LA-UR- 97-226

CONF-9705196--1

Title: WHEN CONSTANTS ARE IMPORTANT

Author(s): Valeriu Beiu

Submitted to: The 11th International Conference on Control Systems and Computer Science
May 28-31, 1997
Bucharest, Romania

Los Alamos
NATIONAL LABORATORY

Form No. 836 R5
ST 2629 10/91

DISCLAIMER

When Constants Are Important

Valeriu Beiu *

Los Alamos National Laboratory, Division NIS-1, MS D466

Los Alamos, New Mexico 87545, USA

[phone: +1–505–667.2430 ❖ fax: +1–505–665.7395 ❖ e-mail: beiu@lanl.gov]

Abstract *— In this paper we discuss several complexity aspects pertaining to neural networks, commonly known as the 'curse of dimensionality'. The focus will be on: (i) size complexity and depth–size tradeoffs; (ii) complexity of learning; and (iii) precision and limited interconnectivity. Results have been obtained for each of these problems when dealt with separately, but few things are known as to the links amongst them. We start by presenting known results and try to establish connections between them. These show that we are facing very difficult problems — exponential growth in either space (i.e. precision and size) and/or time (i.e. learning and depth) — when resorting to neural networks for solving general problems. The paper will present a solution for lowering some constants, by playing on the depth–size tradeoff. Further directions for research are pointed out in the conclusions.*

Keywords *— neural networks, size complexity, complexity of learning, precision, interconnectivity (fan-in).*

1. Introduction

A *network* is an acyclic graph having several input nodes (*inputs*) and some (at least one) output nodes (*outputs*). If with each edge a synaptic *weight* is associated and each node computes the weighted sum of its inputs to which a nonlinear activation function is then applied (*artificial neuron*, or simply *neuron*):

$$f(\boldsymbol{x}) = f(x_1, \ldots, x_\Delta) = \sigma\left(\sum_{i=1}^{\Delta} w_i x_i + \theta\right), \tag{1}$$

the network is a *neural network* (NN), with $w_i \in \mathbb{R}$ called the synaptic *weights*, $\theta \in \mathbb{R}$ known as the *threshold*, Δ being the *fan-in* to one neuron, and σ a non-linear activation function. Such NNs are commonly characterised by two cost functions: (i) its *depth* (i.e. the number of layers); and (ii) its *size* (i.e. the number of neurons).

* On leave of absence from the Department of Computer Science, "Politehnica" University of Bucharest, Spl. Independenței 313, RO–77206 Bucharest, România.

The paper is structured in four parts. In section 2 we detail some previous results for the *size*, *precision* and *fan-in* required by NNs, and the complexity of learning. As the parameters involved are scaling exponentially, we shall present in section 3 a solution for lowering some constants relating to *size*. Conclusions as well as open problems are ending the paper.

2. Previous Results

NNs have been shown to be quite effective in many applications (see *Applications of Neural Networks* in [3], and *Part F: Applications of Neural Computation*, *Part G: Neural Networks in Practice: Case Studies* from [22]). This success has generated two directions of research:

- one to find existence/constructive proofs for the *"universal approximation problem;"*
- another one to find tight bounds on the *size* needed by the approximation problem.

2.1. Depth and Size

The first line of research has concentrated on the approximation capabilities of NNs. It was started in 1987 [26, 40, 41] by showing that Kolmogorov's superpositions [37] can be interpreted as a NN. The first nonconstructive proof has been given using a continuous activation function [19, 20, 29]. Different enhancements have been later presented (see overview in [12]), but all these results — with the partial exception of [4, 36, 39] — were obtained *"provided that sufficiently many hidden units are available."* This has led to more constructive solutions [35, 47, 48], and recently to an explicit numerical algorithm for superpositions [57]. These solutions are obtained in very small *depth*, but their *size* grows very fast.

The other line of research was to find the smallest *size* NN which can realize an arbitrary function given a set of m vectors from $\mathbb{R}^n$. Many results have been obtained for NNs having a *threshold* activation function [10, 55]. One of the first lower bounds on the *size* of a NN for "almost all" n-ary Boolean functions (BFs) was $size \geq 2\,(2^n/n)^{1/2}$ [46]. Later, a very tight upper bound $size \leq 2\,(2^n/n)^{1/2} \times \{1 + \Omega\,[(2^n/n)^{1/2}]\}$ has been proven in $depth = 4$ [42]. Similar exponential bounds can be found in [17], while [54] gives an $\Omega\,(2^{n/3})$ lower bound for ar-

bitrary BFs. For classification problems, one of the first results was that a NN of $depth = 3$ and $size = m - 1$ could compute an arbitrary dichotomy. The main improvements have been:

- Baum [6] presented a NN with one hidden layer having $\lceil m/n \rceil$ neurons[1] if the points are *in general position* (if the points are binary vectors, $m - 1$ nodes are needed);
- a slightly tighter bound $\lceil 1 + (m-2)/n \rceil$ was proven in [28] for a more relaxed topological assumption; the $m - 1$ condition was shown to be the least upper bound;
- Arai [2] showed that $m - 1$ hidden neurons are necessary, but improved the bound for the dichotomy problem to $m/3$ (without any condition on the inputs).

These results show that the *size* grows exponentially, as $m \leq 2^{n-1}$. The existence lower bounds for the arbitrary dichotomy problem are also exponential (see [24, 50]):

- a *depth*-2 NN requires at least $m/[n \log(m/n)]$ hidden neurons if $3n \leq m \leq 2^{n-1}$;
- a *depth*-3 NN requires at least $2(m/\log m)^{1/2}$ neurons in each of the two hidden layer if $n^2 \ll m \leq 2^{n-1}$;
- an arbitrarily interconnected NN without feedback needs $(2m/\log m)^{1/2}$ neurons if $n^2 \ll m \leq 2^{n-1}$.

One study tried to unify these two lines of research [18] by first presenting analytical solutions in one dimension (having infinite *size*!), and then giving practical solutions for the one-dimensional cases (i.e. including an upper bound on the *size*). Extensions to the n-dimensional case using three- and four-layers solutions were derived under piecewise constant approximations, and under piecewise linear approximations.

As can been seen, the known bounds for *size* are exponential for arbitrary functions, but:

- they reveal a gap between the upper and the lower bounds; and
- they suggest that ***NNs with more hidden layers (depth ≠ constant) have a smaller size.***

The only clear exception is given by Kolmogorov's superpositions theorem which shows that a NN having only $2n + 1$ neurons in the hidden layer can approximate any function.

[1] $\lceil x \rceil$ is the ceiling of x, i.e., the smallest integer greater than or equal to x, and $\lfloor x \rfloor$ is the floor of x, i.e., the largest integer less than or equal to x. In this paper all the logarithms are taken to base 2.

2.2. *Complexity of Learning*

The outcomes presented previously have proven the intrinsic power of NNs and have stimulated a more practical line of research for finding and/or improving learning algorithms.

These algorithms are based on the *learning ability* of NNs. The key idea is to progressively modify the *weights* during several *training phases*, such that the NN can finally perform a specific task. Many *learning rules* are inspired by the *generalized Hebb rule* [25]; still, there are algorithms (e.g., the constructive ones), which do not stick to this rule [9, 45]. This 'teaching' can be achieved in three ways: (i) *supervised*, when for each input of the learning set, the desired output is known; (ii) *reinforced*, when a reward is given to the NN by the environment on its response to a given input; (iii) *unsupervised*, when the goal associated to each input is not pre-defined. Learning techniques suffer from the inherent error correction function (which does not guarantee that the global minimum will ever be reached), and from the very long time needed for solving the problem [30, 31, 32, 34, 53] (few exceptions are known for particular cases [7, 33]). Minsky and Papert [44] showed that training times increase very rapidly and have even written that: *"... the entire structure of recent connectionist theories might be built on quicksand: it is all based on toy-sized problems with no theoretical analysis to show that performance will be maintained when the models are scaled up to realistic size. The connectionist authors fail to read our work as a warning that networks, like brute force, scale very badly."* The hope that learning time could be reduced by using more complex activation functions has fallen short of its expectations; it has been proven that training a 3-node NN is: (i) NP-complete if σ is the linear threshold function [16]; (ii) NP-complete if σ is the piecewise linear activation function [21]; (iii) NP-hard if σ is a sigmoid activation function [56].

The constructive class of learning algorithms [9] (also some of the so-called VLSI-friendly learning algorithms [45]), are successively adding neurons and/or layers for obtaining a better and better approximation on the given data-set. Because for certain problems the *size* grows exponentially, such algorithms will also require an exponential time to build the network.

2.2. *Precision and Interconnectivity*

Because most NNs are (and have been) simulated, two aspects which have been in too many cases neglected are the *precision* of *weights* and *thresholds* and the *fan-in* of the neurons. When hardware implementations of NNs have been thought of [38, 45], *precision* and *interconnectivity* became important [15].

Finite *precision* computation has started to be analysed [27, 59], and it was shown that in most cases several bits suffice [10, 12, 45]. Recently it was proven [5] that the generalization error of NNs used for classification depends on the size of the *weights*, rather than the number of *weights* by showing that the misclassification probability converges as $\boldsymbol{O}\left((cA)^{l}/\sqrt{m}\right)$. Here A is the sum of the magnitude of the *weights*, l is the *depth*, m is the number of examples, and c is a constant. Beside supporting heuristics that attempt to keep the *weights* small during training, this also ***suggests that NNs having more layers are converging faster !***

Interconnectivity has been analysed in relation to the *area* of a VLSI chip [23], and it was shown that the *area* grows as the cube of the node's *fan-in* (Δ^3). Recently, the fact that AT^2–optimal digital circuits are obtained for small constant *fan-ins* has been proven [11].

The known *weight* bounds: $1.618^{\Delta} < \textit{weight} < (\Delta+1)^{(\Delta+1)/2}/2^{\Delta}$, show that we can expect to need between Δ and $\Delta \log \Delta$ bits per *weight* [49, 51]. They also show that *interconnectivity* and *precision* are interrelated, and we should mention that *fan-in* also relates to *depth*, as smaller *fan-ins* lead to deeper NNs (larger *depths*).

3. Lowering Some Constants

The starting point is $\mathbb{F}_{k,m}$ which has been defined as *"the class of Boolean functions* $f(x_1, x_2, \ldots, x_k)$ *that have exactly m groups of ones"* [52]. By allowing m to grow exponentially with respect to k, $\mathbb{F}_{k,m} \equiv \mathbb{B}_k$ (the set of all k-ary Boolean functions). Red'kin [52] presents a solution (based on COMPARISONs) achieving an excellent $2\sqrt{m}+3$ *size* in *depth*-3, but requiring exponential *precision* and polynomial *fan-in*. Another solution [9, 14], having polynomial

weights and *thresholds* ($weights < 2^{\Delta/2}$), constructs *fan-in* $= \Delta$ NNs of $\boldsymbol{O}(mk/\Delta)$ *size* and $\boldsymbol{O}(\log(mk)/\log\Delta)$ *depth*. It can be adapted to accept real inputs by properly quantizing the input space [14]. Based on computing the entropy of the data-set, the way the quantization should be done was detailed [8]. It links both to *size* complexity [58], and to *connectivity* [1]. The result is that the number of bits required to solve a dichotomy is $\boldsymbol{O}(mn)$, or more precisely:

$$k = \#bits < mn \cdot [\log(D/d) + 5/2] \tag{2}$$

where D is the largest, and d the smallest distance between examples from the two different classes. The *size* of the NN depends on this quantization process [13]. Equation (2) has been obtained using a whole n-dimensional ball of radius D and volume $\pi^{n/2}D^n/\Gamma(n/2+1)$ to upper bound the space containing the examples. Because the examples belonging to one class are always inside the intersection of two n-dimensional balls, the result can be improved by computing the volume of the intersection of two n-dimensional balls [13], giving:

$$k = \#bits < \frac{mn}{2} \cdot [\log(D/d) + 1.8396]$$

and showing reduction on some constants (1/2 instead of 1; 1.8396 instead of 2.5).

The *size* complexity of the NN implementing one $\mathbb{F}_{k,m}$ function has been computed as:

$$size(k, m, \Delta) = km \cdot \left[\frac{1}{\Delta/2} + \frac{1}{(\Delta/2)^2} + \ldots + \frac{1}{(\Delta/2)^{depth}}\right], \tag{3}$$

[12, 14], but a substantial enhancement can be obtained as the *fan-in* is limited. Due to that limitation, the maximum number of ***different*** BFs which can be computed in each layer is:

$$(2k/\Delta) \cdot 2^{\Delta}, \quad \frac{2k/\Delta}{\Delta/2} \cdot 2^{\Delta(\Delta/2)}, \quad \ldots, \quad \frac{2k/\Delta}{(\Delta/2)^{depth-1}} \cdot 2^{\Delta(\Delta/2)^{depth-1}}. \tag{4}$$

If m and/or k are large (enough), the first terms in (3) will be larger than the equivalent ones in (4) because the same BFs are redundantly computed many times. The simple solution to lower the *size* is to use the smallest term given either by (3) or by (4).

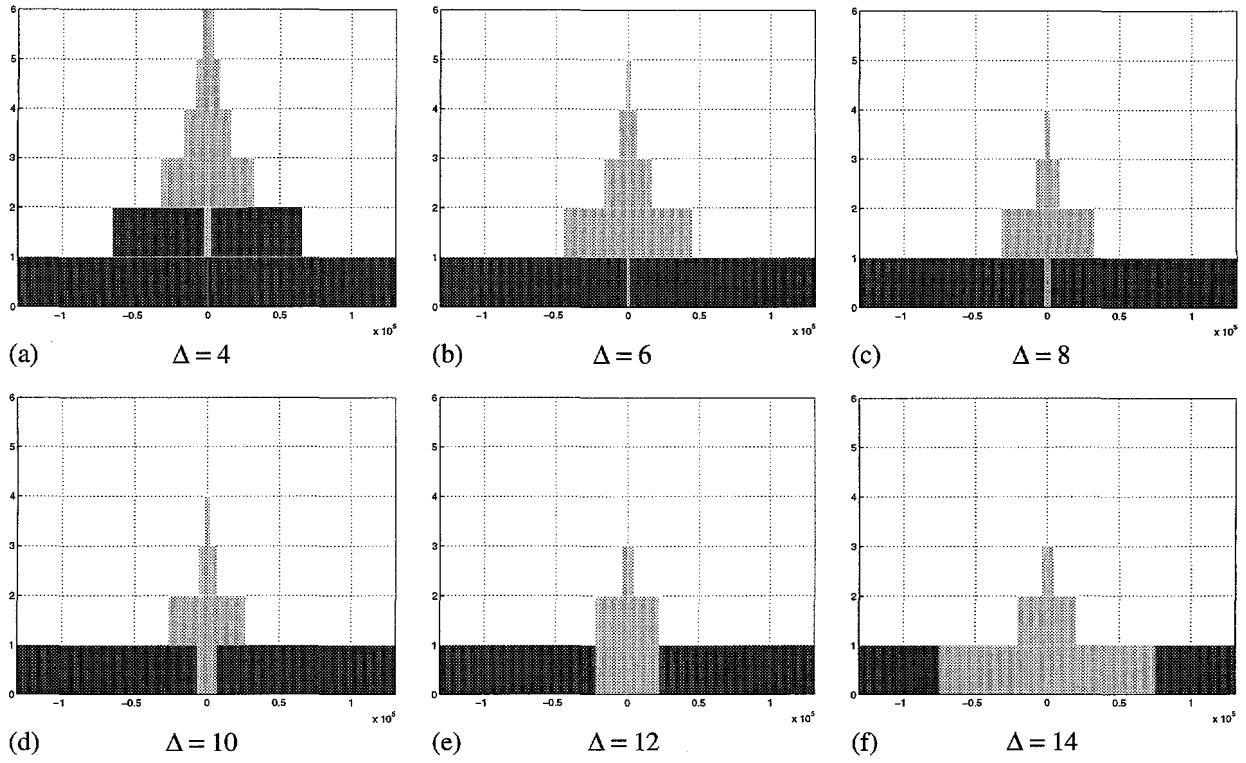

Figure 1. Different NNs (obtained by varying the *fan-in*) for solving any classification problem defined by 2048 examples having 64 binary inputs each (for more explanations see text).

For a better understanding we have considered the following example: $k = 64$ binary inputs (roughly equivalent to $n = 8$ real inputs) and $m = 2048$ examples (an extremely small set as compared to $2^{64} = 1.84 \cdot 10^{19}$). Different solutions have been obtained by varying the *fan-in*, and some of them can be seen in Figure 1, where the vertical axis represents the number of layers, while the area of the rectangles represents the number of gates (in each layer). The area of dark tinted rectangles represent redundant neurons which can be discarded. The area of light tinted rectangles represents the number of needed neurons in each layer. The *size* of the dif-

Table 1.

The exact *size* and *depth* for 2048 examples of 64 bits each when varying the *fan-in*.

Fan-in	4	5	6	7	8	9	**10**	11	12	13	14	15	16	32	64
Depth	6	5		4					3						2
Size	127,488	133,865	140,630	112,933	90,112	89,202	**82,740**	89,368	96,939	129,812	198,949	307,200	299,008	282,624	270,336

ferent NNs are represented by the sum of the area of all the light tinted rectangles (instead of the area of all the rectangles). Numerical results are presented in Table 1. For this example, the smallest NN is obtained for $\Delta = 10$, in $depth = 4$, while ***the size is reduced from 331,776 neurons to 82,740 neurons, i.e. by 75% !*** This is significant, and for larger values of m and/or k the percentage is even larger.

4. Conclusions and Open Problem

In this paper we have first presented a short overview for some complexity aspects relating to NNs. As most known bounds are exponential, we have then focused on reducing some constants, and have shown constructively how to lower the *size* of NNs for classification problems. An open question is to compute a bound for the last solution we have detailed.

Relating to Kolmogorov's theorem two open questions are: (i) can we determine the synaptic *weights* of such a NN in a reasonable amount of time (and if yes, how); and (ii) which is the *precision* required by these *weights*.

References

[1] Y.S. Abu-Mostafa. Connectivity versus Entropy. In D.Z. Anderson (ed.): *Neural Information Processing Systems* (NIPS*87, Denver, CO), American Institute of Physics, NY, 1-8, 1988.

[2] M. Arai. Bounds on the Number of Hidden Units in Binary-valued Three-layer Neural Networks. *Neural Networks*, **6**(6), 855-860, 1993.

[3] M.A. Arbib (ed.). *The Handbook of Brain Theory and Neural Networks*. MIT Press, Cambridge, MA, 1995.

[4] A.R. Barron. Universal Approximation Bounds for Superpositions of a Sigmoidal Function. *IEEE Trans. on Information Theory*, **39**(3), 930-945, 1993.

[5] P.L. Bartlett. The Sample Complexity of Pattern Classification with Neural Networks: The Size of the Weights Is More Important than the Size of the Network. *Tech. Rep.*, Dept. of Sys. Eng., Res. Sch. of Info. Sci. & Eng., Australian Nat. Univ. (Canberra), Australia (ftp: syseng.anu.edu.au/pub/peter/TR96d.ps.Z), May 1996. Extended abstract in *NIPS*96*, Denver, CO, 1997.

[6] E.B. Baum. On the Capabilities of Multilayer Perceptrons. *J. of Complexity*, **4**, 193-215, 1988.

[7] E.B. Baum. Neural Net Algorithm that Learn in Polynomial Time from Examples and Queries. *IEEE Trans. on Neural Networks*, **2**(1), 5-19, 1991.

[8] V. Beiu. Entropy Bounds for Classification Algorithms. *Neural Network World*, **6**(4), 497-505, 1996.

[9] V. Beiu. Optimal VLSI Implementations of Neural Networks: VLSI-Friendly Learning Algorithms. *Chapter 18* in J.G. Taylor (ed.): *Neural Networks and Their Applications*, John Wiley & Sons, Chichester, 255-276, 1996.

[10] V. Beiu. Digital Integrated Circuit Implementations. *Chapter E1.4* in [22], 1996.

[11] V. Beiu. Constant Fan-In Digital Neural Networks Are VLSI-Optimal. In S.W. Ellacott, J.C. Mason and I.J. Anderson (eds.): *Mathematics of Neural Networks: Models, Algorithms and Applications*, Kluwer Academic Press, 1997 (in press).

[12] V. Beiu. *VLSI Complexity of Discrete Neural Networks*. Gordon and Breach & Harwood Academics Publishing, Newark, 1997 (in press).

[13] V. Beiu and T. de Pauw. Tight Bounds on the Size of Neural Networks for Classification Problems. Article submitted and under review, 1997.

[14] V. Beiu and J.G. Taylor. Direct Synthesis of Neural Networks. *Proc. MicroNeuro'96* (Lausanne, Switzerland), IEEE CS Press, Los Alamitos, CA, 257-264, 1996.

[15] V. Beiu and J.G. Taylor. On the Circuit Complexity of Sigmoid Feedforward Neural Networks. *Neural Networks*, **9**(7), 1155-1171, 1996.

[16] A.L. Blum and R.L. Rivest. Training a 3-Node Neural Network Is NP-Complete. *Neural Networks*, **5**(1), 117-127, 1992.

[17] J. Bruck and R. Smolensky. Polynomial Threshold Functions, AC^0 Functions and Spectral Norms, *Res. Rep. RJ 7410* (67387), IBM Yorktown Heights, NY, 11/15/1989; also in *SIAM J. Computing*. **21**(1) 33-42, 1992.

[18] A. Bulsari. Some Analytical Solutions to the General Approximation Problem for Feedforward Neural Networks. *Neural Networks*, **6**(7), 991-996, 1993.

[19] G. Cybenko. Continuous Valued Neural Networks with Two Hidden Layers Are Sufficient. *Tech. Rep.*, Tufts University, 1988.

[20] G. Cybenko. Approximations by Superpositions of a Sigmoid Function. *Math. of Control, Signals and Systems*, **2**, 303-314, 1989.

[21] B. DasGupta, H.T. Siegelmann and E.D. Sontag. On the Complexity of Training Neural Networks with Continuous Activation Functions. *Tech. Rep. 93-61*, Department of CS, University of Minnesota, 1993; also in *IEEE Trans. on Neural Networks*, **6**, 1490-1504, 1995.

[22] E. Fiesler and R. Beale (eds.). *Handbook of Neural Computation*. Oxford Univ. Press and the Inst. of Physics Publishing, NY, 1996.

[23] D. Hammerstrom. The Connectivity Analysis of Simple Association –or– How Many Connections Do You Need. In D.Z. Anderson (ed.): *Neural Information Processing Systems* (NIPS*87, Denver, CO), Amer. Inst. of Physics, NY, 338-347, 1988.

[24] M.H. Hassoun. *Fundamentals of Artificial Neural Networks*. MIT Press, Cambridge, MA, 1995.

[25] D.O. Hebb. *The Organization of Behaviour*. Wiley, New York, 1949.

[26] R. Hecht-Nielsen. Kolmogorov's Mapping Neural Network Existence Theorem. *Proc. IEEE Intl. Conf. on Neural Networks*, IEEE Press, 11-14, June 1987.

[27] J.L. Holt and J.-N. Hwang. Finite Precision Error Analysis of Neural Network Hardware Implementations. *IEEE Trans. on Comp.*, **42**(3), 281-290, 1993.

[28] S.-C. Huang and Y.-F. Huang. Bounds on the Number of Hidden Neurons of Multilayer Perceptrons in Classification and Recognition. *IEEE Trans. on Neural Networks*, **2**(1), 47–55, 1991.

[29] B. Irie and S. Miyake. Capabilities of Three-Layered Perceptrons. *Proc. Intl. Conf. on Neural Networks ICNN'88* (San Diego, CA), vol. **1**, 641-648, 1988.

[30] J.S. Judd. Learning in Neural Networks Is Hard. *Proc. IEEE Intl. Conf. on Neural Networks*, IEEE Press, 685-692, June 1987.

[31] J.S. Judd. On the Complexity of Loading Shallow Neural Networks. *J. of Complexity*, **4**, 177-192, 1988.

[32] J.S. Judd. *Neural Network Design and the Complexity of Learning*. MIT Press, Cambridge, MA, 1990.

[33] J.S. Judd. Constant-Time Loading of Shallow 1-Dimension Networks. *Tech. Rep.*, Siemens Corporate Research, Princeton, New Jersey, 1992. Also in J.E. Moody, S.J. Hanson and R.P. Lippmann (eds.): *Advances in Neural Inform. Proc. Sys. 4* (NIPS*91, Denver, CO), Morgan Kaufmann, San Mateo, CA, 863-870, 1992.

[34] J.S. Judd. Time Complexity of Learning. In [3], 984-987, 1995.

[35] H. Katsuura and D.A. Sprecher. Computational Aspects of Kolmogorov's Superposition Theorem. *Neural Networks*, **7**(3), 455-461, 1993.

[36] P. Koiran. On the Complexity of Approximating Mappings Using Feedforward Networks. *Neural Networks*, **6**(5), 649-653, 1993.

[37] A.N. Kolmogorov. On the Representation of Continuous Functions of Many Variables by Superposition of Continuous Functions of One Variable and Addition. *Dokl. Akad. Nauk SSSR*, **114**, 953-956, 1957. English translation in *Transl. Amer. Math. Soc.*, **2**(28), 55-59, 1963.

[38] A.V. Krishnamoorthy, R. Paturi, M. Blume, G.D. Linden, L.H. Linden and S.C. Esener. Hardware Tradeoffs for Boolean Concept Learning. *Proc. World Conf. on Neural Networks '94* (San Diego, CA), Lawrence Erlbaum Associates, Inc., and INNS Press, Hillsdale, vol. **1**, 551-559, 1994.

[39] V. Kůrková. Kolmogorov's Theorem and Multilayer Neural Networks. *Neural Networks*, **5**(4), 501-506, 1992.

[40] Y. LeCun. *Models connexionistes de l'apprentisage*. M.Sc. thesis, Université Pierre et Marie Curie, Paris, 1987.

[41] R.P. Lippmann. An Introduction to Computing with Neural Nets. *IEEE ASSP Mag.*, **4**(2), 4-22, 1987.

[42] O.B. Lupanov. The Synthesis of Circuits from Threshold Elements. *Problemy Kibernetiki*, **20**, 109-140, 1973.

[43] G.A. Miller. The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information. *Psych. Rev.*, **63**, 71-97, 1956.

[44] M.L. Minsky and S.A. Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, MA, 1969 (second edition 1989).

[45] P.D. Moerland and E. Fiesler. Neural Network Adaptations to Hardware Implementations. *Chapter E1.2* in [22], 1996.

[46] E.I. Neciporuk. The Synthesis of Networks from Threshold Elements. *Problemy Kibernetiki*, **11** 49-62, 1964. English translation in *Automation Express*, **7**(1), 35-39 and **7**(2), 27-32, 1964.

[47] M. Nees. Approximate Versions of Kolmogorov's Superposition Theorem, Proved Constructively. *J. of Computational and Applied Math.*, **54**(2), 239-250, 1994.

[48] M. Nees. Chebyshev Approximation by Discrete Superposition. Application to Neural Networks. *Advances in Computational Mathematics*, **5**(2), 137-152, 1996.

[49] I. Parberry. *Circuit Complexity and Neural Networks*. MIT Press, Cambridge, MA, 1994.

[50] H. Paugam-Moisy. *Optimisation des réseaux des neurones artificiels*. Ph.D. thesis, Laboratoire de l'Informatique du Parallélisme LIP-IMAG, École Normale Supérieure de Lyon, 46 Allée d'Italie, 69364 Lyon, France, 1992.

[51] P. Raghavan. Learning in Threshold Networks: A Computational Model and Applications. *Tech. Rep. RC 13859*, IBM Res., 1988 (also in *Proc. 1st Workshop on Computational Learning Theory*, ACM Press, 19-27, 1988).

[52] N.P. Red'kin. Synthesis of Threshold Circuits for Certain Classes of Boolean Functions. *Kibernetika*, **6**(5), 6-9, 1970.

[53] V.P. Roychowdhury, K.-Y. Siu and A. Orlitsky (eds.). *Theoretical Advances in Neural Computation and Learning*. Kluwer Academic, Boston, 1994.

[54] K.-Y. Siu, V. Roychowdhury and T. Kailath. Depth-Size Tradeoffs for Neural Computations. *IEEE Trans. on Comp.*, **40**(12), 1402-1412, 1991.

[55] K.-Y. Siu, V.P. Roychowdhury and T. Kailath. *Discrete Neural Computation: A Theoretical Foundation*. Prentice Hall, Englewood Cliffs, 1994.

[56] J. Šíma. Back-propagation is not Efficient. *Neural Networks*, **9**(6), 1017-1023, 1996.

[57] D.A. Sprecher. A Numerical Implementation of Kolmogorov's Superpositions. *Neural Networks*, **9**(5), 765-772, 1996.

[58] R.C. Williamson. ε-entropy and the Complexity of Feedforward Neural Networks, in R.P. Lippmann, J.E. Moody and D.S. Touretzky (eds.): *Neural Information Processing Systems* (NIPS*90, Denver, CO), Morgan Kaufmann, San Mateo, 946-952, 1991.

[59] J. Wray and G.G.R. Green, "Neural Networks, Approximation Theory, and Finite Precision Computation," *Neural Networks*, **8**(1), 31-37, 1995.