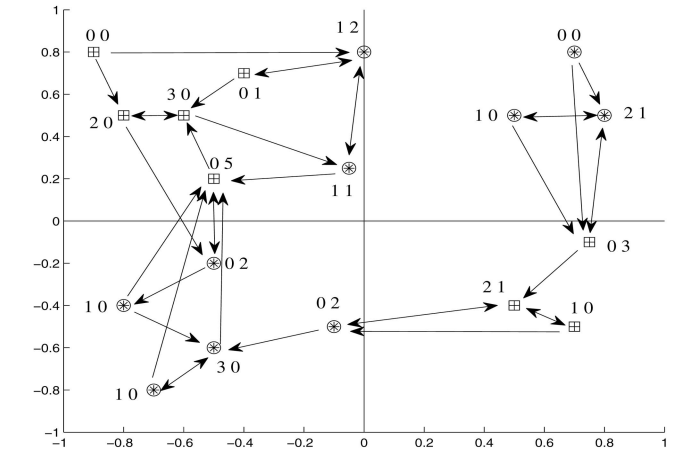

Fig. 1. Graph representation of *ccnn* for a toy data set with two classes, with number of incoming within and between-class edges reported near each node.

Definition 2.1 (ccnn). *Given a class c, the nearest neighbor of a conditioned to c, denoted by $\mathtt{1NN}(a, c)$, is the nearest neighbor of a computed among those points in A, excluding a, having class label c. We call ccnn the set of pairs (a, b) such that $b = \mathtt{1NN}(a, c)$ for some class c of C.*

Let $\mathtt{1NN}(a)$ denote the nearest neighbor of a in A. In the following, we assume for simplicity that $\mathtt{1NN}(a)$ and $\mathtt{1NN}(a, c)$ are unique.

We use a graph-based representation of the class conditional nearest neighbor, where nodes are instances and there is an edge (a, b) iff (a, b) is in *ccnn*.

Such a graph-based representation of the training set is shown in Fig. 1 for a toy binary classification problem. The in-degree of each point is also plotted. Observe that the two points with zero in-degree are relatively isolated from other points. Moreover, points with high number of incoming edges from a different class are closer to the 1NN decision boundary.

Constructing such a graph representation requires quadratic time complexity in the number of points. Nevertheless, by using metric trees or other spatial data structures, this bound can be reduced [22], [25]. For instance, for low input dimension, using *kd* trees, whose construction takes time proportional to $nlog(n)$, nearest neighbor search exhibits approximately $O(n^{1/2})$ behavior [22].

It is easy to check that *ccnn* is characterized in graph terms by the union of the two orthogonal graphs: the within and between-class directed nearest neighbor graphs, defined as follows:

Definition 2.2. *The* within-class *1NN* graph, *denoted by $G_{wc} = (V, E_{wc})$, is such that $V = A$ and*

$$E_{wc} = \{(a_i, a_j) \mid a_j = \mathtt{1NN}(a_i, l(a_i))\}.$$

The between-class *1NN* graph, *denoted by $G_{bc} = (V, E_{bc})$, is such that $V = A$ and*

$$E_{bc} = \{(a_i, a_j) \mid a_j = \mathtt{1NN}(a_i, c), c \in C \text{ and } c \neq l(a_i)\}.$$

G_{wc} represents the directed 1NN relation *between points of the same class* in the training set. G_{bc} represents the directed 1NN relation *between points of each pair of different classes* in the training set.

The *ccnn* relation contains and integrates two popular relations introduced in past work on training set analysis and instance selection for the 1NN rule: the (directed) nearest neighbor (NNG) and the NUN.

The (directed) nearest neighbor relation has been applied with success in studies and applications of the nearest neighbor rule (see, e.g., [40]). This relation is strictly contained in *ccnn*. Moreover,

one can easily show that NNG and G_{wc} coincide for those training sets A such that $\epsilon^A = 0$.

The NUN is a useful concept used for decades in diverse application domains, for instance, geology [30] for detecting border points. Here, we consider the NUN concept as defined in [14] in the machine learning community: The nearest unlike neighbor of a is its nearest neighbor among those points in A having different class label. This concept was applied in [16] to provide a measure of confidence in the decisions made by the 1NN-based decision systems, and employed in [15] for defining hybrid condensing algorithms. Clearly, in general, the NUN relation is strictly included in the between-class nearest neighbor (G_{bc}), and it coincides with it only for binary classification problems.

In the sequel, we use the in-degree of points in the G_{wc} and G_{bc} graphs for developing a large margin instance selection heuristic. For simplicity, we will refer to the in-degree of a in G_{wc} and G_{bc} as the *within* and *between* in-degree of a, respectively.

3 Class Conditional Instance Selection

Margins play an important role in machine learning research, as a tool for performing theoretic analysis [5], [37], for developing new machine learning algorithms [11], and for improving the accuracy performance of nearest neighbor-based classifiers [6], [13], [21]. In particular, the *hypothesis margin* is defined as the distance between the hypothesis and the closest hypothesis that assigns alternative label to the given instance. For the 1NN rule, the hypothesis margin of an instance a with respect to a training set A can be easily computed as follows:

$$\theta_A(a) = \|a - nearestmiss(a)\| - \|x - nearesthit(a)\|,$$

where $nearesthit(a)$ and $nearestmiss(a)$ are the nearest neighbors of a with equal and different class labels, and $\|\cdot\|$ denotes the euclidean norm [13].

A *Learning Vector Quantization* (LVQ) algorithm seeks a set of prototypes of a given size in the input space, typically by minimizing a suitable loss function using gradient search. In [13], it is shown that this popular prototype selection algorithm belongs to a family of maximal margin algorithms [19], [41]. Furthermore, it is theoretically shown that if a prototype selection algorithm selects a small set of prototypes with large margin θ and small θ-error, then the resulting 1NN rule will generalize well.

Here, we consider instance selection, that is, prototypes are constrained to be members of the training set. Instance selection can be interpreted as training process for a family of learning machines, also known in the literature as Voronoi networks [26]. Since LVQ and Voronoi networks are different and we are not aware of large margin bounds results for Voronoi networks, we use the above theoretical result given in [13] as a guideline for developing a large margin instance selection method that consists of the following two phases:

- Class Conditional selection phase (CC). It removes from the training set outliers, isolated points, and points close to the 1NN decision boundary. This phase aims at *enlarging the hypothesis margin* and *reducing the empirical error*.
- Thin-out selection phase (THIN). It thins out points that are not important to the decision boundary of the resulting 1NN rule. This phase aims at selecting a *small number of instances* without negatively affecting the 1NN empirical error.

The two phases are described in detail below.

3.1 Class Conditional Selection: CC

A subset of the original training set is constructed from an initial small core S by adding incrementally points to S. The crucial step of CC is the criterion used for selecting one instance at each iteration. We propose a static instance selection criterion based on a directed information-theoretic divergence measure known as K-divergence (see, for instance, [28]).

1: $(a_1, \ldots, a_n) = A$ sorted in decreasing order of Score;
2: $S = (a_1, \ldots, a_{k_0})$;
3: $i = k_0 + 1$;
4: $go\_on = 1$;
5: $ub = n - |\{a \text{ s.t. } \texttt{Score}(a) \leq 0\}|$;
6: **while** $i < ub$ and go_on **do**
7: $\quad Temp = S \cup \{a_i\}$;
8: $\quad$ **if** $\epsilon^S \leq \epsilon^A$ **then**
9: $\quad\quad go\_on = 0$;
10: $\quad$ **end if**
11: $\quad$ **if** $\epsilon^{Temp} < \epsilon^S$ and go_on **then**
12: $\quad\quad$ S = Temp;
13: $\quad\quad i = i + 1$;
14: $\quad$ **else**
15: $\quad\quad go\_on = 0$;
16: $\quad$ **end if**
17: **end while**

Fig. 2. Pseudocode of CC. Input: training set A. Output: subset S of A.

Let p_1 and p_2 be two discrete probability distributions over X. The K-divergence between p_1 and p_2 is

$$K(p_1, p_2) = \sum_{x \in X} p_1(x) \log \frac{p_1(x)}{\frac{1}{2}p_1(x) + \frac{1}{2}p_2(x)}.$$

$K(p_1, p_2)$ can also be defined as the Kullback-Leibler (KL) divergence of p_1 and $\frac{1}{2}p_1 + \frac{1}{2}p_2$, $KL(p_1 \| \frac{1}{2}p_1 + \frac{1}{2}p_2)$. The K-divergence is a nonsymmetric, bounded measure of divergence [28].

Here, we choose as p_1 and p_2 the normalized within and between-in-degree distributions, denoted by p_w and p_b, respectively; that is, $p_w(a)$ and $p_b(a)$ are the within and between-in-degree of point a divided by the total in-degree of G_{wc} and G_{bc}, respectively. Note that p_w and p_b are, in general, different distributions and $p_w(a) \neq 1 - p_b(a)$.

Denote by

$$K(p_1, p_2)(a) = p_1(a) \log \frac{p_1(a)}{\frac{1}{2}p_1(a) + \frac{1}{2}p_2(a)},$$

the contribution of instance a to the K-divergence. We consider the difference of the contributions of a to $K(p_w, p_b)$ and $K(p_b, p_w)$ as a measure for scoring points.

Definition 3.1. *The* class conditional score *of an instance a is* $\texttt{Score}(a) = K(p_w, p_b)(a) - K(p_b, p_w)(a)$.

Points that contribute more to $K(p_b, p_w)$ than to $K(p_w, p_b)$ have negative score. In particular, a point a with higher between-class than within-class in-degree has negative score. Removing a from the training set will increase the hypothesis margin of a number of points equal to the difference of its between and within-class in-degrees. Therefore, we discard points with negative score (see line 5 of Fig. 2).

Fig. 2 shows the algorithm in pseudocode. The initial core S consists of the k_0 instances having highest score, with $k_0 = max(c, \lceil \frac{\epsilon^A}{2} \rceil)$, that is, we choose an initial core of instances of size proportional to the difficulty of the task, as measured by ϵ^A. This choice is motivated by the following reasoning. We want to have at least one point for each class. The number of misclassified instances of the training set is equal to ϵ^A. Without any prior

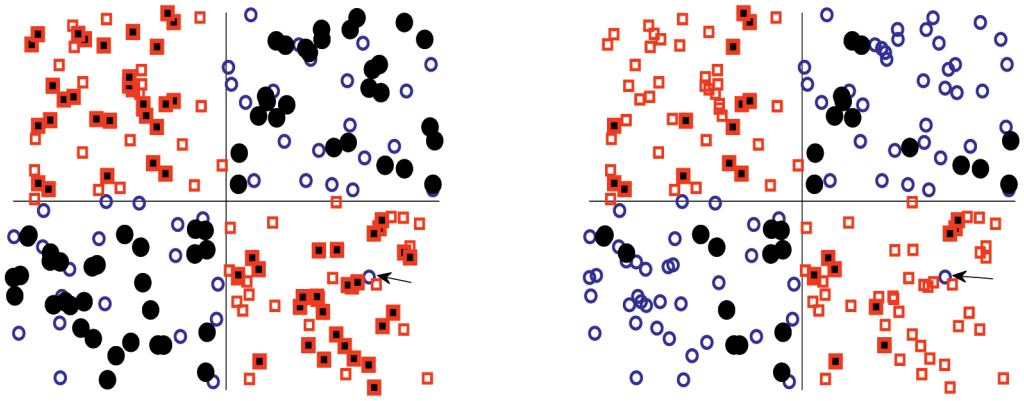

Fig. 3. Application of (a) CC selection and (b) THIN. The points selected have filled markings. The arrow points to one outlier.

knowledge, we assume with equal probability that each of these instances is either an outlier or a regular instance. In order to be correctly classified, these regular instances have to be included in the training set. Therefore, S will contain at least $\lceil \frac{\epsilon^A}{2} \rceil$ points. Application of automatic parameter tuning procedures to choose k_0, for instance, internal cross validation, may possibly yield improved performance.

At each iteration, CC selects the instance with the highest score that best contributes to achieve a large hypothesis margin, that is, being far from points of other classes and close to points of the same class. The iterative process continues while the empirical error ϵ^S decreases, until ϵ^S becomes smaller than ϵ^A.

The algorithm is applied independently to c pairs of classes. Such pairs are selected as follows: For each class c, the class most similar to c is selected. Here, similarity between classes c and c' is defined by considering the subgraph induced by nodes of c and c', and by computing the correlation between the within and between-class in-degrees of the points in class c in such a graph. The union of the selected instances obtained from the c independent applications of the algorithm is given as final result of CC.

Fig. 3a illustrates the effect of the algorithm on a training set of the XOR classification problem. In particular, it shows that CC selection discards the outlier point (indicated by an arrow), isolated instances (with negative or zero score, see line 5), as well as instances close to the 1NN decision boundary, thus, enlarging the 1NN hypothesis margin.

We turn now to the description of the second phase of the method.

3.2 Thin-Out Instance Selection: THIN

The reduced training set S given as output of CC is further processed by selecting only points considered important to the decision boundary. Pseudocode of THIN is given in Fig. 4, where G_{bc}^S denotes the between-class graph constructed using only points in S.

The algorithm takes as input the subset S of the training set produced by CC and outputs the subset S_f of S constructed as follows: S_f is initialized to the set of those points close to the 1NN decision boundary of S, that is, having positive in-degree in the between-class graph G_{bc}^S. Points of S_f are removed from S. The resulting set is denoted by S_1 (initialized in line 3 and updated in line 11). The process is iterated as follows (lines 5-13): Points having positive in-degree in the $G_{bc}^{S_1}$ are added to S_f if they were not "iolated" in the previous iteration, that is, if their in-degree was not zero (see line 6). This latter condition is justified by the fact that removing points with zero in-degree from a training set does not change the original 1NN decision boundary (see [29]). The iterative process terminates when the empirical error increases (when go_on becomes false, see line 7).

The effect of the THIN on the XOR example is illustrated in Fig. 3b. Indeed, points close to the 1NN decision boundary are selected.

3.3 Computational Complexity

Constructing G_{wc} and G_{bc} and sorting the CC of the training set take time proportional to $n \log(n)$. The iterative process, in the worst-case, amounts to calculate at each iteration ϵ^S, where at each iteration, one point is added to S.

The computation of ϵ^S requires n tests. Indeed, suppose for each $b \in A$, we memorize its nearest neighbor in $S \setminus \{b\}$, say b_S. Let a be the new point added to S and let $S' = S \cup \{a\}$. Then, for each $b \in A \setminus \{a\}$, if $\|b_S - b\| > \|a - b\|$, then set $b_{S'} = a$; otherwise, $b_{S'} = b_S$.

Thus, CC performs at most $n - k_0 + 1$ iterations. Then the worst-case runtime complexity of CC selection is $O(max(n(n - k_0 + 1), n \log(n))$. THIN does not increase the computational complexity of the algorithm. Experimental evidence on the real-life data sets here considered shows that a small number of iterations are performed in practice. Indeed, on the considered data sets, CC reduces the training set to about 40 percent of its initial size and THIN performs, on average, 3.5 iterations on the reduced training set.

4 Experiments

In order to assess the performance of the proposed instance selection algorithm, experiments on artificial and real-life data sets are conducted. These data sets are publicly available at Raetsch's benchmark repository,[1] Chapelle's repository,[2] and UCI Machine Learning repository.[3]

4.1 Methods

We perform experiments with five algorithms as follows:

- 1NN that uses the entire training set.
- KNN that uses the entire training set and with number K of neighbor selected by leave-one-out cross validation.
- CCIS, consisting of CC selection followed by THIN.
- *Iterative Case Filtering* (ICF) introduced in [7].
- The best performing of the *Decremental Reduction Optimization* algorithms, DROP3 [45], [46].

The *Iterative Case Filtering* method first applies a noise reduction algorithm iteratively until it cannot remove any point, and then, it

1. http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm.
2. http://www.kyb.tuebingen.mpg.de/bs/people/chapelle/lds/.
3. http://mlearn.ics.uci.edu/MLRepository.html.

1: $S_f = \{a \in S$ with in-degree in $G_{bc}^S > 0\}$;
2: $S_{prev} = S$;
3: $S_1 = S \setminus S_f$;
4: go_on =1;
5: **while** go_on **do**
6: $S_t = \{a \in S_1$ with in-degree in $G_{bc}^{S_1} > 0$ and with in-degree in $G_{bc}^{S_{prev}}$ or in $G_{wc}^{S_{prev}} > 0\}$
7: go_on = $\epsilon^{S_f \cup S_t} < \epsilon^{S_f}$;
8: **if** go_on **then**
9: $S_f = S_f \cup S_t$;
10: $S_{prev} = S_1$;
11: $S_1 = S \setminus S_f$;
12: **end if**
13: **end while**

Fig. 4. Pseudocode of THIN. Input: the output S of CC. Output: subset S_f of S.

iteratively removes points as follows: At each iteration, all points for which the so-called *reachability* set is smaller than the *coverage* one are deleted. The reachability of a point a consists of the points inside the largest hypersphere containing only points of the same class as a. The *coverage* of a is defined as the set of points that contains a in their reachability set.

DROP3 belongs to the family of Decremental Reduction Optimization (DROP) algorithms. First, DROP3 applies a preprocessing step that discards points of A misclassified by their K nearest neighbors. Next, it removes a point a from A if the accuracy of the KNN rule on the set of its associates does not decrease. Each point has a list of K nearest neighbors and a list of associates, which are updated each time a point is removed from A. A point a_1 is an *associate* of a if a belongs to the set of K nearest neighbors of a_1. If a is removed, then the list of K nearest neighbors of each of its associates a_1 is updated by adding a new neighbor point a_2, and a_1 is added to the list of associates of a_2. The removal rule is applied to the points sorted in decreasing order of distance from their nearest neighbor of the other classes (nearest enemy). In this way, points farthest from their nearest enemy are selected first.

DROP3 achieves the best mix of storage reduction and generalization accuracy of the DROP methods [46]. Moreover, results of experiments conducted in [45], [46] show that DROP3 achieves higher accuracy and smaller storage requirements than several other methods, such as CNN [23], SNN [35], E-NN [44], the All KNN method [39], IB2, IB3 [1], and the Explore method [9]. Therefore, we use DROP3 and ICF as representatives of the state of the art.

We consider the artificial and real-life data sets with different characteristics reported in Table 1.

4.2 Artificial Data Sets

The Banana data set from Raetsch's benchmark repository consists of 100 partitions of the data set into training and test sets. The two other data sets are from Chapelle's benchmark data [10]: g50c and g10n, generated from two standard normal multivariate Gaussians. In g50c, the labels correspond to the Gaussians, and the means are located in 50-dimensional space such that the Bayes' error is 5 percent. In contrast, g10n is a deterministic problem in 10 dimensions, where the decision function traverses the centers of the Gaussians and depends on only two of the input dimensions. The original 10 partitions of each data set into training and test sets from Chapelle's repository are used.

4.3 Real-Life Data Sets

The following 19 publicly available real-life data sets are used:

TABLE 1
Characteristics of Data Sets

Data	CL	VA	TR	Cl.Inst.	TE	Cl.Inst.
Banana	2	2	400	212-188	4900	2712-2188
g50	2	50	550	252-248	50	23-27
g10n	2	10	550	245-255	50	29-21
B.Cancer	2	9	200	140-60	77	56-21
Diabetes	2	8	468	300-168	300	200-100
German	2	20	700	478-222	300	222-78
Heart	2	13	170	93-77	100	57-43
Image	2	18	1300	560-740	1010	430-580
Ringnorm	2	20	400	196-204	7000	3540-3460
F.Solar	2	9	666	293-373	400	184-216
Splice	2	60	1000	525-475	2175	1123-1052
Thyroid	2	5	140	97-43	75	53-22
Titanic	2	3	150	104-46	2051	1386-66
Twonorm	2	20	400	186-214	7000	3511-3489
Waveform	2	21	400	279-121	4600	3074-1526
Coil20	20	1024	1440	70	40	2
Text	2	7511	1946	959-937	50	26-24
Uspst	10	256	2007	267-201 -169-192 137-171 169-155-175	50	6-5-9-4-3 3-4-5-5
Iris	3	4	120	40-40-40	30	10-10-10
Bupa	2	6	276	119-157	69	26-43
Pima	2	8	615	398-217	153	102-51
Breast-W	2	9	546	353-193	137	91-46

CL = number of classes, TR = training set, TE = test set, VA = number of variables, Cl.Inst. = number of instances in each class.

- Twelve data sets from Raetsch's repository, already used in [34], collected from the UCI, DELVE, and STATLOG benchmark repositories. For each experiment, the 100 (20 for Splice and Image) partitions of each data set into training and test sets contained in the repository are used.
- Three data sets from Chapelle's repository previously used in [10]: Coil20, consisting of gray-scale images of 20 different objects taken from different angles, in steps of 5 degrees; Uspst, the test data part of the USPS data on handwritten digit recognition; and Text consisting of the classes "mac" and "mswindows" of the Newsgroup20 data set. For each experiment, the 10 partitions of each data set into training and test sets from the repository are used.
- Four standard benchmark data sets from the UCI Machine Learning repository: Iris, Bupa, Pima, and Breast-W. For each experiment, 100 random partitions of each data set into training (80 percent of the data) and test (20 percent of the data) sets are used.

4.4 Results

Cross validation is applied to each data set. For each partition of the data set, the instance selection algorithm is applied to the training set from which a subset S is returned. 1NN that uses only points of S is applied to the test set.

In order to assess whether differences in accuracy and storage reduction on all runs of the entire group of data sets are significant, a nonparametric paired test, the Wilcoxon signed rank test, for zero median is applied to compare CC with 1NN and KNN (see Table 2) and CCIS with ICF and DROP3 (see Table 3). As shown, for instance, in [17], comparison of the performance of two algorithms based on the t-test is only indicative because the assumptions of the test are not satisfied, and the Wilcoxon test is shown to provide more reliable estimates. The value in the row labeled "Wilcoxon p" of the tables indicates the resulting p-value.

Results are reported in Tables 2 and 3. All instance selection algorithms are tested using one neighbor. The average accuracy on the test set over the given partitions is reported for each algorithm. The average percentage of instances that are excluded from S is also reported under the column with label R.

TABLE 2
Results of Experiments

Data	1NN	KNN	CC	R
Banana	86.4	88.3	88.3	39.7
g50c	79.6	92.0	87.4	71.2
g10n	75.2	78.2	76.0	63.7
B.Cancer	67.3	71.2	69.6	53.1
Diabetes	69.9	72.6	73.2	55.6
German	70.5	74.1	73.6	59.1
Heart	76.8	81.8	81.5	55.6
Image	96.6	95.1	94.4	42.5
Ringnorm	65.0	56.7	66.6	64.5
F.Solar	60.8	62.3	64.3	79.9
Splice	71.2	73.4	72.6	72.4
Thyroid	95.6	92.7	93.4	40.9
Titanic	67.0	74.1	76.9	81.7
Twonorm	93.3	96.2	95.7	60.8
Waveform	84.2	87.3	86.4	59.9
Coil20	100.0	99.3	98.5	29.9
Text	92.8	92.0	89.4	54.2
Uspst	94.6	92.2	91.8	46.9
Iris	95.5	96.0	95.4	41.9
Breast-W	95.9	97.3	97.1	55.6
Bupa	61.7	66.1	66.4	60.6
Pima	67.3	71.4	71.1	53.3
Average	80.2	82.4	82.3	56.5
Median	78.9	83.4	83.1	55.6
Wilcoxon p	0.0170	0.1779	n/a	n/a

TABLE 3
Results of Experiments with State-of-the-Art Algorithms

Data	CCIS	R	ICF	R	DROP3	R
Banana	87.7	69.6	86.1	79.2	87.6	68.2
g50c	87.6	74.3	82.2	56.3	82.8	77.7
g10n	76.4	67.5	73.0	53.9	75.0	71.4
B.Cancer	62.5	84.5	67.0	79.0	69.7	72.9
Diabetes	71.7	76.1	69.8	83.1	72.3	73.4
German	69.0	85.8	68.6	82.2	72.0	74.3
Heart	80.5	71.9	76.7	80.9	80.2	72.1
Image	88.8	74.2	93.8	80.3	95.1	64.9
Ringnorm	63.2	74.7	61.2	85.5	54.7	80.6
F.Solar	61.5	84.7	61.0	52.0	61.4	93.8
Splice	72.3	75.9	66.3	85.5	67.6	79.0
Thyroid	93.1	81.1	91.9	85.6	92.7	65.7
Titanic	75.4	85.5	67.5	54.3	67.7	94.3
Twonorm	95.2	77.8	89.2	90.7	94.3	72.7
Waveform	85.9	83.5	82.1	86.8	84.9	73.6
Coil20	99.0	46.1	98.5	42.6	95.5	64.4
Text	89.2	55.9	88.2	68.8	88.0	66.7
Uspst	91.8	58.4	86.2	87.8	91.4	67.3
Iris	94.7	78.4	95.3	69.7	95.5	66.4
Breast-W	96.7	91.7	95.4	93.8	96.8	74.2
Bupa	64.3	73.8	60.9	74.3	63.1	73.8
Pima	68.6	76.5	67.9	78.7	69.4	73.3
Average	80.7	74.9	78.6	75.0	79.9	73.7
Median	83.2	76.0	79.4	79.8	81.5	73.1
Wilcoxon p	n/a	n/a	0.0037	0.5922	0.1396	0.5057

Results of Table 2 show that CC outperforms 1NN and achieves accuracy performance similar to that of KNN with number K of neighbors selected by leave-one-out cross validation. Moreover, CC significantly reduces the size of the training set. The test accuracy of CC is significantly better than the one of CCIS, ICF, and DROP3 (application of the Wilcoxon test gives 0 p-value). This may be due to the more aggressive storage reduction performed by these algorithms.

Table 3 reports results of CCIS, ICF, and DROP3. CCIS outperforms ICF and achieves accuracy performance similar to that of DROP3. Storage reduction of the three algorithms does not differ significantly.

Finally, we briefly address the stability issue, that is, whether removing a small fraction of the data affects the resulting instance set significantly. Results of experiments on the considered data sets show stability of the method. This can be due to the fact that when the data set is not too small, then removing a small fraction of the data does not affect significantly the in-degree of the remaining points in G_{bc} and G_{wc}.

Given the diversity of the characteristics of the data sets considered in the experiments, the results provide experimental evidence of the effectiveness of the proposed large margin instance selection algorithm based on *ccnn* for improving the performance of the 1NN rule.

5 Conclusion

This paper proposed *ccnn*, a combined proximity-label-based relation over pairs of instances. A graph-based framework was used for analyzing its relation with the popular nearest neighbor and nearest unlike neighbor concepts and developing a large margin-based algorithm for instance selection. The proposed instance selection method can be interpreted as a novel large-margin-based procedure for training Voronoi networks [26].

An extensive comparative experimental analysis with state-of-the art instance selection methods provided empirical evidence of the effectiveness of the proposed technique for enhancing the performance of the 1NN rule.

In general, the results of this paper show that *ccnn* provides a useful tool for defining and analyzing properties of a training set related to the performance of the 1NN rule.

In future work, we intend to investigate the application of the proposed graph-based framework for developing novel machine learning algorithms: for instance, for feature selection, by seeking a set of features that maximizes a merit criterion based on CC, and for distance learning, in the style of [18], [32].

Furthermore, it would be interesting to investigate whether the proposed scoring function could be used to define a data-driven measure of training set difficulty [36].

Finally, note that the analysis conducted in the present paper uses only the in-degree of nodes. It remains to be investigated whether other graph-theoretic properties, such as path distance, clustering coefficient, and diameter, correspond to interesting properties of the training set related to the performance of 1NN-based classifiers.

Acknowledgments

The author thanks the editor and the reviewers for their constructive comments and the ML and SNN groups at the Radboud University for useful discussions. This work was partially supported by the NWO project 639.023.604.

References

[1] D.W. Aha, D. Kibler, and M.K. Albert, "Instance-Based Learning Algorithms," *Machine Learning*, vol. 6, pp. 37-66, 1991.
[2] F. Angiulli, "Fast Nearest Neighbor Condensation for Large Data Sets Classification," *IEEE Trans. Knowledge and Data Eng.*, vol. 19, no. 11, pp. 1450-1464, Nov. 2007.
[3] F. Angiulli and G. Folino, "Distributed Nearest Neighbor-Based Condensation of Very Large Data Sets," *IEEE Trans. Knowledge and Data Eng.*, vol. 19, no. 12, pp. 1593-1606, Dec. 2007.
[4] V. Barnett, "The Ordering of Multivariate Data," *J. Royal Statistical Soc.*, vol. 139, no. 3, pp. 318-355, 1976.
[5] P.L. Bartlett, "For Valid Generalization, the Size of the Weights Is More Important than the Size of the Network," *Advances in Neural Information Processing Systems 9*, pp. 134-140, The MIT Press, 1997.
[6] B. Hammer, M. Strickert, and T. Villmann, "On the Generalization Ability of GRLVQ Networks," *Neural Processing Letters*, vol. 21, no. 2, pp. 109-120, 2005.

[7] H. Brighton and C. Mellish, "On the Consistency of Information Filters for Lazy Learning Algorithms," *Proc. Third European Conf. Principles of Data Mining and Knowledge Discovery*, pp. 283-288, 1999.
[8] H. Brighton and C. Mellish, "Advances in Instance Selection for Instance-Based Learning Algorithms," *Data Mining and Knowledge Discovery*, vol. 6, pp. 153-172, 2002.
[9] R.M. Cameron-Jones, "Instance Selection by Encoding Length Heuristic with Random Mutation Hill Climbing," *Proc. Eighth Australian Joint Conf. Artificial Intelligence*, pp. 99-106, 1995.
[10] O. Chapelle and A. Zien, "Semi-Supervised Classification by Low Density Separation," *Proc. 10th Int'l Workshop Artificial Intelligence and Statistics*, pp. 57-64, 2005.
[11] C. Cortes and V. Vapnik, "Support Vector Networks," *Machine Learning*, vol. 20, pp. 273-297, 1995.
[12] T. Cover and P. Hart, "Nearest Neighbor Pattern Classification," *IEEE Trans. Information Theory*, vol. 13, no. 1, pp. 21-27, Jan. 1967.
[13] K. Crammer, R. Gilad-Bachrach, A. Navot, and N. Tishby, "Margin Analysis of the LVQ Algorithm," *Proc. Conf. Neural Information Processing Systems*, pp. 462-469, 2002.
[14] B.V. Dasarathy, "Minimal Consistent Set (mcs) Identification for Optimal Nearest Neighbor Decision Systems Design," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 24, no. 3, pp. 511-517, Mar. 1994.
[15] B.V. Dasarathy, "Fuzzy Understanding of Neighborhoods with Nearest Unlike Neighbor Sets," *Proc. SPIE*, pp. 34-43, 1995.
[16] B.V. Dasarathy, "Nearest Unlike Neighbor (NUN): An Aid to Decision Confidence Estimation," *Optical Eng.*, vol. 34, no. 9, pp. 2785-2792, 1995.
[17] J. Demsar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *J. Machine Learning Research*, vol. 7, pp. 1-30, 2006.
[18] C. Domeniconi, J. Peng, and D. Gunopulos, "Locally Adaptive Metric Nearest Neighbor Classification," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 9, pp. 1281-1285, Sept. 2002.
[19] Y. Freund and R.E. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *J. Computer and System Sciences*, vol. 55, no. 1, pp. 119-139, 1997.
[20] S. García, J. Ramón Cano, and F. Herrera, "A Memetic Algorithm for Evolutionary Prototype Selection: A Scaling up Approach," *Pattern Recognition*, vol. 41, no. 8, pp. 2693-2709, 2008.
[21] R. Gilad-Bachrach, A. Navot, and N. Tishby, "Margin Based Feature Selection—Theory and Algorithms," *Proc. Int'l Conf. Machine Learning*, 2004.
[22] P.J. Grother, G.T. Candela, and J.L. Blue, "Fast Implementation of Nearest Neighbor Classifiers," *Pattern Recognition*, vol. 30, pp. 459-465, 1997.
[23] P.E. Hart, "The Condensed Nearest Neighbor Rule," *IEEE Trans. Information Theory*, vol. 14, no. 3, pp. 515-516, May 1968.
[24] N. Jankowski and M. Grochowski, "Comparison of Instances Selection Algorithms II. Results and Comments," *Artificial Intelligence and Soft Computing*, pp. 580-585, Springer, 2004.
[25] J.M. Kleinberg, "Two Algorithms for Nearest-Neighbor Search in High Dimensions," *Proc. 29th ACM Symp. Theory of Computing*, pp. 599-608, 1997.
[26] K. Krishna, M.A.L. Thathachar, and K.R. Ramakrishnan, "Voronoi Networks and Their Probability of Misclassification," *IEEE Trans. Neural Networks*, vol. 11, no. 6, pp. 1361-1372, Nov. 2000.
[27] L.I. Kuncheva and L.C. Jain, "Nearest Neighbor Classifier: Simultaneous Editing and Feature Selection," *Pattern Recognition Letters*, vol. 20, nos. 11-13, pp. 1149-1156, 1999.
[28] J. Lin, "Divergence Measures Based on the Shannon Entropy," *IEEE Trans. Information Theory*, vol. 37, no. 1, pp. 145-151, Jan. 1991.
[29] E. Marchiori, "Hit Miss Networks with Applications to Instance Selection," *J. Machine Learning Research*, vol. 9, pp. 997-1017, 2008.
[30] R.B. McCammon, "Map Pattern Reconstruction from Sample Data; Mississippi Delta Region of Southeast Louisiana," *J. Sedimentary Petrology*, vol. 42, no. 2, pp. 422-424, 1972.
[31] R. Paredes and E. Vidal, "Learning Weighted Metrics to Minimize Nearest-Neighbor Classification Error," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 7, pp. 1100-1110, July 2006.
[32] J. Peng, D.R. Heisterkamp, and H.K. Dai, "Adaptive Quasiconformal Kernel Nearest Neighbor Classification," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 656-661, May 2004.
[33] E. Pkalska, R.P.W. Duin, and P. Paclík, "Prototype Selection for Dissimilarity-Based Classifiers," *Pattern Recognition*, vol. 39, no. 2, pp. 189-208, 2006.
[34] G. Rätsch, T. Onoda, and K.-R. Müller, "Soft Margins for AdaBoost," *Machine Learning*, vol. 42, no. 3, pp. 287-320, 2001.
[35] G.L. Ritter, H.B. Woodruff, S.R. Lowry, and T.L. Isenhour, "An Algorithm for a Selective Nearest Neighbor Decision Rule," *IEEE Trans. Information Theory*, vol. 21, no. 6, pp. 665-669, Nov. 1975.
[36] J.S. Sánchez, R.A. Mollineda, and J.M. Sotoca, "An Analysis of How Training Data Complexity Affects the Nearest Neighbor Classifiers," *Pattern Analysis and Applications*, vol. 10, no. 3, pp. 189-201, 2007.
[37] R.E. Schapire, Y. Freund, P.L. Bartlett, and W.S. Lee, "Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods," *Proc. 14th Int'l Conf. Machine Learning*, pp. 322-330, 1997.
[38] M.A. Tahir, A. Bouridane, and F. Kurugollu, "Simultaneous Feature Selection and Feature Weighting Using Hybrid Tabu Search/K-Nearest Neighbor Classifier," *Pattern Recognition Letters*, vol. 28, no. 4, pp. 438-446, 2007.
[39] I. Tomek, "An Experiment with the Edited Nearest-Neighbor Rule," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 6, no. 6, pp. 448-452, 1976.
[40] G.T. Toussaint, "Proximity Graphs for Nearest Neighbor Decision Rules: Recent Progress," *Proc. Interface-2002, 34th Symp. Computing and Statistics*, pp. 83-106, 2002.
[41] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.
[42] J. Wang, P. Neskovic, and L.N. Cooper, "Improving Nearest Neighbor Rule with a Simple Adaptive Distance Measure," *Pattern Recognition Letters*, vol. 28, no. 2, pp. 207-213, 2007.
[43] K.Q. Weinberger, J. Blitzer, and L.K. Saul, "Distance Metric Learning for Large Margin Nearest Neighbor Classification," *Proc. Conf. Neural Information Processing Systems*, 2006.
[44] D.L. Wilson, "Asymptotic Properties of Nearest Neighbor Rules Using Edited Data," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 2, no. 3, pp. 408-420, July 1972.
[45] D.R. Wilson and T.R. Martinez, "Instance Pruning Techniques," *Proc. 14th Int'l Conf. Machine Learning*, pp. 403-411, 1997.
[46] D.R. Wilson and T.R. Martinez, "Reduction Techniques for Instance-Based Learning Algorithms," *Machine Learning*, vol. 38, no. 3, pp. 257-286, 2000.