Predicting the Generalization Ability of Neural Networks Resembling the Nearest-Neighbor Algorithm

Marco Muselli

Istituto per i Circuiti Elettronici
Consiglio Nazionale delle Ricerche
via De Marini, 6 - 16149 Genova, Italy
Email: muselli@ice.ge.cnr.it

Abstract

The definition of *nearest-neighbor probability* $p(\mathcal{C})$ is introduced to characterize classification problems with binary inputs. It measures the likelihood that two patterns, which are close according to the Hamming distance, are assigned to the same class.

It is shown that the generalization ability $g_{NN}(\mathcal{C})$ of neural networks that resemble the nearest-neighbor algorithm can be expressed as a function of $p(\mathcal{C})$ and is upper bounded by $p(\mathcal{C})$ when $p(\mathcal{C}) > 0.5$. In the opposite case a proper operator, called *complementation*, is proposed to improve the classification process in the test phase.

1 Introduction

Despite its evident simplicity, the generalization ability of the nearest-neighbor algorithm in many real-world problems is comparable with the one achieved by sofisticated learning techniques for neural networks. Moreover, its theoretical validity has been well-established [1] allowing to guarantee the general reliability of this classification method.

Unfortunately, the assignment of a class to a new input pattern requires the execution of a search among the samples in the training set to detect the closest one. The execution time needed increases with the size of the training set, which can lead to an excessive computational burden.

A way to overcome such a difficulty is to emulate the nearest-neighbor algorithm through a neural network, whose intrinsic parallelism speeds up the classification process. In this case a training phase is needed to obtain the weights, and sometimes the architecture, of the resulting connectionist model.

Several methods have been proposed to achieve this goal [2, 3, 4, 5, 6]. Most of them introduce a partition of the input space in close subsets, each of which is generally associated with a single neuron. In many models, such as the Learning Vector Quantization (LVQ) [7], the test phase is performed according to the distance from input patterns called *prototypes* not necessarily included in the training set.

An intrinsic limitation of the nearest-neighbor approach is that the classification process is only based on local properties. Thus, it may miss some global behaviors of the classifier to be identified. It would be interesting to evaluate the locality of a training set, i.e. an estimate of the generalization ability achievable by the nearest-neighbor algorithm, through a direct inspection of the given samples. Nevertheless, only asymptotic estimates are available in the literature [1], which are not useful when the size of the training set is small.

The present paper aims to develop a theoretical framework that can be applied to classification problems with binary inputs. The concept of structured random classifier is introduced together with the definition of *nearest-neighbor probability* $p(\mathcal{C})$, which allows to measure the locality of a binary classifier $\mathcal{C}$. Bounds on the generalization ability $g_{NN}(\mathcal{C})$ achievable by the nearest-neighbor algorithm and by neural networks that adopt the same approach will be obtained through the quantity $p(\mathcal{C})$. A simple preprocessing method, called *complementation*, is then proposed to allow the successful treatment of classifiers with poor locality.

For the sake of brevity, the proofs of the theorems will be omitted, but can be found in [8].

2 The Nearest-Neighbor Probability

Consider a single output binary classifier $\mathcal{C}$ which receives Boolean patterns $\boldsymbol{x}$ having n components as input. If the integer values -1 and $+1$ are used to code the two binary states, a Boolean function $f_{\mathcal{C}} : \{-1,+1\}^n \rightarrow \{-1,+1\}$ can be determined, where the value of $f_{\mathcal{C}}(\boldsymbol{x})$ gives the class assigned by $\mathcal{C}$ to any input pattern $\boldsymbol{x}$.

Denote with $d(\boldsymbol{x},\boldsymbol{z})$ the usual Hamming distance between two patterns $\boldsymbol{x},\boldsymbol{z} \in \{-1,+1\}^n$, given by the number of different bits

$$d(\boldsymbol{x},\boldsymbol{z}) = \frac{1}{2}\sum_{i=1}^{n} |x_i - z_i|$$

being x_i the ith component of the binary vector $\boldsymbol{x}$. We will analyze the properties of a general class of binary classifiers, whose behavior is univocally determined by a fixed real number $p \in (0,1)$, called *generation probability*. The Boolean function $f_{\mathcal{C}}$ associated with a classifier $\mathcal{C}$ of this kind is generated through the algorithm in Fig. 1.

STRUCTURED RANDOM CLASSIFIER
(Generation Algorithm)

1. Set $h = 1$. Choose at random with uniform probability a pattern $\boldsymbol{x}_0 \in \{-1,+1\}^n$ and set

$$f_{\mathcal{C}}(\boldsymbol{x}_0) = \begin{cases} +1 & \text{with probability } 0.5 \\ -1 & \text{with probability } 0.5 \end{cases}$$

2. Consider the set D_h of the patterns $\boldsymbol{x} \in \{-1,+1\}^n$ having $d(\boldsymbol{x},\boldsymbol{x}_0) = h$.

3. For each $\boldsymbol{x} \in D_h$ choose at random with uniform probability a pattern $\boldsymbol{z} \in D_{h-1}$ such that $d(\boldsymbol{x},\boldsymbol{z}) = 1$ and set

$$f_{\mathcal{C}}(\boldsymbol{x}) = \begin{cases} f_{\mathcal{C}}(\boldsymbol{z}) & \text{with probability } p \\ -f_{\mathcal{C}}(\boldsymbol{z}) & \text{with probability } 1-p \end{cases}$$

4. Add 1 to h; if $h \leq n$ go to Step 2.

Figure 1: Algorithm used to generate a structured random classifier.

The following definition will be extensively used:

Definition 1 *A classifier $\mathcal{C}$ constructed according to the algorithm in Fig. 1 will be called* structured random classifier*.*

The following theorem gives a basic property of structured random classifiers, which allows to forecast the performance of the nearest-neighbor algorithm.

Theorem 1 *Let $\mathcal{C}$ be a structured random classifier, defined on the input domain $\{-1,+1\}^n$ and characterized by the generation probability p. If $m \leq n$ is a positive integer, for any pair $\boldsymbol{x},\boldsymbol{z}$ the probability*

$$p_m(\mathcal{C}) = \Pr\{f_{\mathcal{C}}(\boldsymbol{x}) = f_{\mathcal{C}}(\boldsymbol{z}) \mid d(\boldsymbol{x},\boldsymbol{z}) = m\} \tag{1}$$

only depends on p, m and n. In particular, we have

$$p_m(\mathcal{C}) = \frac{1 + (2p_1(\mathcal{C}) - 1)^m}{2} \tag{2}$$

Thus, the quantity $p(\mathcal{C}) = p_1(\mathcal{C}) = Pr\{f_{\mathcal{C}}(\boldsymbol{x}) = f_{\mathcal{C}}(\boldsymbol{z}) \mid d(\boldsymbol{x},\boldsymbol{z}) = 1\}$ is a characteristic of any structured random classifier $\mathcal{C}$; it will be called *nearest-neighbor probability*. In the same way, the real number $p_m(\mathcal{C})$ will be named the *mth-order nearest-neighbor probability* of $\mathcal{C}$.

2.1 Influence of $p(\mathcal{C})$ on binary classification

Consider a structured random classifier $\mathcal{C}$, defined on the input domain $\{-1,+1\}^n$ and associated with a Boolean function $f_{\mathcal{C}}$. In the noise-free assumption the behavior of $\mathcal{C}$ is to be determined starting from a given training set S that contains a collection of s samples $(\boldsymbol{x}_j, y_j)$, $j = 1, \ldots, s$, where $y_j = f_{\mathcal{C}}(\boldsymbol{x}_j)$. The vectors $\boldsymbol{x}_j$ are supposed to be obtained by s independent applications of a fixed (unknown) distribution probability Q acting on the domain $\{-1,+1\}^n$.

If $\mathcal{C}$ is a structured random classifier, the generalization ability $g_{NN}(\mathcal{C})$ of the nearest-neighbor algorithm, i.e. the probability of correctly classifying a new input pattern $\boldsymbol{x}$, can be expressed in terms of the nearest-neighbor probability $p(\mathcal{C})$, as shown by the following theorem. Here we have denoted with $\boldsymbol{X}$ and Y the random variables whose realization is $\boldsymbol{x}$ and y, respectively; furthermore, $d(\boldsymbol{x}, S)$ is the distance between $\boldsymbol{x}$ and the training set S, defined as

$$d(\boldsymbol{x}, S) = \min_{\boldsymbol{x}_j \in S} d(\boldsymbol{x}, \boldsymbol{x}_j)$$

Theorem 2 *Let $\mathcal{C}$ be a structured random classifier, defined on the input domain $\{-1,+1\}^n$. The probability $g_{NN}(\mathcal{C})$ that the nearest-neighbor algorithm correctly classifies a new input pattern $\boldsymbol{X}$ not belonging to a given training set S is given by*

$$g_{NN}(\mathcal{C}) = \Pr\{Y = f_{\mathcal{C}}(\boldsymbol{X})\} = \sum_{m \geq 1} \Pr\{d(\boldsymbol{X}, S) = m\} p_m(\mathcal{C}) \tag{3}$$

where the probability $\Pr\{d(\boldsymbol{X}, S) = m\}$ is evaluated according to the distribution Q used to generate $\boldsymbol{x}$.

As one can see, the quantity $g_{NN}(\mathcal{C})$ increases with the nearest-neighbor probabilities $p_m(\mathcal{C})$; in turn, these depend on $p(\mathcal{C})$ according to equation (2). Note that for $p(\mathcal{C}) = 0.5$ equation (3) gives $g_{NN}(\mathcal{C}) = 0.5$, which corresponds to a random classification with uniform probability. On the other hand, when $0.5 < p(\mathcal{C}) \leq 1$ simple bounds for the quantity $g_{NN}(\mathcal{C})$ are given by the following corollary.

Corollary 1 *For any structured random classifier $\mathcal{C}$ having $0.5 < p(\mathcal{C}) \leq 1$ the generalization ability $g_{NN}(\mathcal{C})$ satisfies the following inequalities*

$$0.5 < g_{NN}(\mathcal{C}) \leq p(\mathcal{C}) \tag{4}$$

If $p(\mathcal{C}) < 0.5$ we are not able to derive general bounds for the quantity $g_{NN}(\mathcal{C})$. However, if the distribution probability Q, used to generate the training S, is uniform on its domain $\{-1,+1\}^n$, the following theorem gives an estimate of the generalization ability of the nearest-neighbor algorithm.

Theorem 3 *For any structured random classifier $\mathcal{C}$ the generalization ability $g_{NN}(\mathcal{C})$ satisfies the following inequality*

$$g_{NN}(\mathcal{C}) \leq p(\mathcal{C}) - (1 - p(\mathcal{C}))(2p(\mathcal{C}) - 1) \frac{\left(\begin{array}{c} 2^n - n - 1 \\ s \end{array} \right)}{\left(\begin{array}{c} 2^n - 1 \\ s \end{array} \right)} \tag{5}$$

if the training set S has been obtained by a random sampling with uniform probability in the input space.

2.2 Complement of a Boolean function

If $\mathcal{C}$ is a structured random classifier, the generalization ability $g_{NN}(\mathcal{C})$ can be made not lower than 0.5 by preprocessing the training set through a proper transformation [9], which will be called *complementation*.

Definition 2 *Given a Boolean function $f_{\mathcal{C}} : \{-1,+1\}^n \to \{-1,+1\}$, its* complement *$\tilde{f}_{\mathcal{C}}$ is defined as*

$$\tilde{f}_{\mathcal{C}}(\boldsymbol{x}) = f_{\mathcal{C}}(\boldsymbol{x}) \oplus f_{\mathcal{P}}(\boldsymbol{x}) = f_{\mathcal{C}}(\boldsymbol{x}) f_{\mathcal{P}}(\boldsymbol{x})$$

where $f_{\mathcal{P}} = x_1 \oplus x_2 \oplus \cdots \oplus x_n$ is the parity function and $\oplus$ is the logical XOR *product.*

In analogy with this definition the notation $\tilde{\mathcal{C}}$ will be used henceforth to denote the binary classifier associated with $\tilde{f}_{\mathcal{C}}$.

The complement operation allows one to change the nearest-neighbor probability according to the following theorem

Theorem 4 *The nearest-neighbor probability $p(\tilde{\mathcal{C}})$ of the complement of a structured random classifier $\mathcal{C}$ is*

$$p(\tilde{\mathcal{C}}) = 1 - p(\mathcal{C})$$

Thus, when facing a structured random classifier $\mathcal{C}$ with $p(\mathcal{C}) < 0.5$, the application of Theorem 4 allows to make the generalization ability $g_{NN}(\mathcal{C})$ higher than 0.5 by complementing the associated Boolean function $f_{\mathcal{C}}$.

3 Estimating the nearest-neighbor probability

The results obtained in the previous sections can be applied to any practical situation if we assume that the classification problem at hand is a particular realization of a structured random classifier $\mathcal{C}$. In these cases it is necessary to achieve a refined estimate of the nearest-neighbor probability $p(\mathcal{C})$ which should be obtained by using all the available information.

Consider a Bernoulli random variable B assuming the value 1 with probability p (and thus the value 0 with probability $1-p$). The Chebyshev's inequality [1] bounds the probability that the empirical average of l subsequent realizations of B shifts away from its expected value $EB = p$

$$\Pr\{|\mu(p) - p| \geq \varepsilon\} \leq \frac{\mathrm{Var}(B)}{l\varepsilon^2} \tag{6}$$

where $\mathrm{Var}(B)$ is the variance of B, whereas $\mu(p)$ is the empirical average of B obtained by a sample with l realizations b_i, $i = 1, \ldots, l$

$$\mu(p) = \frac{1}{l}\sum_{i=1}^{l} b_i$$

Since B is a Bernoulli random variable, $\mathrm{Var}(B) = p(1-p)$.

In our case p is one of the high-order nearest-neighbor probabilities $p_m(\mathcal{C})$, whereas B is the output of the two-state function h defined as

$$h(\boldsymbol{x}, \boldsymbol{z}) = 1 - \frac{|f_{\mathcal{C}}(\boldsymbol{x}) - f_{\mathcal{C}}(\boldsymbol{z})|}{2} \tag{7}$$

By virtue of (2) the empirical average $\mu(p_m(\mathcal{C}))$ can be employed to obtain an estimate $\hat{p}^{(m)}(\mathcal{C})$ of the nearest-neighbor probability $p(\mathcal{C})$

$$\hat{p}^{(m)}(\mathcal{C}) = \frac{1 + (2\mu(p_m(\mathcal{C})) - 1)^{1/m}}{2}$$

This estimate can be used if the probability $\Pr\{|\hat{p}^{(m)}(\mathcal{C}) - p(\mathcal{C})| \geq \varepsilon\}$ is sufficiently small, which can be verified by employing the following result.

Theorem 5 *If $\sigma_m^2(\mathcal{C})$ is the variance of the output of the two-state function $h(\boldsymbol{x}, \boldsymbol{z})$ defined in (7), then*

$$\Pr\left\{|\hat{p}^{(m)}(\mathcal{C}) - p(\mathcal{C})| \geq \frac{\varepsilon}{|2\hat{p}^{(m)}(\mathcal{C}) - 1|^m}\right\} \leq \frac{\sigma_m^2(\mathcal{C})}{|C_m|\varepsilon^2} \tag{8}$$

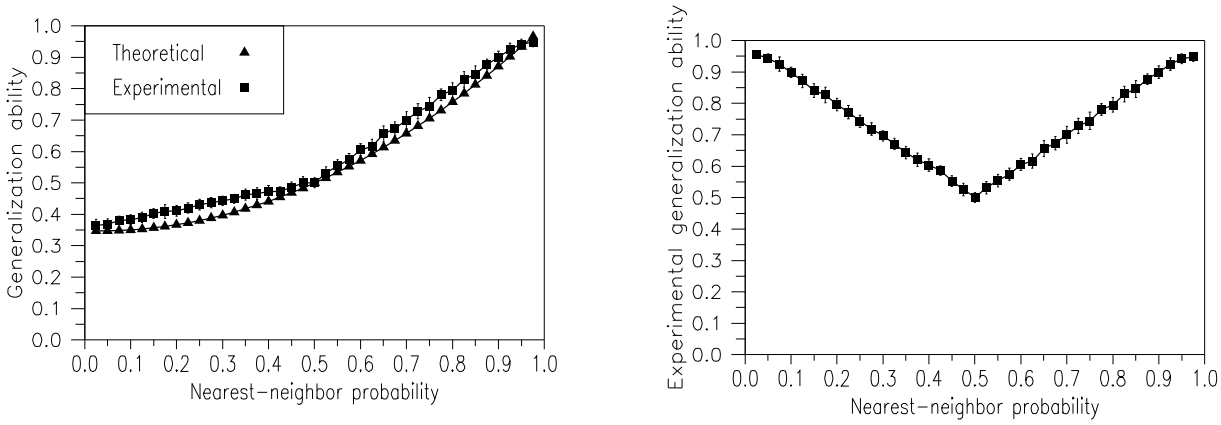
where $C_m = \{(\boldsymbol{x}, \boldsymbol{z}) \in S \times S \ : \ d(\boldsymbol{x}, \boldsymbol{z}) = m\}$.

The values of m that can be used in estimating the nearest-neighbor probability $p(\mathcal{C})$ through $\hat{p}^{(m)}(\mathcal{C})$ are therefore those which lead to a sufficiently small value of the so computed modified confidence. In particular, in our tests only the integers m are taken, which lead to a confidence not higher than that deriving from the Chebyshev's inequality for $m = 1$, provided that they do not exceed the maximum value of 0.1. If for $m = 1$ we obtain a confidence less than 1%, other values of m are not considered. In any case an estimate of the variance $\sigma_m^2(\mathcal{C})$ obtained with less than 20 samples is discarded.

4 Tests on artificial Boolean classifiers

In order to assess the significance of the nearest-neighbor probability $p(\mathcal{C})$ in the analysis of the generalization ability of the nearest-neighbor algorithm and of neural networks employing the same approach, two kind of experiments have been devised. In the first of them we investigate binary classifiers $\mathcal{C}$ whose nearest-neighbor probability $p(\mathcal{C})$ is fixed beforehand. Through a proper algorithm we have generated the truth tables of 30 Boolean functions for any value of the nearest-neighbor probability ranging from 0 to 1 with step 0.025. These values have been estimated by using the procedure described in the previous section; the dimension of the input space was always $n = 10$.

102 (10%) samples among the total 1024 have been randomly chosen (with uniform probability) to form the training set S, whereas the remaining 922 input patterns have then been used to evaluate the generalization ability $g_{NN}(\mathcal{C})$ of the nearest-neighbor algorithm. This is reported in Fig. 2a together with the upper bound (5), whose right hand side has been computed by employing the estimate of $p(\mathcal{C})$ obtained on the pattern in S.



a) b)

Figure 2: Theoretical and experimental generalization ability of the nearest-neighbor algorithm on artificial binary classifiers: a) before complementation, b) after complementation.

As one can see, the difference between the theoretical prevision and the experimental evaluation of the generalization ability performed by the nearest-neighbor method is very small, thus showing the high confidence associated with equation (5). The deviations reported in Fig. 2 are mainly due to the uncertainty in the empirical estimation of the nearest-neighbor probability $p(\mathcal{C})$.

It is noteworthy to observe that the generalization ability is less than 0.5 (random classifier) when $p(\mathcal{C}) < 0.5$. As previously described, this undesirable behavior can be overcome by preprocessing the training (and test) set through the complementation defined in Sec. 2.2. This leads to the curve plotted in Fig. 2b for the experimental evaluation of the generalization ability $g_{NN}(\mathcal{C})$. The symmetry of this empirical curve points out the efficiency of the complement operator.

4.1 Monk's problems

A second experiment deals with three realistic benchmarks, known as *Monk's problems* [10]; the associated classifiers are defined in a "robot" domain, described by six discrete attributes, which lead to 432 different samples. The first two Monk's problems are noise-free, whereas in the third case 5% of the total samples in the training set are misclassifed, simulating the action of noise in their aquisition. Since the inputs are discrete but not Boolean, a proper transformation has been applied, thus resulting in binary input patterns containing 15 components, each of which is associated with the presence of a particular attribute in the corresponding robot.

The estimation of the nearest-neighbor probability gives the values listed in the second column of Tab. 1. Only in the second Monk's problem this value is less than 0.5, thus leading to an advantage in using the

Problem	NN prob.	Upper bound	Class. ability
Monk #1	0.9085	0.9085	0.8052
Monk #2	0.3030	0.5571	0.4525
Monk #2 (compl.)	0.6970	0.6970	0.6502
Monk #3	0.9232	0.9232	0.8391

Table 1: Nearest-neighbor probability estimate, upper bound and real value of the nearest-neighbor algorithm generalization ability for the Monk's problems.

complement operator, whose corresponding results are shown in the third row.

The two rightmost columns of Tab. 1 contain the upper bound and the real value of the generalization ability obtained with the nearest-neighbor algorithm. Since the training sets are not produced by a uniform sampling of the input space, equation (5) is only used when the estimation of the nearest-neighbor probability $p(\mathcal{C})$ falls below 0.5. In the other cases, $p(\mathcal{C})$ is directly employed as an upper bound for the generalization ability.

The difference between the values reported in these columns is always around 0.1; this error is mostly due to the small number of samples at hand: 432 input patterns out of the total $2^{15} = 32768$ can only give partial information about the underlying nearest-neighbor probability.

References

[1] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition.* New York: Springer-Verlag, 1997.

[2] S. G. Smyth, "Designing multilayer perceptrons from nearest-neighbor systems," *IEEE Transactions on Neural Networks*, vol. 3, pp. 329–333, 1992.

[3] H. Yan, "Building a robust nearest neighbor classifier containing only a small number of prototypes," *International Journal of Neural Systems*, vol. 3, pp. 361–369, 1992.

[4] H.-C. Yau and M. T. Manry, "Iterative improvement of a nearest neighbor classifier," *Neural Networks*, vol. 4, pp. 517–524, 1991.

[5] L. P. Tay and D. J. Evans, "Fast learning artificial neural network (FLANN II) using the nearest neighbour recall," *Neural Parallel & Scientific Computations*, vol. 2, pp. 17–28, 1994.

[6] M. Muselli and D. Liberati, "Training digital circuits with Hamming Clustering," To appear on *IEEE Transactions on Circuit and Systems*, 2000.

[7] T. Kohonen, *Self-Organization and Associative Memory* (3rd ed.). Berlin: Springer-Verlag, 1989.

[8] M. Muselli, "Predicting the performance of the nearest-neighbor algorithm," Technical Report ICE 5/99, Institute for Electronic Circuits, Italian National Research Council, 1999.

[9] H. M. A. Andree, G. T. Barkema, W. Lourens, A. Taal, and J. C. Vermeulen, "A comparison study of binary feedforward neural networks and digital circuits," *Neural Networks*, vol. 6, pp. 785–790, 1993.

[10] S. Thrun, J. Bala, E. Bloedorn, I. Bratko, B. Cestnik, K. De Jong, S. Džeroski, S. Fahlman, D. Fisher, R. Hamann, K. Kaufman, S. Keller, I. Kononenko, J. Kreuziger, R. Michalski, T. Mitchell, P. Pachowicz, Y. Reich, H. Vafaie, W. Van de Welde, W. Wenzel, J. Wnek, and J. Zhang, "A performance comparison of different learning algorithms," Technical Report CMU-CS-91-197, Pittsburgh, PA: Department of Computer Science, Carnegie Mellow University, 1991.