Class Conditional Nearest Neighbor and Large Margin Instance Selection

Elena Marchiori

***Abstract*—The one nearest neighbor (1-NN) rule uses instance proximity followed by class labeling information for classifying new instances. This paper presents a framework for studying properties of the training set related to proximity and labeling information, in order to improve the performance of the 1-NN rule. To this aim, a so-called class conditional nearest neighbor (*c.c.n.n.*) relation is introduced, consisting of those pairs of training instances (a, b) such that b is the nearest neighbor of a among those instances (excluded a) in one of the classes of the training set. A graph-based representation of *c.c.n.n.* is used for a comparative analysis of *c.c.n.n.* and of other interesting proximity-based concepts. In particular, a scoring function on instances is introduced, which measures the effect of removing one instance on the hypothesis-margin of other instances. This scoring function is employed to develop an effective large margin instance selection algorithm, which is empirically demonstrated to improve storage and accuracy performance of the 1-NN rule on artificial and real-life data sets.**

***Index Terms*—Computing Methodologies, Artificial Intelligence, Learning, Heuristics design, Machine learning.**

I. 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 (1-NN) rule classifies an unknown point into the class of the nearest of the training set points. 1-NN is used in many applications because of its intuitive interpretation, flexibility, and simple implementation. Moreover, for all distributions, the 1-NN rule's probability of error is bounded above by twice the Bayes's probability of error [11]. However, 1-NN 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, [17], [20], [30]) or noisy instances (see for instance [8], [37]). In order to tackle these problems, various algorithms have been developed, such as those for instance/prototype selection [1], [2], [3], [8], [19], [22], [31], for feature selection [20], [25], [36], and for distance learning [29], [39], [40].

The 1-NN rule does not rely on knowledge of the underlying data distribution (non-parametric classification), but uses directly proximity followed by class labeling information for classifying new points. This paper focusses on the representation and analysis of proximity and labeling properties of the training set related to the performance of the 1-NN rule.

To this aim, we introduce a relation called class conditional nearest neighbor (*c.c.n.n.* in short), defined on pairs of points from a labeled training set as follows. A pair of points (a, b) of the training set satisfies *c.c.n.n.* if b is the nearest neighbor of a *among those points (different from a) in one of the classes of the training set*. This relation describes proximity information *conditioned* to the class label, for each class of the training set.

A graph-based framework representing *c.c.n.n.*, called Hit Miss Network (HMN) [27], is used for analyzing and applying *c.c.n.n.* as follows.

- HMN is partitioned into a so-called hit- (HN) and miss- (MN) network, characterizing the nearest neighbor relation within and between classes, respectively. These graphs are used to study the relation between *c.c.n.n.* and the two popular proximity concepts of nearest neighbor and nearest unlike neighbor [13].
- The in-degree of points in HN and MN are used to analyze how removing one instance from the training set affects the (hypothesis) margin of other instances.
- This analysis is used to introduce a scoring function, called HMscore, which assigns a high score to points whose removal is likely to decrease the margin of many other points.
- HMscore is used to develop a new algorithm for large margin instance selection. Experiments on artificial and real-life data sets, and a comparative analysis of the results with those obtained by two state-of-the-art instance selection algorithms, show that *c.c.n.n.*-based instance selection is an effective approach for improving storage and test accuracy performance of the 1-NN rule.

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

Related Work

To the best of our knowledge, the graph-based analysis of *c.c.n.n.* by means of HMN's and the large margin approach for instance selection provide two original contributions of this paper.

HMN's have been introduced in [27]. In that paper it was shown that structural properties of HMN's correspond to properties of training points related to the decision boundary of the 1-NN rule, such as being border or central point. This observation was used to introduce an instance selection heuristic algorithm for the 1-NN rule based on HMN's.

Here HMN's are used for analyzing the *c.c.n.n.* relation, for studying how removal of points affect the hypothesis margin of other points, and for performing large margin-based instance selection.

Elena Marchiori is with the Department of Computer Science, Radboud University, Nijmegen, The Netherlands. For contact information see http://www.cs.ru.nl/~elenam

Graph representations in the context of 1−NN-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 [37].

A popular relation involving both proximity and class labeling information is the nearest unlike neighbor (NUN) [13], which links one point with its nearest neighbor amongst those points with different class label. NUN has been used in [15] to provide a measure of confidence in the decisions made by the 1−NN based decision systems, and employed in [14] for defining hybrid condensing algorithms.

c.c.n.n. contains both types of label-independent (nearest neighbor) and label-dependent (nearest unlike neighbor) information, as proven in Section II using our graph-based framework.

The proposed application of *c.c.n.n.* to analyze the effect of one point removal on the hypothesis margin of other points, is inspired by works on large margin analysis of prototype selection [12], [6], and feature selection [20]. In [12] 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 [20] hypothesis margin is used to define a loss function for performing feature weighting, and to introduce a large margin bound of the generalization error for the 1−NN rule, which uses a set of selected features.

Here we use the bound given in [12] as guideline for developing a large margin instance selection algorithm, called HMSC. The main differences between HMSC and prototype selection algorithms, such as those analyzed in [12], is that in HMSC prototypes are members of the training set, and the number of prototypes is automatically computed. Indeed, HMSC belongs to the family of instance selection algorithms (see for instance [2], [37], [42]). It differs from previously proposed algorithms for instance selection mainly because it tries to directly enlarge the hypothesis margin. HMSC can be also interpreted as a large margin procedure for training Voronoi networks [24].

The rest of the paper is organized as follows. After presenting the notation used throughout the paper, Section II introduces the *c.c.n.n.* relation, its graph-based representation by means of HMN's, and a comparative analysis of HMN's with NNG and NUN. In Section III *c.c.n.n.* is applied to analyze the effect of removing one instance from the training set on the hypothesis-margin, and used to define an instance scoring function. This scoring function is used in Section IV to develop a large margin instance selection algorithm, whose performance is comparatively analyzed experimentally in Section V on a large collection of data sets. We conclude in Section VI with a summary of the contributions, and point to future work.

Notation

The following notation are used throughout the paper.

- (A, L): a training set, consisting of instances in A with labels in $L = (l_1, \ldots, l_n)$ from c different classes.
- A_i: the set of elements of A with class label i, $i \in \{1, \ldots, c\}$.
- (S, L_S): a collection of pairs of the training set.
- a, b: elements of A, called instances, or points.
- $l(a)$: the class label of a.
- $\texttt{1-NN}(a)$: the nearest neighbor of a.
- $\texttt{1-NN}(a, l)$: the nearest neighbor of a among those points in A_l different from a.
- $\texttt{1-NN}(a, L_1)$ (with $L_1 \subset L$): the nearest neighbor of a among those points in $\cup_{l \in L_1} A_l$ different from a.
- G: a directed graph with nodes representing elements of A.
- $e = (a, b)$: an edge of G, with a the vertex from which e is directed and b the vertex to which e is directed.
- $deg(a)$: the number of edges where a occurs (the degree of a).
- $deg(G)$: the total number of edges of G (the degree of G).
- *in-degree* of a in G ($in\_deg(a, G)$): the number of edges of G pointing to a.
- $|S|$: the number of elements (cardinality) of a set S.
- ϵ^S: here called training or empirical error of S. It is the *leave-one-out* error of (A, L) calculated using the 1−NN rule with S as training set; that is, if $a \in A$ is also in S, then it is classified by the 1−NN rule using as training set S without a.

In the sequel we assume for simplicity that $\texttt{1-NN}(a)$, and $\texttt{1-NN}(a, l)$ are unique, that is, each point has exactly one nearest neighbor (different from a) in A, and in A_l, for each class A_l of the training set.

II. Class Conditional Nearest Neighbor with Hit Miss Networks

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

Definition 2.1 (Class Conditional Nearest Neighbor (c.c.n.n.)): The class conditional nearest neighbor relation on (A,L) is the set

$$\{(a, b) \mid a, b \in A,\ \exists l \in \{1, ..., c\}\ (b = \texttt{1-NN}(a, l))\}.$$

■

We use the following graph-based characterization of *c.c.n.n.*, called Hit Miss Networks [27]. Consider a set of instances A with labels L from c different classes. In an HMN of (A, L), for each label l in L, there is a directed edge from point a to b, its nearest neighbor computed among those points, different from a, having label l. Thus a has c outgoing edges, one for each label. When the labels of a and b are the same, we call (a, b) *hit-edge* and a a *hit of* b, otherwise we call (a, b) a *miss-edge* and a a *miss of* b. The name hit miss network is derived from these terms.

Definition 2.2 (Hit Miss Network): The *Hit Miss Network* of (A, L), denoted by $\texttt{HMN}(A, L)$, is a directed graph $G = (V, E)$ with

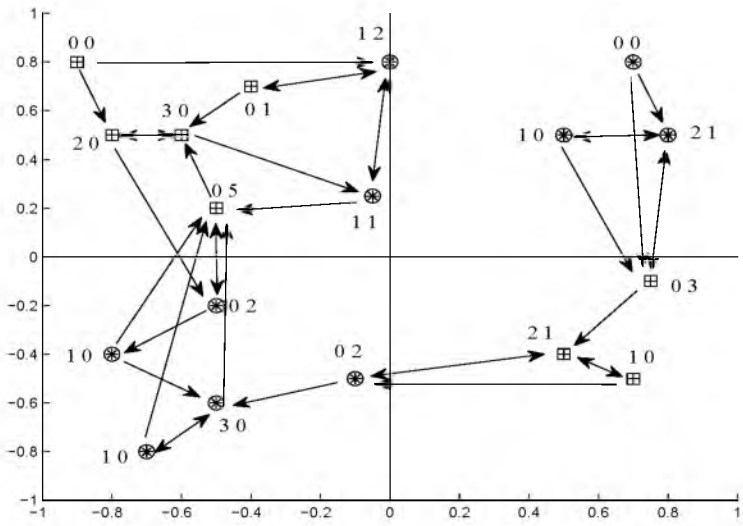

Fig. 1. HMN of a toy data set with two classes, with number of incoming hit and miss edges reported near each node.

- $V = A$ and
- $E = \{(a, 1\text{-NN}(a, l))$ for each $a \in A$ and $l \in \{1, \ldots, c\}\}$.

■

Computing the HMN of a given data set requires quadratic time complexity in the number of points. Nevertheless, by using metric trees or other spatial data structures this bound can be reduced [21], [23]. For instance, using kd trees, whose construction takes time proportional to $nlog(n)$, nearest neighbor search exhibits approximately $O(n^{1/2})$ behavior [21].

Figure 1 shows the HMN of the training set for a toy binary classification task. Observe that the two points with no incoming edges are relatively far from points of the other class. Moreover, points with a high number of miss-edges are closer to the 1−NN decision boundary.

In [27] it was shown that the topological structure of HMN's provides useful information, about points of the training set, related to their vicinity to the decision boundary.

Here we use HMN's for studying class conditional proximity. We start by analyzing the relation between HMN and the popular nearest neighbor graph and nearest unlike neighbor concept. Specifically, we consider the following graphs.

- The hit and miss graphs, which describe the 'within class' and 'between classes' component of *c.c.n.n.*, respectively.
- The nearest neighbor graph, which describes the 'class independent' component of *c.c.n.n.*.
- The nearest unlike neighbor graph, which coincides with the miss graph for binary classification problems.

A. Hit and Miss Networks

Let $G = \text{HMN}(A, L)$. The *hit network*, denoted by $\text{HN}(A, L)$, consists of all the nodes of G and its *hit edges*. Analogously, the *miss network*, denoted by $\text{MN}(A, L)$, consists of all the nodes of G and its *miss edges*. We refer to the in-degree of nodes a in the hit and miss network as *hit* and *miss* degree, denoted by *hit_deg*(a,G) and *miss_deg*(a,G), respectively.

Two subgraphs of a graph are *orthogonal* if they have disjoint sets of edges. The following properties follow directly from the definition of hit and miss network.

Property 2.3: Let $G = \text{HMN}(A, L)$ and $a \in A$.

1) $\text{HN}(A, L)$ and $\text{MN}(A, L)$ are orthogonal subgraphs of G.
2) $G = \text{HN}(A, L) \cup \text{MN}(A, L)$.
3) $in\_deg(a, G) = hit\_deg(a, G) + miss\_deg(a, G)$.
4) $\sum_{a \in A}(miss\_deg(a, G) - (c - 1)hit\_deg(a, G)) = 0$.

■

Figure 1 shows hit and miss degree, plotted on the left- and right-hand side of each point, of a toy training set.

B. Nearest Neighbor Graph

The (directed) nearest neighbor graph (in short NNG(A)) is a popular representation of the nearest neighbor relation over a set of points A, which has been applied with success in studies and applications of the nearest neighbor rule (see, e.g., [37]). For a training set (A, L), such proximity graph consists of A as set of vertices and $\{(a, 1\text{-NN}(a)) \mid a \in A\}$ as set of edges.

The following statements relating NNG and HMN can be directly derived from their definition.

Proposition 2.4: Let (A, L) be a training set.

1) $\text{NNG}(A) \subseteq \text{HMN}(A, L)$.
2) $\text{NNG}(A) = \text{HMN}(A, L)$ if and only if $c = 1$.
3) $\text{NNG}(A) = \text{HN}(A, L)$ if and only if $\epsilon^A = 0$.
4) $\text{NNG}(A)$ represents the *label independent* component of *c.c.n.n.*, that is, $(a, b) \in \text{NNG}(A)$ if and only if $(a, b) \in \text{HMN}(A, L')$ for each L' obtained by applying a permutation to L.

■

We now turn to analyze the relation between HMN and another popular neighbor-based concept, the nearest unlike neighbor.

C. Nearest Unlike Neighbor Graph

The nearest unlike neighbor (NUN) is a useful concept used since decades in diverse application domains, for instance

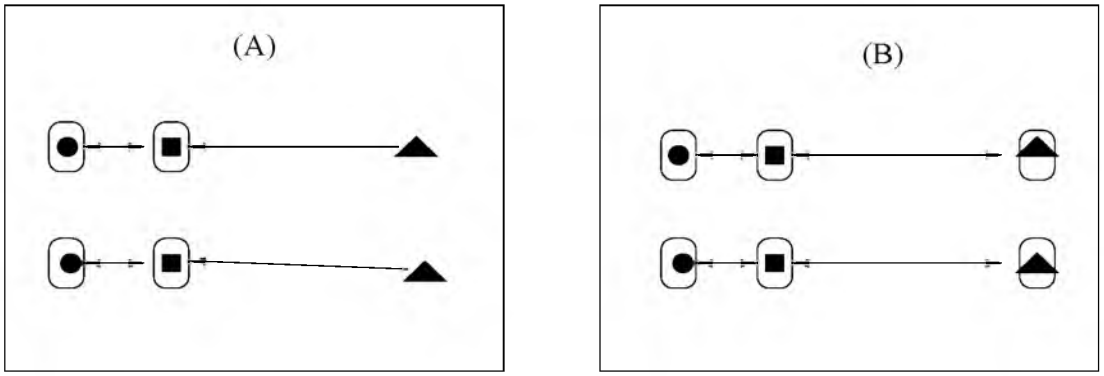

Fig. 2. NUN graph (Figure (A)) and HMN (Figure (B)) of a 3-classes classification problem. NUN and miss points are circled.

geology [28], for detecting border points. Here we consider the NUN concept as defined in [13] in the machine learning community. This concept was further analyzed and applied in [15] to provide a measure of confidence in the decisions made by the 1−NN based decision systems, and employed in [14] for defining hybrid condensing algorithms.

Using our notation, a point b is the *nearest unlike neighbor* (NUN) of a if $b = 1\text{−NN}(a, \cup_{l \neq l(a)}\{l\})$. A point is called NUN *point* if it is the nearest unlike neighbor of at least one point. In graph terms, one can define the NUN graph of (A,L), denoted by NUN(A,L), as the directed graph $G = (V, E)$ such that

- $V = A$ and
- $E = \{(a, b) \mid a \in A, b \text{ is the NUN of } a\}$.

Thus a point $a \in A$ is a NUN point if and only if $in\_deg(a, \text{NUN}(A, L)) > 0$.

If we now call Miss the set of miss points of HMN(A,L), consisting of the set of its vertices with positive miss degree, then example of Figure 2 shows that, for more than two classes, Miss provides more precise information about the 1−NN decision boundary than NUN. Indeed, the Figure shows points of three classes. Part (A) and (B) of the figure shows the NUN and HMN graphs, respectively. Points in NUN and Miss are circled. Note that none of the 'triangle' class points is a NUN. Hence, the NUN set provides a partial description of the 1−NN decision boundary for this training set.

Formally, one can easily verify the following relation between NUN(A, L) and HMN(A, L).

Proposition 2.5: For $i, j \in [1, c], i \neq j$, let $A_{ij} = A_i \cup A_j$ denote the sequence of points of A with class label either i or j, and let L_{ij} the corresponding sequence of class labels. Then

$$\text{MN}(A, L) = \cup_{i,j \in [1,c], i \neq j} \text{NUN}(A_{ij}, L_{ij}).$$

■

This equality shows in particular that NUN and MN graphs coincide for binary classification problems.

The above results show that class conditional nearest neighbor integrates two useful class label independent (NNG) and dependent (NUN) relations.

We turn now to analyze the relation between hit and miss degree of an instance and hypothesis margin, the latter a useful concept used in particular to bound the generalization error of nearest neighbor based classifiers and to develop algorithms for prototype and feature selection (see [12], [20]).

III. Hit Miss Degree and Hypothesis-Margin

Margins play an important role in machine learning research, as a tool for performing theoretic analysis [5], [34], for developing new machine learning algorithms [10], and for improving the accuracy performance of nearest neighbor based classifiers [12], [20].

There are two main definitions of margin of an instance with respect to a classification rule: *sample-margin*, which measures the distance between the instance and the decision boundary induced by the classifier, and *hypothesis-margin*, which is the distance between the hypothesis and the closest hypothesis that assigns alternative label to the given instance. For the 1−NN rule, in [12] it was shown that the hypothesis-margin lower bounds the sample-margin, and that the hypothesis-margin of an instance a with respect to a training set (A, L) 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 label, and $||\cdot||$ denotes the Euclidean norm.

Sample- and hypothesis- margins have been used for constructing theoretic generalization bounds as well as for developing machine learning algorithms (see for instance [5], [12], [18], [20]).

Here we use margins for the latter task, namely as guideline for developing an instance selection algorithm for the 1−NN rule as follows.

We seek a 'simple' hypothesis (S, L_S), consisting of a small subset of points of A and its labels, yielding a 1−NN large hypothesis-margin. We start from the biggest possible hypothesis, namely A, and measure the effect on the hypothesis-margin of removing each point from A. Then we use the resulting measurements for ranking instances, and use the ranking to construct a small (S, L_S), incrementally.

The procedure employs hit and miss degrees as follows. We begin by showing how the hit and miss degree of an instance can be used to quantify the effect of its removal on the hypothesis-margin of the remaining instances. Recall that

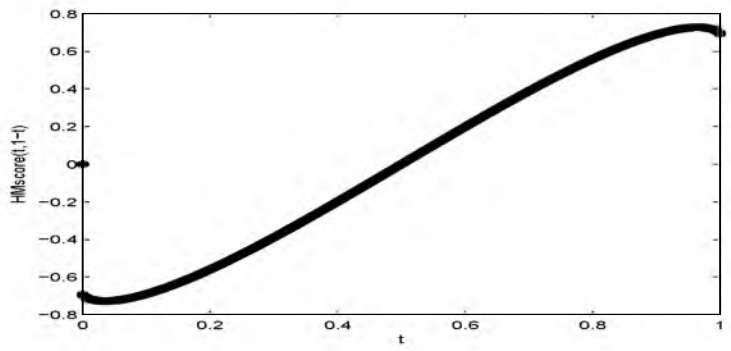

Fig. 3. Plot of $\texttt{HMscore}(t, 1-t)$, with $t \in [0, 1]$.

here the margin of a point is calculated with respect to the training set by excluding that point ('leave-one-out margin'), as in [20].

Proposition 3.1: Let $G = \texttt{HMN}(A, L)$ and $a \in A$. The following mutually exclusive statements hold.

1) If $hit\_deg(a, G) = 0$ and $miss\_deg(a, G) > 0$ then deleting a will increase the hypothesis-margin of $miss\_deg(a, G)$ other points.
2) If $in\_deg(a, G) = 0$ then removing a will not affect the hypothesis-margin of any other point.
3) If $hit\_deg(a, G) > 0$ and $miss\_deg(a, G) > 0$ then removing a will increase the hypothesis-margin of some points and decrease the one of others.
4) If $hit\_deg(a, G) > 0$ and $miss\_deg(a, G) = 0$ then deleting a will decrease the hypothesis-margin of $hit\_deg(a, G)$ other points.

Proof: Let $\mathcal{L}(A) = \sum_{a \in A} \theta_A(a)$. One can easily check the validity of the following equality.

$$\mathcal{L}(A) = \sum_{\substack{a \in A \ s.t. \\ miss\_deg(a) > 0}} \sum_{\substack{b \in A \ s.t. \\ (b,a) \in \texttt{MN}(A,L)}} ||b - a|| - \sum_{\substack{a \in A \ s.t. \\ hit\_deg(a) > 0}} \sum_{\substack{c \in A \ s.t. \\ (c,a) \in \texttt{HN}(A,L)}} ||c - a||.$$

Then the claims follow directly from such equality. ■

This case analysis motivates the introduction of the following scoring function, which scores each point according to how its removal influences the hypothesis-margin of the other training set points. If the removal of one point from the training set yields a decrease of the hypothesis-margin of many other points, then that point is considered relevant and will receive a large score.

Definition 3.2 (Hit-Miss Score): Let a in (A, L). The hit-miss score of a is

$$\texttt{HMscore}(p_a^h, p_a^m) = p_a^h \log \frac{p_a^h}{\frac{1}{2}p_a^h + \frac{1}{2}p_a^m} - p_a^m \log \frac{p_a^m}{\frac{1}{2}p_a^h + \frac{1}{2}p_a^m},$$

where p_a^h and p_a^m are the hit and miss degree of point a divided by the total degree of HN and MN, respectively. ■

Figure 3 shows a plot of the hit-miss score function for $(p_a^h, p_a^m) = (t, 1-t)$ with $t \in [0, 1]$.

The proposed definition of hit-miss score is related to a directed information theoretic divergence measure based on the Shannon entropy, known as K-divergence (see for instance [26]). Indeed, when taking the sum over $a \in A$ of the hit-miss scores, one obtains the difference of the K-divergences of the (normalized) hit and miss degrees of the points, that is,

$$\sum_{a \in A} \texttt{HMscore}(p_a^h, p_a^m) = K(p^h, p^m) - K(p^m, p^h),$$

where $p^h = (p_1^h, \ldots, p_n^h)$, $p^m = (p_1^m, \ldots, p_n^m)$ and

$$K(p^h, p^m) = \sum_{i=1}^{n} p_i^h \log \frac{p_i^h}{\frac{1}{2}p_i^h + \frac{1}{2}p_i^m}.$$

Observe that the K-divergence is a non-symmetric, bounded divergence measure [26]. The above difference of K-divergences can be interpreted as a global measure of relevance of a training set, with respect to the effect of point removal on the hypothesis-margin.

Figure 4 shows the HMscore of instances for a toy training set.

The following observations can be derived directly from the definition of hit-miss score.

1) A point with zero hit-degree (cases 1 or 2 of Proposition 3.1) has zero or negative hit-miss score;
2) points with zero miss-degree and positive hit-degree have positive hit-miss score (case 4);
3) points with higher hit- than miss- degree have positive score, while points with higher miss- than hit- degree have negative hit-miss score (case 3).

These observations justify the application of hit-miss scores for ranking points in order to select a small subset of 'relevant' points, forming a large margin hypothesis. In the next section we introduce an heuristic algorithm based on such a ranking.

IV. Large Margin Instance Selection with Hit-Miss Scores

In large margin classifiers, such as support vector machines, one aims at finding an hypothesis yielding maximal (sample) margin and low training error. In particular, in [12] it is shown that a popular prototype selection algorithm, *Learning Vector Quantization* (LVQ), belongs to a family of maximal margin algorithms [18], [38], with different instances obtained by different choices of loss functions.

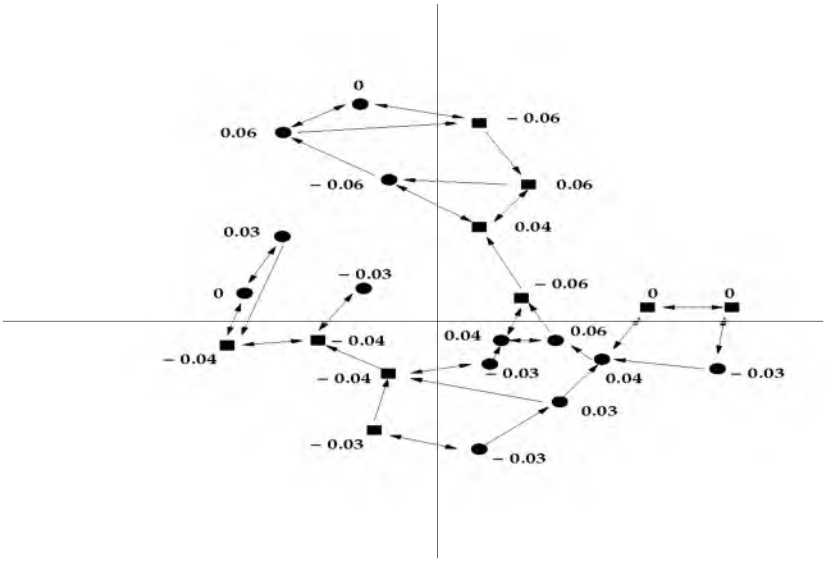

Fig. 4. A training set and HMscore of its points.

An LVQ-based 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 instance selection, prototypes are constrained to be members of the training set, and the selection process is typically based on heuristic search (see for instance [42]). Note that instance selection can be interpreted as training process for a family of learning machines, also known in the literature as Voronoi networks [24]. Since LVQ and Voronoi networks are different, and we are not aware of large margin bounds results for Voronoi networks, we use theoretical results on LVQ as 'guideline' for developing a margin-based instance selection algorithm.

Specifically, the following large margin bound on the generalization error provides the guideline for developing a large-margin based algorithm for instance selection. Below, $margin_\mu(a)$ denotes the sample margin of a.

Theorem 4.1 ([12]): Let (A, L) be a training set drawn from some underlining distribution $\mathcal{D}$ on $\mathbb{R}^m$, with $|A| = n$. Assume that $||a|| \leq R$. Let μ be a set of prototypes with k prototypes for each class. Let $0 < \theta < 1/2$. Let $\alpha^\theta_{(A,L)}(\mu) = \frac{1}{n}|\{a \mid margin_\mu(a) < \theta\}|$. Let $e_\mathcal{D}(\mu)$ be the generalization error. Let $\delta > 0$.

Then with probability $1 - \delta$ over the choices of the training set:

$$\forall \mu \quad e_\mathcal{D}(\mu) \leq \alpha^\theta_{(A,L)}(\mu) + \sqrt{\frac{8}{n}\Big(d \log^2 \frac{32n}{\theta^2} + \log \frac{4}{\delta}\Big)},$$

where d is the VC dimension

$$d = \min\Big(m + 1, \frac{64R^2}{\theta^2}\Big) 2k^c \log(ek^2).$$

In [6] a large margin bound was given for a more general class of LVQ classifiers with adaptive metric.

From the above result it follows that if a prototype selection algorithm selects a small set of prototypes with large margin θ and small θ-error, the bound guarantees that the resulting 1−NN rule will generalize well [12].

In our approach to instance selection, the three objectives, namely small number of instances, large hypothesis-margin, and small θ-error, are incorporated into the search strategy of the algorithm as follows.

- **Small number of instances.** The algorithm expands an initial small core S of points by adding incrementally to S one point at each iteration until a suitable stopping criterion is met.
- **Large hypothesis-margin.** The order in which instances are selected is the one induced by the sorted (in descendent order) list of hit-miss scores.
- **Small θ-error.** The iterative process continues while the empirical error ϵ^S decreases, until ϵ^S becomes sufficiently small.

Figure 5 shows the pseudo-code of the algorithm, called HMscore selection. The training error ϵ^S is considered sufficiently small if it is smaller than ϵ^A. The core subset used to initialized S is set to the first k_0 points of the sorted sequence, with $k_0 \geq c$. This parameter allows the user to set a reasonable lower bound on the hypothesis size.

The algorithm is applied to c pairs of classes selected as follows, and the union of the resulting selected instances is given as final result of HMscore selection. For each class i, the class j most similar to i is selected. Here similarity between class i and j is defined as the correlation between the hit- and miss- degrees of class i in $\mathtt{HMN}(A_{ij}, L_{ij})$. This definition is based on the observations on topological properties of HMN's shown in [27]. From $\mathtt{HMN}(A, L) = \cup_{i,j}\ \mathtt{HMN}(A_{ij}, L_{ij})$, it follows that c suitable components of *c.c.n.n.* are used by HMscore selection.

Figure 6 (left plot) illustrates the effect of the algorithm on a training set of the XOR classification problem. In particular, it shows that HMscore selection discards isolated instances (with zero in-degree, see line 7 of the code) as well as instances close to the 1−NN decision boundary.

In order to simplify the output hypothesis, that is, reduce

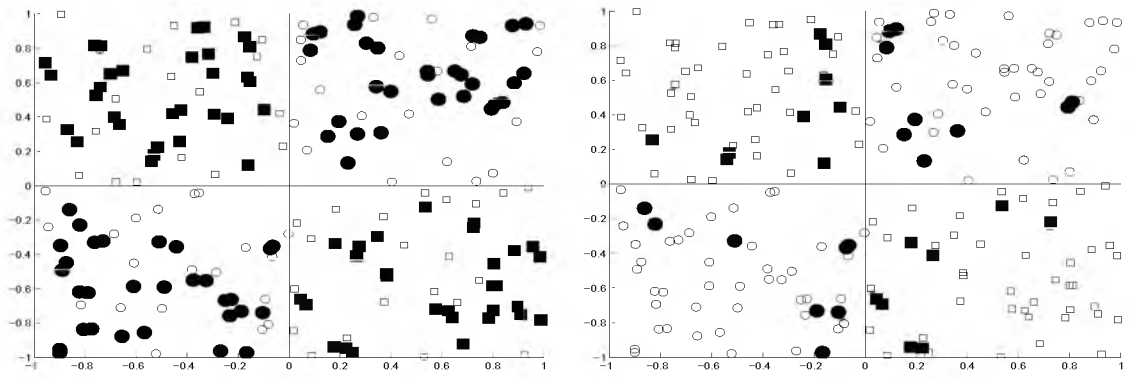

Fig. 6. Application of HMscore selection (left plot) and of Carving (right plot). The points selected have filled markings.

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

Fig. 5. Pseudo-code of HMscore selection. Input: training set (A, L). Output: subset (S, L_S) of the training set.

the number of selected instances, we propose in the sequel a procedure, called Carving.

A. Hypothesis Carving

Pseudo-code of Carving is given in Figure 7.

The algorithm takes as input the set (S, L_S) of instances produced by HMscore selection, and outputs the set (S_o, L_{S_o}) constructed as follows.

S_o is initially equal to those points close to the 1-NN decision boundary of S, that is, having positive miss degree in $\texttt{HMN}(S, L_S)$. Layers consisting of points close to the decision boundary are then iteratively added to S_o (lines 5-13), where a layer is obtained as follows.

First remove the actual set S_o from S (the resulting set is denoted by S_1, initialized in line 3, and updated in line 11), next select those points with positive miss degree in $\texttt{HMN}(S_1, L_{S_1})$ which were not 'isolated' in the hit miss network of the previous iteration, that is, they had positive in-degree in $\texttt{HMN}(S_{prev}, L_{S_{prev}})$ (see line 6). The process is iterated while the empirical error of S_o decreases and the size S_o increases (line 7).

The effect of the Carving on the XOR example is illustrated in the right part of Figure 6, which shows the neat carving effect of the procedure.

1: $S_o = \{a \in S \mid miss\_deg(a, \texttt{HMN}(S, L_S)) > 0\}$;
2: $S_{prev} = S$;
3: $S_1 = S \setminus S_o$;
4: go_on =1;
5: **while** go_on **do**
6: $S_t = \{a \in S_1 \mid miss\_deg(a, \texttt{HMN}(S_1, L_{S_1})) > 0$ and $in\_deg(a, \texttt{HMN}(S_{prev}, L_{S_{prev}})) > 0\}$
7: go_on = $\epsilon^{S_o \cup S_t} < \epsilon^{S_o}$ and $|S_t| > 0$;
8: **if** go_on **then**
9: $S_o = S_o \cup S_t$;
10: $S_{prev} = S_1$;
11: $S_1 = S \setminus S_o$;
12: **end if**
13: **end while**

Fig. 7. Pseudo-code of Carving. Input: the set (S, L_S) of instances selected by HMscore. Output: set (S_o, L_{S_o}) of instances from (S, L_S).

1) Computational Complexity: Constructing HMN and sorting the hit-miss scores takes time proportional to $n \log(n)$. The iterative process, in the worse 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 the algorithm performs at most $n - k_0 + 1$ iterations. Then the worse case run time complexity of HMscore selection is $O(max(n(n - k_0 + 1), n \log(n))$. The carving procedure does not increase the computational complexity of the algorithm. Experimental evidence on real-life data sets shows that a small number of iterations is performed in

practice.

V. Experiments

In order to assess the performance of the proposed instance selection algorithm, experiments on artificial and real-life data sets are conducted, publicly available at Raetsch's benchmark repository [1], Chapelle's repository [2], and UCI Machine Learning repository [3]. In all the experiments we set $k_0 = max(c, \lceil 0.5\epsilon^A + 1 \rceil)$, that is, we choose an initial core of instances of size proportional to the difficulty of the task, as measured by the empirical error of (A, L). This choice is heuristic. Application of automatic parameter tuning procedures, for instance internal cross validation, may possibly improve the performance of the algorithm.

A. Artificial Data Sets

We consider three artificial data sets with different characteristics (see Table I). The Banana data set from Raetsch's benchmark repository, consisting of 100 partitions of the data set into training and test set. Two data sets from Chapelle's benchmark data [9]: `g50c` and `g10n`, generated from two standard normal multi-variate 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%. 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 set from Chapelle's repository are used.

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

TABLE I
CHARACTERISTICS OF ARTIFICIAL DATA SETS. CL = NUMBER OF CLASSES, TR = TRAINING SET, TE = TEST SET, VA = NUMBER OF VARIABLES, CL.INST. = NUMBER OF INSTANCES IN EACH CLASS.

B. Real-life Data Sets

The following 19 publicly available real-life data sets are used, whose characteristics are shown in Table II.

- 12 data sets from Raetsch's repository, already used in [32], 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 set contained in the repository are used.
- 3 data sets from Chapelle's repository previously used in [9]: `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 set from the repository are used.
- 4 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% of the data) and test (20% of the data) set are used.

Data	CL	VA	TR	Cl.Inst.	TE	Cl.Inst.
B.Cancer	2	9	200	140-60	77	56-21
Diabetis	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

TABLE II
CHARACTERISTICS OF REAL-LIFE DATA SETS. CL = NUMBER OF CLASSES, TR = TRAINING SET, TE = TEST SET, VA = NUMBER OF VARIABLES, CL.INST. = NUMBER OF INSTANCES IN EACH CLASS.

C. Results

We perform experiments on these data sets with four algorithms:

- the `1-NN` that uses the entire training set,
- `HMSC`, consisting of `HMscore` selection followed by `Carving`,
- *Iterative Case Filtering* (`ICF`) introduced in [7],
- the best performing of the *Decremental Reduction Optimization* algorithms, `DROP3` [41], [42].

We refer the reader to [27], [42] for a description of `ICF` and `DROP3` and the motivation for choosing these two algorithms as representatives of state-of-the-art instance selection methods.

Cross validation is applied to each data set. For each partition of the data set, each editing algorithm is applied to the training set from which a subset S is returned. `1-NN` that uses only points of S is applied to the test set. All algorithms are tested using one neighbor. The average accuracy on the test set over the given partitions is reported for each algorithm (see Table IV). The average percentage of instances that are excluded from S is also reported under the column with label R.

Results of experiments on artificial data sets are shown in Table III.

[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

Data	1-NN	HMSC	R	ICF	R	DROP3	R
Banana	86.4 +	**87.6**	70.5	86.1 +	**79.2** -	**87.6**	68.2 +
g50c	79.6 +	**86.0**	**89.7**	82.2 +	56.3 +	82.8 +	77.7 +
g10n	75.2 +	**79.6**	**88.2**	73.0 +	53.9 +	75.0 +	71.4 +

TABLE III
RESULTS OF EXPERIMENTS ON ARTIFICIAL DATA SETS.

On these data sets HMSC achieves highest average accuracy, significantly better than that of the other algorithms. On these types of classification tasks, the results indicate robustness of HMSC with respect to the presence of noise (on g50c data) and of irrelevant variables (g10n data).

On real-life data sets, we compare statistically HMSC with each of the other algorithms as follows.

- A paired t-test on the cross validation results on each data set is applied, to assess whether the average accuracy for HMSC is significantly different than each of the other algorithms. In TableIV a '+' indicates that HMSC's average accuracy is significantly higher than the other algorithm at a 0.05 significance level. Similarly, a '-' indicates that HMSC's average accuracy is significantly lower than the other algorithm at a 0.05 significance level. The row labeled 'Sig.acc.+/-' reports the number of times HMSC's average accuracy is significantly better and worse than each of the other algorithms at a 0.05 significance level. A paired t-test is also applied to assess significance of differences in storage reduction percentages for each experiment.
- In order to assess whether differences in accuracy and storage reduction on all runs of the entire group of data sets are significant, a non-parametric paired test, the Wilcoxon Signed Ranks test[4] is applied to compare HMSC with each of the other algorithms. A '+' (respectively '-') in the row labeled 'Wilcoxon' indicates that HMSC is significantly better (respectively worse) than the other algorithm.

Results of the non parametric Wilcoxon test for paired samples at a 0.05 significance level show that on on the entire set of classification tasks HMSC is significantly better than each of the other algorithms with respect to both accuracy and storage reduction. As shown, for instance, in [16], 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.

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 *c.c.n.n.* for improving the performance of the 1-NN rule.

VI. CONCLUSION

This paper proposed *c.c.n.n.*, a combined proximity-label-based relation over pairs of instances. The graph-based HMN framework was used for analyzing its relation with the popular nearest neighbor and nearest unlike neighbor concepts, and for showing how *c.c.n.n.* can be used to study local properties of a training set, in particular the effect of instance removal on the hypothesis-margin of the other instances of the training set, in order to improve the performance of the 1-NN rule.

Specifically, a scoring function for measuring the relevance of points with respect to their effect on the hypothesis-margin of the other points was introduced, and employed to develop a large margin based algorithm for instance selection. 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 1-NN rule.

In general, the results of this paper show that *c.c.n.n.* provides a useful tool for defining and for analyzing properties of a training set related to the performance of the 1-NN 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 hit-miss scores, and for distance learning, in the style of [17], [30].

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 [33].

The proposed instance selection method can be interpreted as a novel large-margin-based procedure for training Voronoi networks [24]. It would be interesting to investigate the effectiveness of such procedure for enhancing training of large margin classifiers, such as support vector machines, as done, for instance, in [35].

Finally, note that the analysis conducted in the present paper uses only the hit and miss degree of HMN's nodes. It remains

[4]Thanks to G. Cardillo for his 'wilcoxon' Matlab routine.

Data	1-NN	HMSC	R	ICF	R	DROP3	R
B.Cancer	67.3 +	**72.3**	**83.1**	67.0 +	79.0 +	69.7 +	72.9 +
Diabetis	69.9 +	**72.6**	**84.4**	69.8 +	83.1 +	72.3	73.4 +
German	70.5 +	**73.6**	**84.6**	68.6 +	82.2 +	72.0 +	74.3 +
Heart	76.8 +	**80.9**	**86.3**	76.7 +	80.9 +	80.2 +	72.1 +
Image	**96.6** -	88.9	74.1	93.8 -	**80.3** -	95.1 -	64.9 +
Ringnorm	**65.0** -	64.8	81.4	61.2 +	**85.5** -	54.7 +	80.6 +
F.Solar	60.8 +	**64.3**	87.1	61.0 +	52.0 +	61.4 +	**93.8** -
Splice	71.2 +	**72.4**	**85.6**	66.3 +	85.5	67.6 +	79.01 +
Thyroid	**95.6** -	93.0	80.3	91.9 +	**85.6** -	92.7 +	65.7 +
Titanic	67.0 +	**76.9**	85.9	67.5 +	54.3 +	67.7 +	**94.3** -
Twonorm	93.3 +	**95.2**	**91.9**	89.2 +	90.7	94.3 +	72.7 +
Waveform	84.2 +	**85.8**	**89.3**	82.1 +	86.8 +	84.9 +	73.6 +
Coil20	**100** -	99.0	46.1	98.5 +	42.6 +	95.5 +	**64.4** -
Text	**92.8** -	89.2	56.3	88.2 +	**68.8** -	88.0 +	66.7 -
Uspst	**94.6** -	91.6	57.7	86.2 +	**87.8** -	91.4	67.3 -
Iris	95.5 -	94.7	**79.0**	95.3 -	69.7 +	**95.5** -	66.4 +
Breast-W	95.9 +	96.4	91.9	95.4 +	**93.8** -	**96.8**	74.2 +
Bupa	61.7 +	**65.8**	**83.1**	60.9 +	74.3 +	63.1 +	73.8 +
Pima	67.3 +	**71.5**	**83.4**	67.9 +	78.7 +	69.4 +	73.3 +
Average	80.3	**81.5**	**79.6**	78.3	76.9	79.6	73.9
Median	78.2	**80.9**	**83.4**	76.7	80.9	80.2	73.3
Sig.+/-	12/6	n/a	n/a	17/2	11/6	14/2	14/5
Wilcoxon	+	n/a	n/a	+	+	+	+

TABLE IV
RESULTS OF EXPERIMENTS ON REAL-LIFE DATA SETS. AVERAGE AND MEDIAN ACCURACY AND TRAINING SET REDUCTION PERCENTAGE FOR EACH ALGORITHM OVER ALL THE DATA SETS IS REPORTED NEAR THE BOTTOM OF THE TABLE.

to be investigated whether other graph-theoretic properties of HMN's, such as path distance, clustering coefficient and diameter, correspond to interesting properties of the training set related to the performance of 1−NN-based classifiers.

ACKNOWLEDGEMENTS

Thanks to the ML and SNN groups at the Radboud University, in particular Tom Heskes, for stimulating discussion on the subject of this paper.

REFERENCES

[1] D.W. Aha, D. Kibler, and M.K. Albert. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[2] F. Angiulli. Fast nearest neighbor condensation for large data sets classification. *IEEE Transactions on Knowledge and Data Engineering*, 19(11):1450–1464, 2007.

[3] F. Angiulli and G. Folino. Distributed nearest neighbor-based condensation of very large data sets. *IEEE Trans. on Knowl. and Data Eng.*, 19(12):1593–1606, 2007.

[4] V. Barnett. The ordering of multivariate data. *J. Roy. Statist. Soc.*, Ser. A 139(3):318–355, 1976.

[5] P. L. Bartlett. For valid generalization, the size of the weights is more important than the size of the network. In *Advances in Neural Information Processing Systems 9*, pages 134–140, 1997.

[6] B.Hammer, M.Strickert, and T.Villmann. On the generalization ability of GRLVQ networks. *Neural Processing Letters*, 21(2):109–120, 2005.

[7] H. Brighton and C. Mellish. On the consistency of information filters for lazy learning algorithms. In *PKDD '99: Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery*, pages 283–288. Springer-Verlag, 1999.

[8] H. Brighton and C. Mellish. Advances in instance selection for instance-based learning algorithms. *Data Mining and Knowledge Discovery*, (6):153–172, 2002.

[9] O. Chapelle and A. Zien. Semi-supervised classification by low density separation. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 57–64, 2005.

[10] C. Cortes and V. Vapnik. Support vector networks. In *Machine Learning*, pages 273–297, 1995.

[11] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.

[12] K. Crammer, R. Gilad-Bachrach, A. Navot, and N. Tishby. Margin analysis of the LVQ algorithm. In *NIPS*, pages 462–469, 2002.

[13] B. V. Dasarathy. Minimal consistent set (mcs) identification for optimal nearest neighbor decision systems design. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(3):511–517, 1994.

[14] B.V. Dasarathy. Fuzzy understanding of neighborhoods with nearest unlike neighbor sets. *SPIE*, 2493:34–43, 1995.

[15] B.V. Dasarathy. Nearest unlike neighbor (nun): An aid to decision confidence estimation. *Opt. Eng.*, 34(9):2785–2792, 1995.

[16] J. Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[17] C. Domeniconi, J. Peng, and D. Gunopulos. Locally adaptive metric nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1281–1285, 2002.

[18] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119139, 1997.

[19] S. García, J. Ramón Cano, and F. Herrera. A memetic algorithm for evolutionary prototype selection: A scaling up approach. *Pattern Recognition*, 41(8):2693–2709, 2008.

[20] R. Gilad-Bachrach, A. Navot, and N. Tishby. Margin based feature selection - theory and algorithms. In *ICML*, 2004.

[21] P.J. Grother, G.T. Candela, and J.L. Blue. Fast implementation of nearest neighbor classifiers. *Pattern Recognition*, 30:459–465, 1997.

[22] N. Jankowski and M. Grochowski. Comparison of instances selection algorithms ii. results and comments. In *Artificial Intelligence and Soft Computing*, pages 580–585. Springer, 2004.

[23] J.M. Kleinberg. Two algorithms for nearest-neighbor search in high dimensions. In *Proc. 29th ACM Symposium on Theory of Computing (STOC)*, pages 599–608, 1997.

[24] K. Krishna, M.A.L. Thathachar, and K.R. Ramakrishnan. Voronoi networks and their probability of misclassification. *IEEE Transactions on Neural Networks*, 11(6):1361 –1372, 2000.

[25] L.I. Kuncheva and L.C. Jain. Nearest neighbor classifier: simultaneous editing and feature selection. *Pattern Recogn. Lett.*, 20(11-13):1149–1156, 1999.

[26] J. Lin. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

[27] E. Marchiori. Hit miss networks with applications to instance selection. *Journal of Machine Learning Research*, 9:997–1017, 2008.

[28] R. B. McCammon. Map pattern reconstruction from sample data; Mississippi Delta region of southeast Louisiana. *Journal of Sedimentary Petrology*, 42(2):422–424, 1972.

[29] R. Paredes and E. Vidal. Learning weighted metrics to minimize nearest-neighbor classification error.

[30] J. Peng, D.R. Heisterkamp, and H.K. Dai. Adaptive quasiconformal kernel nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):656– 661, 2004.

[31] E. Pkalska, R.P. W. Duin, and P. Paclík. Prototype selection for dissimilarity-based classifiers. *Pattern Recogn.*, 39(2):189–208, 2006.

[32] G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320, 2001.

[33] J.S. Sánchez, R.A. Mollineda, and J.M. Sotoca. An analysis of how training data complexity affects the nearest neighbor classifiers. *Pattern Anal. Appl.*, 10(3):189–201, 2007.

[34] R. E. Schapire, Y. Freund, P. L. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. In *Machine Learning: Proceedings of the Fourteenth International Conference*, pages 322–330, 1997.

[35] H. Shin and S. Cho. Neighborhood property based pattern selection for support vector machines. *Neural Computation*, (19):816–855, 2007.

[36] M. A. Tahir, A. Bouridane, and F. Kurugollu. Simultaneous feature selection and feature weighting using hybrid tabu search/k-nearest neighbor classifier. *Pattern Recogn. Lett.*, 28(4):438–446, 2007.

[37] G.T. Toussaint. Proximity graphs for nearest neighbor decision rules: recent progress. In *Interface-2002, 34th Symposium on Computing and Statistics*, pages 83–106, 2002.

[38] V. Vapnik. *The Nature Of Statistical Learning Theory*. Springer-Verlag, 1995.

[39] J. Wang, P. Neskovic, and L.N. Cooper. Improving nearest neighbor rule with a simple adaptive distance measure. *Pattern Recogn. Lett.*, 28(2):207–213, 2007.

[40] K.Q. Weinberger, J. Blitzer, and L.K. Saul. Distance metric learning for large margin nearest neighbor classification. In *In NIPS*. MIT Press, 2006.

[41] D.R. Wilson and T.R. Martinez. Instance pruning techniques. In *Proc. 14th International Conference on Machine Learning*, pages 403–411. Morgan Kaufmann, 1997.

[42] D.R. Wilson and T.R. Martinez. Reduction techniques for instance-based learning algorithms. *Machine Learning*, 38(3):257–286, 2000.