BUDAPEST 2004

Proceedings

2004 IEEE International Joint Conference on Neural Networks

IEEE

IEEE Computational Intelligence Society

International Neural Network Society

European Neural Network Society

Japanese Neural Network Society

JNNS

Budapest, Hungary
25-29 July 2004

Plenary Poster Session: Learning and Generalization
Chair: Joos Vandewalle and Gusztav Hencsey, Room: P

On a Generalization Complexity Measure for Boolean Functions

Leonardo Franco
Department of Experimental Psychology
University of Oxford
South Parks Road
Oxford OX1 3UD, UK
E-mail: Leonardo.Franco@psy.ox.ac.uk

Martin Anthony
Department of Mathematics
London School of Economics and Political Science
London WC2A 2AE, UK
E-mail: m.anthony@lse.ac.uk

Abstract— **We analyze Boolean functions using a recently proposed measure of their complexity. This complexity measure, motivated by the aim of relating the complexity of the functions with the generalization ability that can be obtained when the functions are implemented in feed-forward neural networks, is the sum of two components. The first of these is related to the 'average sensitivity' of the function and the second is, in a sense, a measure of the 'randomness' or lack of structure of the function. In this paper, we investigate the importance of using the second term in the complexity measure. We also explore the existence of very complex Boolean functions, considering, in particular, the symmetric Boolean functions.**

I. Introduction

The complexity of Boolean functions is one of the central and classical topics in the theory of computation. Recently, Franco [7] and Franco & Cannas [11] introduced a complexity measure for Boolean functions that appears to be related to the generalization error when learning the functions by neural networks. This complexity measure has been derived from results showing that the generalization ability obtained for Boolean functions and for the number of examples (or similarly queries) needed to learn the functions when implemented in neural networks is related to the number of pairs of examples that are similar (close with respect to the Hamming distance), but have opposite outputs [9], [10]. When only the bordering (or boundary) examples (those at Hamming distance 1) are considered, the complexity measure becomes equivalent to average sensitivity, a measure introduced by Linial et al. [16]. Average sensitivity has been linked in [16] to the complexity of learning in the probabilistic 'PAC' model of learning [18]; and many results about the average sensitivity of Boolean functions have been obtained for different classes of Boolean functions [5], [4]. It has been shown [7] and is further analyzed in this paper that terms that account for the number of pairs of opposite examples at Hamming distance larger than 1 are important in obtaining a better match between the complexity of different kinds of Boolean functions and the observed generalization ability.

II. The complexity measure and its interpretation

In its most general form, the complexity measure considered here consists of a sum of terms, C_i, each of which accounts for the number of neighboring examples at a given Hamming distance having different outputs. The complexity measure can be written in a general form as:

$$C = C_1 + \sum_{i=2}^{N/2} \alpha_i C_i, \qquad (1)$$

where α_i are constant values that weight how pairs of oppositely-classified examples (that is, elements of $\{0,1\}^N$) at Hamming distance i contribute to the total complexity, and N is the number of input bits. Each term C_i has a normalization factor that takes into account both the number of neighboring examples at Hamming distance i and the total number 2^N of examples. Explicitly,

$$C_i[f] = \frac{1}{2^N \binom{N}{i}} \sum_{x \in \{0,1\}^N} \sum_{\{y : d(y,x)=i\}} |f(x) - f(y)|, \qquad (2)$$

where $d(y,x)$ is the Hamming distance between x and y (the number of coordinates in which they differ). Thus, $C_i[f]$ may be interpreted as the probability, uniformly over choice of example x, and uniformly over the choice of an example y at Hamming distance i from x, that $f(y) \neq f(x)$.

The first term, C_1 is proportional to the number of bordering (or boundary) examples, those with an immediate neighbor having opposite output. Equivalently, it is the probability that 'flipping' a uniformly chosen bit in a uniformly chosen example will change the output of f. This is proportional to the 'average sensitivity' $s(f)$ [16], [3], [15]: in fact, $C_1[f] = s(f)/N$. (The average sensitivity $s(f)$ is related to the notion of the 'influence' of a variable; see [15], for instance: $s(f)$ is the sum of the influences of the N variables.) The number of bordering examples has been shown to be related to the generalization ability that can be obtained when Boolean functions are implemented in neural networks [9], to the number of examples needed to obtain perfect generalization [9], [10], to a bound on the number of examples needed to specify a linearly separable function [1], and to the query complexity of monotone Boolean functions [17]. Moreover, links between the sensitivity of a Boolean function and its learnability in the PAC sense has been established [16] and many results regarding the average sensitivity of Boolean functions have been

0-7803-8359-1/04/$20.00 ©2004 IEEE

derived [5], [15], [12], [4]. Following [15], the complexity measures can be related to the Fourier coefficients of f; see the full version of this paper [8].

III. Importance of the second order term of the complexity measure

The second order term has been shown to be relevant in order to produce an accurate match between the complexity measure and the observed generalization ability when the functions are implemented in neural networks (Franco, 2002). Experimental results indicate that the first-order complexity term $C_1[f]$ alone does not give as good a correspondence as does the combination $C_1[f] + C_2[f]$.

Figure 1a shows the generalization ability vs. the first order complexity (C_1) obtained from simulations performed for three different classes of functions. The first class of functions (indicated as 'F. const + rand.mod' in the figure) was generated by modifications of the constant, identically-1, function, producing functions with first-order complexities C_1 between 0 and 0.5. These were generated as a function of a parameter p in the following way: for every example, a random uniform number in the range $[0, 1]$ was selected and then compared to the value of p. If the random value was smaller than p then the output of the function on that example was randomly selected with equal probability to be 0 or 1. Thus, for each example, with probability p, the output is randomly chosen, and with probability $1-p$ the output is 1. The second set of function ('F. parity + rand. mod.' in the figure) was generated in the same way but through random modifications of the parity function, to obtain functions with a complexity between 1 and 0.5. (The parity function is the function that has output 1 on an example precisely when the example has an odd number of entries equal to 1.) The third set of functions ('F. parity u + rand. mod' in the figure) was generated as follows: starting with the parity function, for each positive example (that is, an example with output 1 on the parity function), the output is changed to 0 with probability p. This yields functions with complexities ranging from 1 to 0, all of which, except the initial parity function, are unbalanced (in the sense that the number of outputs equal to 0 and 1 are different). Figure 1a shows, for each of these three classes, the generalization ability computed by simulations performed in a neural network architecture with $N = 8$ inputs and 8 neurons in the hidden layer trained with backpropagation, using half of the total number of examples for training, one fourth for validation and one fourth for testing the generalization ability. In figure 1b the generalization ability is plotted against the second order term of the complexity measure, C_2 and it can be observed that the general behaviour of the generalization seems uncorrelated to this second term. But when we plot, in figure 1c, the generalization ability versus $C_1 + C_2$ better agreement is obtained for the three different classes of Boolean functions. The discrepancy observed in the generalization ability in Fig.1 for functions with similar complexity C_1 appears to be almost totally corrected when $C_1 + C_2$ is used.

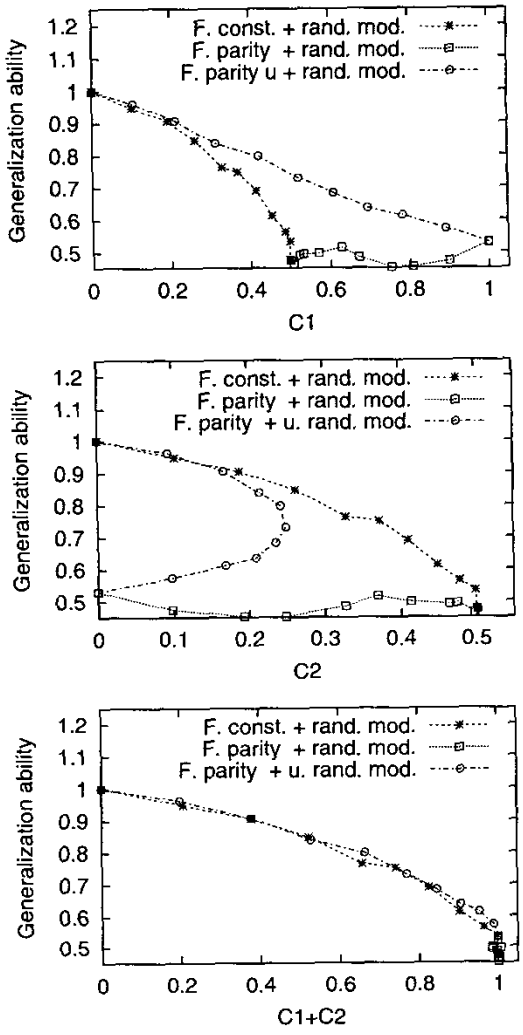

Fig. 1. Generalization ability vs. first term (top graph), second term (middle graph), and first plus second terms (bottom graph) of the complexity measure for three different classes of functions. (See text for details.)

Another indication of the importance of the second order complexity term C_2 comes from the fact that when only C_1 is considered, the functions with highest complexity turn out to be the very well known parity function and its complement [13], [10], yet numerical simulations have shown that there are functions which are more complex to implement on a neural network (in the sense that the generalization error is higher). Indeed, for this class of very complex functions the generalization error obtained is greater than 0.5 (which is what would be expected for random functions). In [7] it has been shown that the average generalization error over a whole set of functions using the same architecture is 0.5, indicating that there exist functions for which the generalization error is higher than 0.5. Similar results have been obtained for time

series implemented by perceptrons [19].

IV. A METHOD FOR FINDING VERY COMPLEX BOOLEAN FUNCTIONS AND GROUND STATES

In this section we investigate how to find Boolean functions with a high complexity. One technique that seems to be useful is to consider functions having a number of 'irrelevant attributes'. Such an approach is motivated by considerations from statistical mechanics. In [7], [11] an analogy is established between the Boolean function complexity measure and the Hamiltonian of magnetic systems. This analogy implies that there is a correspondence between the ground state of magnetic systems and the most complex functions. Ground states of many magnetic systems have been observed often to have a certain type of order (short or long range order) and it is a subject of controversy under which conditions this order does not arise [11], [6]. In some cases the ordered ground state consists of two equal size antiferromagnetic domains, corresponding in the language of Boolean functions to a parity function on $N-1$ variables, with the Nth variable being irrelevant. Finding the ground states of magnetic systems is a complicated task only rigorously undertaken in very few cases. It has been shown that in most cases the problem of rigorously establishing that a state is the ground state is computationally intractable [2], [14].

A Boolean function is said to have A irrelevant attributes if there are $i_1, i_2, \ldots, i_A$ such that the value $f(x_1, x_2, \ldots, x_N)$ of the function does not depend on $x_{i_1}, x_{i_2}, \ldots, x_{i_A}$. For the sake of simplicity, let us suppose these 'irrelevant attributes' are $x_{N-A+1}, x_{N-A+2}, \ldots, x_N$. Then, the value of f is determined entirely by its 'projection' f^* onto the relevant $N-A$ attributes, given, in this case, by

$$f^*(x_1, x_2, \ldots, x_{N-A}) = f(x_1, x_2, \ldots, x_{N-A}, 0, 0, \ldots, 0).$$

(The choice of 0 for the last A co-ordinates here is arbitrary, the point being that the value of f is independent of these.) The complexity of f can be related to that of f^* as follows:

$$\begin{aligned} C_{12}[f] &= \left(\frac{N-A}{N} + \alpha_2 \frac{2A(N-A)}{N(N-1)}\right) C_1[f^*] \\ &\quad + \alpha_2 \frac{(N-A)(N-A-1)}{N(N-1)} C_2[f^*]. \end{aligned} \tag{3}$$

It is easiest to see why (3) holds by using the probabilistic interpretations of C_1 and C_2; see [8] for details.

Consider the N-dimensional Boolean functions defined as the parity function on $N-A$ variables for $A = 0, 1, \ldots, N$. The complexity of these functions including the first and second order terms, C_1 and C_2, can be written in terms of A as:

$$\begin{aligned} C_{12}[f] &= C_1[f] + \alpha_2 C_2[f] \\ &= \frac{N-A}{N} + \alpha_2 \frac{2A(N-A)}{N(N-1)} \qquad (4) \\ &= \frac{(N-A)(N-1+2\alpha_2 A)}{N(N-1)} \qquad (5) \end{aligned}$$

This follows from (3), because the functions concerned have A irrelevant attributes, and because the projection f^* onto the $N-A$ relevant attributes is the parity function on $N-A$ variables, having $C_1[f^*] = 1$ and $C_2[f^*] = 0$. It can also be seen directly: the two terms in Eq. 4 represent the fraction of pairs of examples with opposite outputs at Hamming distances 1 and 2 respectively. Assume now that $\alpha_2 = 1$. To find the most complex function of this particular type, we set the derivative of C_{12} respect to A (assuming, that A is a continuous parameter), and we see that C_{12} is maximized when

$$A = A_{max} = \frac{N+1}{4}. \tag{6}$$

The complexity of the corresponding function is

$$C_{12}[f(A_{max})] = \frac{(3N-1)^2}{8N(N-1)} > \frac{9}{8} \tag{7}$$

The complexity of the function found is larger than 1.125 for any N, indicating that we have found a very complex function. (For comparison, we note that the complexity of the parity function and of a random function are approximately 1.0)

We empirically analyzed the generalization error obtained when Boolean functions are implemented on feed-forward neural networks, to see if this correlates with the complexity measure. For the functions that implement the parity function on $(N-A)$ variables, however, we did not find that the complexity of generalization (that is, the generalization error) correlated well with the complexity measure. It seems that this might be explained by the fact these functions are quite regular. (In fact as the number of relevant variables decreased, the functions seemed to be less complex to implement on neural networks.) Thus we decided to look instead some related functions with a greater element of randomness in their definition. We considered functions that implement the parity function of N variables, where the final A variables on any given input example were subject to random alteration: each of the final A variables of an example were, with probability 0.5, left unchanged, and with probability 0.5, were set to 0. On each example, then, the function constructed computed the parity function of $(N-A+\delta)$ variables on that example, where δ is a value between 0 and A, distributed according to a binomial distribution with mean $A/2$. The set of functions found is a complex one for which the generalization error can, for some values of A, be larger than 0.5.

In Fig. 2a we show the values of the complexity $C_1 + C_2$ of parity functions that depend on $N-A$ variables for the cases $N = 14, 10, 8, 4$. In Fig. 2b the generalization error obtained for the Boolean functions that implement the parity function on $(N-A+\delta)$ variables (in the sense described above) is shown for the same cases. The simulations were performed in one hidden layer architectures, trained with backpropagation, using half of the total number of examples for training, one fourth for validation and the remaining fourth to measure the generalization error. The error bars plotted show the standard error of the mean (SME), when 50 averages were taken. The SME decreases as N increases and for fixed N it was

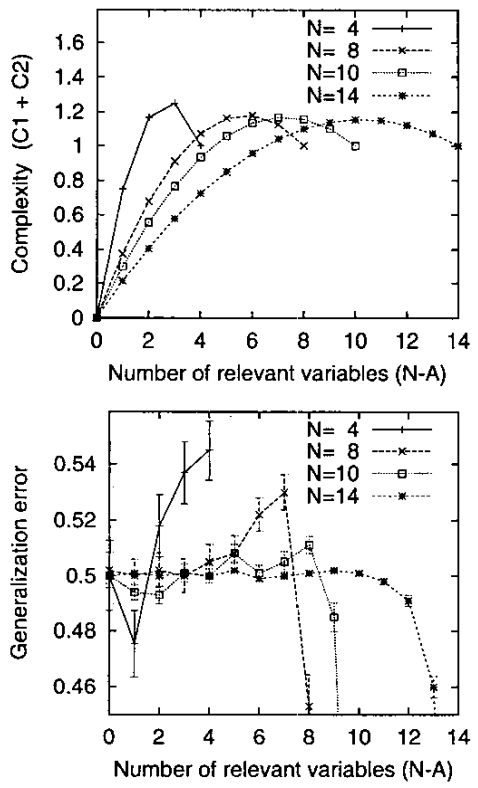

Fig. 2. a) The complexity $(C_1 + C_2)$ for Boolean functions that implement the parity function on $N - A$ variables as a function of the number of relevant variables $(N-A)$, for different values of N. b) Generalization error versus the number of relevant variables for the case of the Boolean functions described in the text as the parity on $(N - A + \delta)$ bits, for different values of N.

approximately constant as a function of A, except for the case $A = 0$, in which case the SME was normally larger. As the graphs indicate, we generally found that the value of A_{max} for which the generalization error is a maximum increases with N, to a maximum, and then falls again, as we indicated by the analysis for the parity function with irrelevant attributes.

We can also find functions for which C_2 is very high, independently of C_1. We find that functions defined as the parity on $\frac{N}{2}$ variables have a complexity C_2 equal to

$$C_2[f(A = \frac{N}{2})] = \frac{N}{2(N-1)} \tag{8}$$

which is larger than 0.5 for all N. For the case $N = 4$, this gives a function with $C_2 = \frac{2}{3}$ and we have exhaustively verified that this is the largest value that can be obtained for any Boolean function with $N = 4$. For cases where $N \neq 4$, we do not know how close the value obtained for C_2 is to its maximum possible; and, as we mentioned above, it could be that the demonstration that this value corresponds to the functions with maximum complexity is intractable [14]. For symmetric Boolean functions, however, as we observe in the next section, the maximum value that can be achieved for C_2 is 0.5.

V. Complexity of Symmetric Boolean Functions

An important class of Boolean functions is the class of symmetric functions, those for which the output depends only on the number of input bits ON (or, equivalently, on the weight of the example). This class includes many important functions, such as the parity function and the majority function. We first determine independently the maximum values of C_1 and C_2 that such functions can achieve and then by using an approximation, in which we consider only the input examples with a balanced or almost balanced number of input bits ON and OFF, we analyze which symmetric functions have high C_{12} complexity measure.

For the case of C_1 it is trivial to see that the parity function and its complement, for which $C_1 = 1$, are the only Boolean functions for which C_1 is maximum, and they are symmetric. The maximum possible value for C_2 is 0.5, as we now show.

For a given number of inputs bits, N, we may organize the examples in *levels* according to the number of bits ON (number of bits equal to 1), N_n. The number of OFF bits in a given example is then equal to $N_f = N - N_n$. The number of examples $N_{H=2}(x)$ at Hamming distance 2 from any given example x is $\binom{N}{2}$. For any example x, this number may be decomposed as

$$N_{H=2}(x) = N^{SL}_{H=2}(x) + N^{DL}_{H=2}(x) \tag{9}$$

where $N^{SL}_{H=2}(x)$ is the number of examples at distance 2 in the same level as x, and $N^{DL}_{H=2}(x)$ is the number of examples at distance 2 and in a different level (either level $i+2$ or $i-2$). Now, $N^{SL}_{H=2}(x)$ and $H^{DL}_{H=2}(x)$ depend only on the level to which x belongs. Explicitly, for all x in level i,

$$N^{SL}_{H=2}(x) = i(N-i), \tag{10}$$

as this is the number examples in the same level that have one different bit ON and one different bit OFF but the same total number of bits ON. For x in level i,

$$N^{DL}_{H=2}(x) = \binom{i}{2} + \binom{N-i}{2}, \tag{11}$$

as this is the number of different examples that can be obtained from x by flipping two ON bits to OFF or by flipping two OFF bits to ON. (The binomial coefficient $\binom{m}{k}$ is interpreted as 0 if $k > m$.)

For a symmetric Boolean function, examples in the same level have the same output. It follows that, in considering the complexity measure $C_2[f]$, only examples at distance 2 and in a different level need be considered. Therefore, if L_i denotes level i, we have

$$C_2[f] \leq \frac{1}{2^N}\frac{1}{\binom{N}{2}}\sum_{i=0}^{N}\sum_{x\in L_i} N^{DL}_{H=2}(x) \tag{12}$$

$$= \frac{1}{2^N}\frac{1}{\binom{N}{2}}\sum_{i=0}^{N}\binom{N}{i}\left(\binom{i}{2} + \binom{N-i}{2}\right) \tag{13}$$

$$= \frac{1}{2^N}\sum_{i=0}^{N} C(i,N), \tag{14}$$

where

$$C(i,N) = \frac{1}{\binom{N}{2}}\binom{N}{i}\left(\binom{i}{2} + \binom{N-i}{2}\right) \qquad (15)$$

is 2^N times the maximum possible contribution to $C_2[f]$ from the examples in level i.

We note that

$$\sum_{i=0}^{N}\binom{N}{i}\binom{i}{2} = 2^{N-2}\binom{N}{2},$$

since both sides of this identity are different expressions for the number of pairs $(\{x,y\},S)$ where $x,y \in [N] = \{1,2,\ldots,N\}$ and $x,y \in S \subseteq [N]$. Also,

$$\sum_{i=0}^{N}\binom{N}{i}\binom{N-i}{2} = 2^{N-2}\binom{N}{2}$$

since both sides are the number of pairs $(\{x,y\},S)$ where $x,y \in [N]$, $S \subseteq [N]$ and $x,y \notin S$. It follows that

$$C_2[f] \leq \frac{1}{2^N}\sum_{i=0}^{N} C(i,N) = \frac{1}{2}. \qquad (16)$$

The maximum value of 0.5 for C_2 can be achieved for any N by the symmetric functions with the property that the outputs chosen for the different levels alternate between 0 and 1 every two levels.

VI. Approximating the Complexity of Symmetric Boolean Functions

We now analyze approximately the C_{12}-complexity of symmetric functions, where $C_{12}[f] = C_1[f] + C_2[f]$. The idea of the approximation is to focus on the middle layers, these being the largest, to compute 'locally' the complexity around these layers. These local approximations will give approximations to the complexity of functions consistent with the same outputs on the middle five layers, provided the values assigned to higher and lower layers are such that the complexity observed around the middle layers 'extrapolates', at least on average, to the rest of the layers. Such an approach does not give exact results, but we believe the approximations obtained are useful indicators of the values of C_{12} for symmetric functions.

The middle layer (in the case of even N) and the middle two layers (in the case of odd N) are the most populated ones. Since the complexity measure that we are considering involves examples up to Hamming distance 2, to compute our approximation, we consider only the layers within distance 2 of these largest ones. We consider here the case N even and base our analysis, therefore, on the levels $\frac{N}{2}-2, \frac{N}{2}-1, \frac{N}{2}, \frac{N}{2}+1, \frac{N}{2}+2$. (Note that, although these layers are the largest, they only account for a fraction of order $1/\sqrt{N}$ of all 2^N examples on $\{0,1\}^N$, so for the approximations to be valid, it has to be assumed that the complexity observed locally around these central layers is continued throughout the rest of the layers. A possible alternative approach would be to work with a number of central layers of order $\sqrt{N}$, but this is clearly more difficult.)

Output values	Parameters (X_1,X_2,X_3,X_4)
...10101 ...	(2,2,0,0)
...10110 ...	(1,2,1,1)
...10100 ...	(2,1,0,1)
...00110 ...	(1,1,1,2)
...00100...	(2,0,0,2)
...01110 ...	(0,2,0,2)
...10111 ...	(1,1,1,0)
...00111 ...	(1,0,1,1)
...01111 ...	(0,1,0,1)
...11111 ...	(0,0,0,0)

TABLE I

We express the approximated complexity of the symmetric Boolean functions in terms of the value of the interactions of the examples at Hamming distance 1 and 2 of these 5 levels and introduce a function F that will account for this value. $F(X_1,X_2,X_3,X_4)$, where $X_1,X_2,X_4 \in \{0,1,2\}$ and $X_3 \in \{0,1\}$ reflect the values of the 'interactions' between the different levels considered. Explicitly, X_1 reflects the value of the interaction at Hamming distance 1 between levels $\frac{N}{2}-1$ and $\frac{N}{2}$ and between levels $\frac{N}{2}$ and $\frac{N}{2}+1$. Thus, X_1 takes value 0 if the Boolean function assigns the same output to all these four layers; it has value 1 if, for one of these pairs of layers, the outputs are different and for the other pair they are equal; and it has value 2 if for each pair of layers, the outputs are different. Similarly, X_2 reflects the value of the interaction at Hamming distance 1 between levels $\frac{N}{2}-2$ and $\frac{N}{2}-1$ and between levels $\frac{N}{2}+2$ and $\frac{N}{2}+1$. The parameters X_3 and X_4 describe distance-2 interactions: X_3 reflects the value of the interaction at Hamming distance 2 between levels $\frac{N}{2}-1$ and $\frac{N}{2}+1$, and X_4 accounts for the interaction at Hamming distance 2 between the middle level $\frac{N}{2}$ and levels $\frac{N}{2}\pm 2$. Without any loss of generality we assume outputs of examples in level $\frac{N}{2}$ to be 1. By symmetry, we then need only consider the ten configurations of output values to the five layers that are shown in the first column of Table VI. Here, the assignment ...10100... means, for example, that layer $N/2-2$ is assigned 1, layer $N/2-1$ is assigned 0, and so on. We also indicate, in the second column, the corresponding parameters (X_1,X_2,X_3,X_4).

We measure the C_{12} complexity 'locally' in these five central layers. These approximations are given by

$$F(X_1,X_2,X_3,X_4) = f(X_1,X_2) + g(X_3,X_4),$$

where f measures the C_1 complexity, 'relativized' to these layers, and g is an approximation for the C_2 complexity, locally around these layers. The function f is defined by

$$f(X_1,X_2) = \frac{X_1\binom{N}{\frac{N}{2}}\frac{N}{2} + X_2\binom{N}{\frac{N}{2}-1}\left(\frac{N}{2}-1\right)}{2\binom{N}{\frac{N}{2}}\frac{N}{2} + 2\binom{N}{\frac{N}{2}-1}\left(\frac{N}{2}-1\right)}. \qquad (17)$$

The approximation g is defined as follows.

$$g(X_3,X_4) = \frac{1}{2}\frac{X_3\binom{N}{\frac{N}{2}-1}\binom{\frac{N}{2}+1}{2} + X_4\binom{N}{\frac{N}{2}}\binom{\frac{N}{2}}{2}}{\binom{N}{\frac{N}{2}-1}\binom{\frac{N}{2}+1}{2} + 2\binom{N}{\frac{N}{2}}\binom{\frac{N}{2}}{2}}. \qquad (18)$$

X	f	g	F	mean $C_1/C_2/C_{12}$
(2,2,0,0)	1	0	**1**	0.8666/0.0969/**0.9635**
(1,2,1,1)	0.7143	0.3421	**1.0564**	0.6571/0.3064/**0.9635**
(2,1,0,1)	0.7857	0.1579	**0.9436**	0.7095/0.1936/**0.9031**
(1,1,1,2)	0.5	0.5	**1**	0.5/0.4031/**0.9031**
(2,0,0,2)	0.5714	0.3158	**0.8872**	0.5524/0.2903/**0.8427**
(0,2,0,2)	0.4286	0.3158	**0.7444**	0.4476/0.2903/**0.7379**
(1,1,1,0)	0.5	0.1842	**0.6842**	0.5/0.2097/**0.7097**
(1,0,1,1)	0.2857	0.3421	**0.6278**	0.3429/0.3064 /**0.6493**
(0,1,0,1)	0.2143	0.1579	**0.3722**	0.2905/0.1936/**0.4841**
(0,0,0,0)	0	0	**0**	0.1334/ 0.0969/**0.2303**

TABLE II

Here, the leading factor of $1/2$ reflects the fact that, for symmetric functions, C_2 can be no more than $1/2$ (as shown in the previous section). Table II shows the values of f, g and F for the configurations of interest (see Table VI), for the case $N = 14$. (The first column indicates the appropriate parameter values.) In the final column of the table, we give the mean values of C_1, C_2, C_{12} over all symmetric functions on $N = 14$ variables which extend the given configuration on the five central layers. So, for instance, for the last entry of the first column, we consider all those symmetric Boolean functions on $\{0,1\}^{14}$ which assign values $1,0,1,0,1$ to layers $5,6,7,8,9$ (respectively)—that is, all those that extend the pattern $\ldots 10101 \ldots$ of the central layers—and we compute the mean values of C_1, C_2 and C_{12} over all such functions.

VII. Discussion and conclusions

We analyzed in this paper a recently proposed measure for the complexity of Boolean functions related to the difficulty of generalization when neural networks are trained using examples of the function. We studied the first and second order terms of the complexity measure and demonstrated the importance of the second term in obtaining accurate comparisons with generalization error. Furthermore, we indicated that the second-order complexity term provides an estimate of the randomness existing in the output of a Boolean function.

By using an assumption on the nature of the most complex functions based on some results from statistical mechanics, we have been able to obtain very complex Boolean functions, with a complexity larger than 1. We showed empirically that the difficulty of generalization for these functions was related to the complexity measure (once the functions were modified by adding a controlled element of randomness).

For the class of symmetric Boolean functions, we first obtained a general bound on the maximum value that the second-order term of the complexity can take and, secondly, by focusing on the most populated levels of inputs, we found approximate values for the complexity of certain symmetric functions (and, in particular, we were able to obtain an indication that some complex symmetric functions existed). In most cases, these approximations compared well (for $N = 14$) with computationally calculated actual values of the complexities.

As a whole, the results presented in this paper show that the complexity measure introduced in (Franco, 2002) can be used to characterize different classes of Boolean functions in relationship to the complexity of generalization, and we think that this may lead to new lines of research contributing to a better understanding of the emergence of generalization in neural networks.

Acknowledgements

Fruitful discussions with Dr. S.A. Cannas are gratefully acknowledged. L.F. acknowledge support from MRC IRC Oxford UK. M.A. acknowledges support from the EU through the PASCAL Network of Excellence.

References

[1] M. Anthony, M., G. Brightwell and J. Shawe-Taylor. On specifying Boolean functions by labelled examples. *Discrete Applied Mathematics, 61*, 1-25, 1995.

[2] F. Barahona On the computational complexity of ising spin glasses. J.Phys A: Math. Gen., 15: 3241-3253, 1982.

[3] M. Ben-Or, M. and N. Linial. Collective coin flipping. In *Randomness and Computation* (S. Micali ed.) Academic Press, New York, pp. 91-115, 1989.

[4] A. Bernasconi, C. Damm and I. Shparlinski. The average sensitivity of square-freeness. *Computational Complexity, 9*, 39-51, 2000.

[5] R. B. Boppana. The Average Sensitivity of Bounded-Depth Circuits. *Information Processing Letters, 63*, 257-261, 1997.

[6] P. Chandra, P. Coleman and I. Ritchey. The Anisotropic Kagome Antiferromagnet: A Topical Spin Glass? *Journal de Physique I, 3*, 591-610, 1993.

[7] L. Franco. A measure for the complexity of Boolean functions related to their implementation in neural networks. Submitted. Available from: http://arxiv.org/pdf/cond-mat/0111169, 2002

[8] L. Franco and M. Anthony. The influence of opposite examples and randomness on the generalization ability of Boolean functions. CDAM Research Report, Centre for Discrete and Applicable Mathematics, London School of Economics, CDAM-LSE-2003-21, December 2003

[9] L. Franco and S. A. Cannas. Generalization and Selection of Examples in feedforward Neural Networks. *Neural Computation, 12*, 2405-2426, 2000.

[10] L. Franco and S. A. Cannas. Generalization properties of modular networks implementing the parity function. *IEEE Transactions in Neural Networks, 12*, 1306-1313, 2001.

[11] L. Franco and S. A. Cannas, Non glassy ground-state in a long-range antiferromagnetic frustrated model in the hypercubic cell. *Physica A, 332*, 337-348, 2004.

[12] A. Gál and A. Rosén. A theorem on sensitivity and applications in private computation. In: *Proceedings of the 31st ACM Symposium on the Theory of Computing* , 348-357, 1999.

[13] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Macmillan, 1994.

[14] S. Istrail. Statistical Mechanics, Three-dimensionality and NP-completeness I. Universality of intractability for the partition function of the Ising model Across non-planar lattices. STOC 2000 Portland Oregon USA, 2000.

[15] J. Kahn, G. Kalai and N. Linial. The influence of variables on Boolean functions. In: *Proceedings 29th IEEE Symposium on the Foundations of Computer Science (FOCS' 88)*, 68-80, 1988.

[16] N. Linial, Y. Mansour and N. Nisan. Constant depth circuits, Fourier transform and learnability. *Journal of the ACM, 40*, 607-620, 1993.

[17] V. I. Torvik and E. Triantaphyllou. Minimizing the Average Query Complexity of Learning Monotone Boolean Functions. *INFORMS Journal on Computing, 14*, 142-172, 2002.

[18] L. G. Valiant. A Theory of Learnable. *Communications of the ACM, 27*, 1134-1142, 1984.

[19] H. Zhu and W. Kinzel. Antipredictable sequences: harder to predict than random sequences. *Neural Computation, 10*, 2219-2230, 1998.