Nearest neighbor representations of Boolean functions [*]

Péter Hajnal[†] Zhihao Liu[‡] György Turán[§]

Abstract Lower and upper bounds are given for the number of prototypes required for various nearest neighbor representations of Boolean functions.

1. Introduction

A nearest neighbor representation of a classification of a set of points in $\mathbf{R}^n$ is given by a set of prototypes such that each point belongs to the same class as the prototype closest to it. More generally, for a k-nearest neighbor representation, the class containing a point is determined by taking the most frequent class label among the k closest prototypes. Nearest neighbor representations are much studied and used in computational geometry, machine learning and pattern recognition (see, for example, Mulmuley [8], Mitchell [7] and Duda et al. [3]).

In general, one tries to use as few prototypes as possible. This leads to questions about the smallest number of prototypes representing a given classification. We consider the special case, suggested by Kasif [6], of binary classifications of the n-dimensional hypercube. A binary classification of the hypercube can be viewed as a Boolean function, and therefore we use this terminology in the rest of the paper. The minimal number of prototypes needed to represent a Boolean function is a complexity measure which is related to other, well-studied complexity measures such as linear decision tree complexity or threshold circuit complexity. Prototypes may be restricted to belong to the set itself, and thus one obtains two versions of the problem. In related work, Wilfong [11] considered the computational complexity of finding a minimal set of prototypes for planar point sets, and Baum [1] considered a probabilistic version for the whole space $\mathbf{R}^n$.

We prove several bounds for the nearest neighbor complexities of Boolean functions. In Section 3 we consider the case when the prototypes are Boolean, and give examples where this restriction

[*] A preliminary version of this paper, *Z. Liu: The nearest neighbor rule representation of Boolean functions*, was presented at the Intel Science Talent Search in 2002.

[†] University of Szeged, Bolyai Institute

[‡] California Institute of Technology

[§] University of Illinois at Chicago, and Research Group on Artificial Intelligence of the Hungarian Academy of Sciences at the University of Szeged. This material is based upon work supported by the National Science Foundation under grants CCR-0100336 and CCF-0431059.

leads to a large increase in the number of prototypes. It is shown in Section 4 that the trivial upper bound of 2^n for the number of prototypes can be improved asymptotically for every function, and an exponential lower bound is proved for almost all functions. We then prove, in Section 5, lower bounds for an explicit function, the *mod* 2 inner product. The lower bound is linear for the nearest neighbor representation, and almost linear for the k-nearest neighbor representation. There are many related open problems. Some of these are mentioned in Section 6.

2. Preliminaries

The Euclidean distance in $\mathbf{R}^n$ (resp., the Hamming distance in $\{0,1\}^n$) is denoted by $d(x,y)$ (resp., $d_H(x,y)$); for $x \in \{0,1\}^n$ it holds that $d(x,y) = \sqrt{d_H(x,y)}$. The componentwise partial order on $\{0,1\}^n$ is denoted by $x \leq y$. If $x \leq y$ then we also say that x is covered by y. For a vector $x = (x_1, \ldots, x_n) \in \{0,1\}^n$, we write $x^{(i)}$ for the vector obtained from x by switching its i'th component, and we write $|x|$ for its weight, i.e., the number of its 1 components. Switching a component 1 in x to 0 we get a lower neighbor of x.

Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. Points x with $f(x) = 1$ (resp., $f(x) = 0$) are called positive (resp., negative).

A *nearest neighbor (NN) representation* of f is a pair of disjoint subsets (P, N) of $\mathbf{R}^n$, such that for every $x \in \{0,1\}^n$ it holds that

- if x is positive then there is a $y \in P$ such that $d(x,y) < d(x,z)$ for every $z \in N$,
- if x is negative then there is a $y \in N$ such that $d(x,y) < d(x,z)$ for every $z \in P$.

The points in P (resp., N) are called positive (resp., negative) *prototypes*. The *size* of the representation is $|P \cup N|$. The nearest neighbor complexity, $NN(f)$, of f is the minimum of the sizes of the representations of f. A nearest neighbor representation is *Boolean* if $P \cup N \subseteq \{0,1\}^n$, i.e., if the prototypes are Boolean vectors. The minimum of the sizes of the Boolean nearest neighbor representations is denoted by $BNN(f)$.

Similarly, a *k-nearest neighbor (k-NN) representation* of f is a pair of disjoint subsets (P, N) of $\mathbf{R}^n$, such that for every $x \in \{0,1\}^n$ it holds that

- x is positive iff at least $\frac{k}{2}$ of the k points in $P \cup N$ closest to x belong to P.

For definiteness, it is assumed that for every x, the k smallest distances of x from the prototypes are all smaller than the other $|P \cup N| - k$ distances from the prototypes. The case $k = 1$ is the same as the nearest neighbor representation. The *size* of the representation is again $|P \cup N|$. The k-nearest neighbor complexity, $k - NN(f)$, of f is the minimum of the sizes of the k-nearest neighbor representations of f.

3. Boolean nearest neighbors

It follows from the definitions by letting all points in $\{0,1\}^n$ be prototypes that for every n-variable Boolean function

$$NN(f) \leq BNN(f) \leq 2^n. \tag{1}$$

The n-variable parity function shows that the second inequality can be an equality, and there can be an exponential gap between the general and Boolean nearest neighbor complexities.

Proposition 1. a) *For every n-variable symmetric function f it holds that $NN(f) \leq n+1$.*

b) *$BNN(x_1 \oplus \cdots \oplus x_n) = 2^n$.*

Proof For part *a)*, let $y_\ell = (\frac{\ell}{n}, \ldots, \frac{\ell}{n})$, for $\ell = 0, \ldots, n$. If $x \in \{0,1\}^n$ has weight w then a direct calculation shows that $d(x, y_w) < d(x, y_\ell)$ for every $\ell \neq w$. Thus $P = \{y_\ell : f(1^\ell 0^{n-\ell}) = 1\}$ and $N = \{y_\ell : f(1^\ell 0^{n-\ell}) = 0\}$ is a NN representation of size $n+1$.

For part *b)*, consider a NN representation of the parity function and let p be a positive prototype. If y is a neighbor of p then y is negative, but there is a positive prototype at distance 1 from y. Hence y must itself be a negative prototype. Repeating this argument it follows that every point is a prototype. □

A Boolean function f is a threshold function if there are weights $w_1, \ldots, w_n \in \mathbf{R}^n$ and a threshold $t \in \mathbf{R}^n$ such that for every $x \in \{0,1\}^n$ it holds that $f(x) = 1$ iff $w_1 x_1 + \ldots + w_n x_n \geq t$. The special case when $w_1 = \ldots = w_n = 1$ is denoted by TH_n^t. In particular, when $t = \frac{n}{2}$, we get the n-variable majority function $MAJ_n(x)$.

Theorem 2. a) *For every threshold function f it holds that $NN(f) = 2$.*

b) *If n is odd then $BNN(MAJ_n) = 2$ and if n is even then $BNN(MAJ_n) \leq \frac{n}{2} + 2$.*

c) $BNN\left(TH_n^{n/3}\right) = 2^{\Omega(n)}$.

Proof Part *a)* follows by taking a single positive, resp. negative, prototype, on a line perpendicular to the hyperplane defining the threshold function, at equal distances from the hyperplane.

Part *b)* is obtained for odd n by taking the all 0, resp. all 1, vectors as prototypes. In the even case let the all 0 vector be the single negative prototype, and let select arbitrary $\frac{n}{2} + 1$ vectors of weight $n-1$ as positive prototypes. Then every vector x of weight $\frac{n}{2}$ shares a 0 component with some positive prototype. Their distance is $\frac{n}{2} - 1$, and so this prototype is closer to x than the all 0 vector. It is easy to check that if x has weight different from $\frac{n}{2}$, then the prototype closest to it has the right label.

For part *c)*, let $t = \lceil \frac{n}{3} \rceil$ and consider the sets of Boolean prototypes $P, N \subseteq \{0,1\}^n$ for TH_n^t. Let x be a vector of weight t, and p be a positive prototype closest to x. We claim that $x \leq p$.

Otherwise assume that $x_i = 1$, $p_i = 0$ and consider $y = x^{(i)}$, with closest negative prototype q. Then $d_H(x,p) = d_H(y,p)+1 > d_H(y,q)+1$. On the other hand $d_H(x,p) < d_H(x,q) \leq d_H(y,q)+1$, a contradiction.

It follows similarly that if y is a vector of weight $t-1$ and q is a negative prototype closest to y then $q \leq y$. This implies that for every vector x of weight t there is a negative prototype q such that $q \leq x$ (a prototype closest to a lower neighbor of x will have this property). Thus for every vector x of weight t it holds that $d_H(x,q) \leq t$ for some negative prototype q. This means that if p is a positive prototype closest to x then $d_H(x,p) < t$ and so $|p| < 2t$.

Consider now the set of vectors of weight t. Each is covered by a positive prototype of weight less than $2t$. Each such positive prototype can cover at most $\binom{2t}{t}$ vectors of weight t. Hence we need at least

$$\frac{\binom{n}{t}}{\binom{2t}{t}} = 2^{\Omega(n)}$$

positive prototypes. □

The argument of part *c)* generalizes to every function TH_n^t, where $|t - \frac{n}{2}| \geq \delta n$ for any fixed $\delta > 0$.

4. General bounds

The first bound shows that the upper bound of (1) for nearest neighbor complexity can be improved asymptotically by a factor of $\frac{1}{n}$.

Theorem 3. *For every n-variable Boolean function*

$$NN(f) \leq (1+o(1))\frac{2^{n+2}}{n}.$$

Proof A set $B_a \subseteq \{0,1\}^n$ is a *ball of radius one* if it consists of a vector $a \in \{0,1\}^n$ (the center of the ball) and all its neighbors. A set $S_a \subseteq \{0,1\}^n$ is a *sphere of radius one* if it consists of all the neighbors of a vector $a \in \{0,1\}^n$.

Lemma 4. *Let A be a subset of a sphere S of radius one with $|A| = \ell \geq 3$, and let $c_A = \frac{1}{|A|}\sum_{x \in A} x$ be the centroid of A. Then*

a) *$d(c_A, x) < 1$ for every $x \in A$,*

b) *$d(c_A, x) \geq 1$ for every x such that $x \notin A$ and x is different from the center of S.*

Proof Assume *w.l.o.g.* that S consists of the unit vectors, and A consists of the first ℓ unit vectors. Then $c_A = (\frac{1}{\ell}, \ldots, \frac{1}{\ell}, 0, \ldots, 0)$, where the first ℓ coordinates are nonzero. If $x \in A$ then

$$d(c_A, x) = \left(\frac{\ell-1}{\ell}\right)^2 + (\ell-1)\left(\frac{1}{\ell}\right)^2 = \frac{\ell-1}{\ell} < 1.$$

If $x \notin A$ and x is different from the center of S then if x has a 1 component in the last $n-\ell$ coordinates then $d(c_A, x) \geq 1$. Otherwise x has at least two 1's in the first ℓ coordinates and so as $\ell \geq 3$ it holds that

$$d(c_A, x) \geq 2\left(\frac{\ell-1}{\ell}\right)^2 + (\ell-2)\left(\frac{1}{\ell}\right)^2 = 2 - \frac{3}{\ell} \geq 1. \Box$$

Partition $\{0,1\}^n$ into subsets $A_1, \ldots, A_s$ such that each A_i is a subset of some ball B_i of radius one with center a_i, and let A_i^1 (resp. A_i^0) be the set of points $x \neq a_i$ in A_i with $f(x) = 1$ (resp., $f(x) = 0$). In each A_i pick the following prototypes:

- if $|A_i^1| \geq 3$ then let $c_{A_i^1}$ be a positive prototype, otherwise let A_i^1 be a set of positive prototypes,
- if $|A_i^0| \geq 3$ then let $c_{A_i^0}$ be a negative prototype, otherwise let A_i^0 be a set of negative prototypes,
- if the center $a_i \in A_i$ then let a_i be a prototype with label $f(a_i)$.

The correctness of this set of prototypes follows from Lemma 4. The theorem then follows from the result that $\{0,1\}^n$ can be covered with $(1+o(1))\frac{2^n}{n}$ balls of radius one (Kabatyansky and Panchenko [5], see also Cohen et al. [2], generalizing Hamming codes). $\Box$

As the next result shows, almost all n-variable functions have exponential complexity.

Theorem 5. *For almost all n-variable Boolean functions*

$$NN(f) > \frac{2^{n/2}}{n}.$$

Proof Consider a set of prototypes $p_1, \ldots, p_m$ for some function f. By slightly perturbing the points if necessary, it may be assumed *w.l.o.g.* that $d(x, p_i) \neq d(x, p_j)$ for every $x \in \{0,1\}^n$ and $1 \leq i < j \leq m$. The distances $d(x, p_i)$ and $d(x, p_j)$ can be compared by considering the hyperplane H_{p_i,p_j} going through the midpoint of the segment $p_i p_j$, perpendicular to the segment, and determining on which side of the hyperplane x lies. If for another set of prototypes $q_1, \ldots, q_m$ (again, without ties), the hyperplanes H_{q_i,q_j} determine the same dichotomy of $\{0,1\}^n$ as H_{p_i,p_j} for every $1 \leq i < j \leq m$, then $q_1, \ldots, q_m$ are prototypes for the same function f.

Hyperplanes can realize at most 2^{n^2} dichotomies of x (see, e.g., Kailath *et al.* [10]) and thus m prototypes can realize at most

$$2^{n^2\binom{m}{2}} \tag{2}$$

n-variable Boolean functions. If a function can be realized with less than m prototypes then it can also be realized with m prototypes. A direct calculation shows that for $m = \frac{2^{n/2}}{n}$ the quantity (2) is $o\left(2^{2^n}\right)$. $\Box$

One actually gets the same bound for k-nearest neighbors as well. The only difference in the proof is that a set of m prototypes can represent m different functions for different values of k. Thus the upper bound (2) has to be multiplied by m, but the same bound remains valid.

Theorem 6. *For almost all n-variable Boolean functions f it holds that for every k*

$$k - NN(f) > \frac{2^{n/2}}{n}.\square$$

5. Bounds for an explicit function

In this section we give a lower bound for the nearest neighbor and the k-nearest neighbor complexities of a specific function. The *mod* 2 inner product function of $2n$ variables is defined by

$$IP_n(x_1, \ldots, x_n, y_1, \ldots, y_n) = (x_1 \wedge y_1) \oplus \ldots \oplus (x_n \wedge y_n).$$

The first part of the theorem applies to the nearest neighbor complexity, and the second part applies to the k-nearest neighbor complexity for all possible values of k.

Theorem 7. a) $NN(IP_n) \geq \frac{n}{2} + 1$,

b) $\min_k k - NN(IP_n) \geq (1 - o(1))\frac{n}{\log n}$.

Proof For part *a)*, we first formulate a general connection between nearest neighbor complexity and the complexity of computing a function by linear decision trees.

A linear decision tree over the variables $x_1, \ldots, x_n$ is a binary tree, where each inner node is labeled by a linear test of the form $w_1x_1 + \ldots + w_nx_n : t$, for some $w_1, \ldots, w_n, t \in \mathbf{R}$, the edges leaving the node are labelled $\leq$ and $>$, and the leaves are labeled 0 and 1. For an input vector $x \in \{0,1\}^n$, the function value computed by the tree is the label of the leaf reached by following the path corresponding to the results of the tests for x. The linear decision tree complexity, $LDT(f)$, of a function f is the minimum of the depths of linear decision trees computing f.

Lemma 8. *For every Boolean function f it holds that $LDT(f) \leq NN(f) - 1$.*

Proof Consider a set of prototypes $p_1, \ldots, p_m$ for f. Given $x \in \{0,1\}^n$, the standard algorithm for finding the minimum of the numbers $d(x, p_i)$ cycles through the p_i's and keeps track of the current minimum. A comparison, as in the proof of Theorem 5, corresponds to the evaluation of a linear test. Thus we obtain a linear decision tree for f of depth $m - 1$. $\square$

In view of the lemma, the lower bound of part *a)* is implied by the following lower bound of Gröger and Turán [4].

Lemma 9. $LDT(IP_n) \geq \frac{n}{2}$. $\square$

For part *b)*, we need a variation on Lemma 8 which relates linear decision tree complexity to k-nearest neighbor complexity. Compared to Lemma 8, the difference in the proof of the following lemma is that instead of a minimum finding algorithm one has to use a sorting algorithm to sort the distances $d(x, p_i)$. Once the distances $d(x, p_i)$ are sorted, we can determine the classification provided by the k-nearest neighbor representation, and thus we obtain a linear decision tree for the function.

Lemma 10. *For every Boolean function f and every k it holds that*

$$LDT(f) \leq (1 + o(1)) \cdot k - NN(f) \cdot \log(k - NN(f)).\square$$

Part *b)* then follows directly from Lemmas 9 and 10. □

6. Remarks and open problems

It would be interesting to prove an exponential lower bound for the nearest neighbor complexity of an explicitly defined function. It follows by an argument similar to the one in Lemma 8 that if a function can be represented with m prototypes then it can be computed by a threshold circuit of depth 3 and size $O(m^2)$, where the gates on the bottom level are threshold gates, gates on the middle level are $\wedge$ gates and the final gate is an $\vee$ gate. These circuits have a simple geometric interpretation: they correspond to a separation of the true and false points by a union of polyhedra. A related class of circuits, where the final gate is a parity gate instead of an $\vee$ gate, is discussed in Regan [9]. There are no exponential lower bounds known for the depth 3 threshold circuit complexity of an explicitly defined function (see, e.g., Siu et al. [10] for a survey of threshold circuit complexity), not even in the special case mentioned above, as far as we know. Thus a lower bound for the nearest neighbor complexity could be of interest for threshold circuits as well.

Another question is whether the upper bound $n + 1$ in Proposition 1 is optimal for the parity function (it is, for $n = 2, 3$). The gap between the upper bound of Theorem 3 and the lower bound of Theorem 5 should be narrowed. The relationship between nearest neighbor complexity and k-nearest neighbor complexity is open. Finally, other versions of nearest neighbor complexity could also be studied, for example, weighted versions (see, e.g. [3]) and other metrics.

Acknowledgement We would like to thank Simon Kasif for suggesting the problem discussed in this paper.

References

[1] E. B. Baum: When are k-nearest neighbor and backpropagation accurate for feasible-sized sets of examples?, in: *Computational Learning Theory and Natural Learning Systems, Vol. I:*

Constraints and Prospects, S. J. Hanson, G. A. Drastal, R. L. Rivest eds., 415-442. MIT Press, 1994.

[2] G. Cohen, I. Honkala, S. Litsyn, A. Lobstein: *Covering Codes.* North - Holland Math. Library, Vol. 54, Elsevier, 1997.

[3] R. O. Duda, P. E. Hart, D. G. Stork: *Pattern Classification*, 2nd ed. Wiley, 2001.

[4] H. - D. Gröger, Gy. Turán: On linear decision trees computing Boolean functions, *18. ICALP* (1991), 707-718. Springer LNCS 510.

[5] G. A. Kabatyansky, V. I. Panchenko: Packing and covering of Hamming spaces with balls of unit radius, *Probl. Inf. Trans.* **24** (1988), 3-16.

[6] S. Kasif, personal communication, 2000.

[7] T. M. Mitchell: *Machine Learning.* McGraw-Hill, 1997.

[8] K. Mulmuley: *Computational Geometry: An Introduction Through Randomized Algorithms.* Prentice Hall, 1993.

[9] K. Regan: Polynomials and Combinatorial Definitions of Languages, in: *Complexity Theory Retrospective II*, L. Hemaspaandra and A. Selman, eds., 261-293. Springer, 1997.

[10] K. - Y. Siu, V. Roychowdhury, T. Kailath: *Discrete Neural Computation: A Theoretical Foundation.* Prentice Hall, 1995.

[11] G. Wilfong: Nearest neighbor problems, *Int. J. of Comp. Geom. and Appl.* **2** (1992), 383-416.