LA-UR- 97-3493

Title: SMALL FAN-IN IS BEAUTIFUL

CONF-980538--

Author(s): Valeriu Beiu
Hanna E. Makaruk

Submitted to: IEEE INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS
ANCHORAGE, ALASKA
MAY 5-9, 1998

Los Alamos
NATIONAL LABORATORY

Form No. 836 R5
ST 2629 10/91

Small Fan-In Is Beautiful

Valeriu Beiu [1] and Hanna E. Makaruk [2]

Space & Atmospheric Div. NIS-1, MS D466, and Theoretical Div. T-13, MS B213
Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA
E-mail: beiu@lanl.gov and hanna@t13.lanl.gov

Abstract

The starting points of this paper are two size-optimal solutions: (i) one for implementing arbitrary Boolean functions [12]; and (ii) another one for implementing certain subclasses of Boolean functions [18]. Because VLSI implementations do not cope well with highly interconnected nets—the area of a chip grows with the cube of the fan-in [11]—this paper will analyse the influence of limited fan-in on the size optimality for the two solutions mentioned. First, we will extend a result from Horne & Hush [12] valid for fan-in $\Delta = 2$ to arbitrary fan-in. Second, we will prove that size-optimal solutions are obtained for small constant fan-ins for both constructions, while relative minimum size solutions can be obtained for fan-ins strictly lower that linear. These results are in agreement with similar ones proving that for small constant fan-ins ($\Delta = 6 \ldots 9$) there exist VLSI-optimal (i.e., minimising AT^2) solutions [6], while there are similar small constants relating to our capacity of processing information [16].

1: Introduction

In this paper we shall consider *feedforward neural networks* (NNs) made of linear threshold gates (TGs), or perceptrons. A TG is computing a Boolean function (BF):

$$f: \{0, 1\}^n \rightarrow \{0, 1\},$$

where an input vector is $\mathbf{Z}_k = (z_{k,0}, \ldots, z_{k,n-1})$, and:

$$f(\mathbf{Z}_k) = sgn\left(\textstyle\sum_{i=0}^{n-1} w_i z_{k,i} + \theta\right).$$

The notations used are $w_i \in \mathbb{R}$ for the synaptic *weights*, $\theta \in \mathbb{R}$ for the *thresholds*, and *sgn* for the sign (nonlinear activation) function.

[1] On leave of absence from the "Politehnica" University of Bucharest, Computer Science Department, Spl. Independenței 313, RO-77206 Bucharest, România.

[2] On leave of absence from the Polish Academy of Sciences, Institute of Fundamental Technological Research, Świetokrzyska 21, 00-049 Warsaw, Poland.

The *cost functions* commonly associated to a NN are:

- *depth* (*i.e.*, number of edges on the longest input-to-output path, or number of layers); and
- *size* (*i.e.*, number of neurons).

However, the *area* of the connections counts, and the *area* of one neuron can be related to its associated *weights*, thus *"comparing the number of nodes is inadequate for comparing the complexity of NNs as the nodes themselves could implement quite complex functions"* [25]. That is why several authors have taken into account the total *number-of-connections* [1, 11, 15, 17], or the total *number-of-bits* needed to represent the *weights* and the *thresholds* [9, 10], or the sum of all the *weights* and the *thresholds* [3]. The sum of all the *weights* and the *thresholds* (also applied for defining the minimum-integer TG realisation of a BF) has been recently used—under the name of *"total weight magnitude"*—in the context of computational learning theory for improving on several VC-theory bounds [2]. A quite similar definition of 'complexity' $\sum w_i^2$ has also been advocated [27]. Such approximations can easily be related to assumptions on how the *area* of a chip scales with the *weights* and the *thresholds* [5, 8]:

- for digital implementation, the *area* scales with the cumulative storage of *weights* and *thresholds* (as the bits for representing those *weights* and *thresholds* have to be stored);
- for analog implementations (*e.g.*, using resistors or capacitors) the same type of scaling is valid (although it is possible to come up with implementations having binary encoding of the parameters—for which the *area* would scale with the cumulative log-scale size of the parameters);
- some types of implementations (*e.g.*, transconductance ones) even offer a constant size per element, thus in principle scaling only with the number of parameters (*i.e.*, with the total *number-of-connections*).

With respect to *delay*, two VLSI models have been commonly in use [22]:

- the simplest one assumes that *delay* is proportional to the input capacitance, hence a TG introduces a *delay* proportional to its *fan-in*;
- a more exact one considers the capacitance along any wire, hence the *delay* is proportional to the *length* of the connecting wires.

It is worth emphasising that it is anyhow desirable to limit the range of parameter values [26] for VLSI implementations because: (*i*) the maximum value of the *fan-in* [13, 24]; and (*ii*) the maximal ratio between the largest and the smallest *weight* cannot grow over a certain (technological) limit [10, 14].

The focus of this paper will be on NNs having limited *fan-in* (the *fan-in* will be denoted by Δ), and we will discuss the influence of limiting the *fan-in* on the *size* optimality of two different *size*-optimal solutions. We will present both theoretical proofs and simulation results in support of our claim that the two *size*-optimal NN solutions can be obtained for small *fan-ins*. For simplification, we shall consider only NNs having n binary inputs and m binary outputs (if real inputs and outputs are needed, it is always possible to quantize them up to a certain number of bits such as to achieve a desired precision [8]). Section 2 will present two pervious results dealing with arbitrary BFs [12], and with $\mathbb{F}_{n,m}$ functions [18]. In Section 3 we will first generalize those results to arbitrary *fan-ins*, and then show that the *size* can be minimized for small *fan-ins*. Conclusions, and a discussion of open questions and further directions for research complete the paper. Due to space limitations some of the lengthy mathematical proofs have been omitted, but the interested reader can find them in [5, 6, 8].

2: Previous results

One starting point is a classic construction for synthesising one BF with *fan-in* 2 AND-OR gates. It was extended to the multioutput case and modified to apply to NNs.

Proposition 1 (Theorem 3 from [12]) *Arbitrary Boolean functions of the form $f:\{0,1\}^n \rightarrow \{0,1\}^m$ can be implemented in a neural network of perceptrons restricted to $\Delta = 2$, with a node complexity of $\Theta\{m\,2^n/(n+\log m)\}$, and requiring $O(n)$ layers.*

Sketch of proof The idea is to decompose each BF into two subfunctions using Shannon's Decomposition [20]:

$$f(x_1 x_2 \ldots x_{n-1} x_n) = \bar{x}_1 f_0(x_2 \ldots x_{n-1} x_n) + x_1 f_1(x_2 \ldots x_{n-1} x_n).$$

By doing this recursively for each subfunction, the output BFs will be implemented by binary trees. Horne & Hush [12] use a trick for eliminating most of the lower level nodes by replacing them with a subnetwork that computes *all the possible BFs* needed by the higher level nodes. Each subcircuit eliminates one variable and has three nodes (one OR and two ANDs). Thus the upper tree has:

$$size_{upper} = 3\,m \times \sum_{i=0}^{n-q-1} 2^i = 3\,m\,(2^{n-q}-1) \tag{1}$$

nodes, and $depth_{upper} = 2\,(n-q)$. These subfunctions now depend on only q variables, and a lower subnetwork that computes all the possible BFs of q variables is built. It has:

$$size_{lower} = 3 \times \sum_{i=1}^{q} 2^{2^i} < 4 \cdot 2^{2^q} \tag{2}$$

nodes, and $depth_{lower} = 2\,q$ (see also Figure 2 in [12]).

That q which minimises $size_{BFs} = size_{upper} + size_{lower}$ is determined by solving $d\,(size_{BFs})/dq = 0$, and gives:

$$q \approx \log\{n + \log m - 2\log(n + \log m)\}. \tag{3}$$

By substituting (3) in (1) and (2), the minimum *size*

$$size_{BFs} \approx 3m\,2^{n-q} \approx 3m\,2^n/(n+\log m)$$

can be determined. ❑

Proposition 2 (Theorem 1 from [18]) *The complexity realisation (i.e., number of threshold elements) of $\mathbb{F}_{n,m}$ (the class of Boolean functions $f(x_1 x_2 \ldots x_{n-1} x_n)$ that have exactly m groups of ones) is at most $2\,(2m)^{1/2} + 3$.*

The construction has: a first layer of $\lceil (2m)^{1/2} \rceil$ TGs (COMPARISONs) with $fan\text{-}in = n$ and $weights \le 2^{n-1}$; a second layer of $2\lceil (m/2)^{1/2} \rceil$ TGs of $fan\text{-}in = n + \lceil (2m)^{1/2} \rceil$ and $weights \le 2^n$; one more TG of $fan\text{-}in = 2\lceil (m/2)^{1/2} \rceil$ and $weights \in \{-1, +1\}$ in the third layer.

3: Limited fan-in and optimal solutions

Proposition 3 (this paper) *Arbitrary Boolean functions $f:\{0,1\}^n \rightarrow \{0,1\}^m$ can be implemented in a neural network of perceptrons having the fan-in limited to Δ in $O(n/\log\Delta)$ layers.*

Proof We use the same approach as Horne & Hush [12] for the case when the *fan-in* is limited to Δ. Each output BF can be decomposed in $2^{\Delta-1}$ subfunctions (*i.e.*, $2^{\Delta-1}$ AND gates). The OR gate would have $2^{\Delta-1}$ inputs. Thus we have to decompose it in a Δ-ary tree of $fan\text{-}in = \Delta$ OR gates. This decomposition step eliminates $\Delta - 1$ variables and generates a Δ-ary tree having:

$$depth = 1 + \lceil (\Delta-1)/\log\Delta \rceil, \text{ and}$$

$$size = 2^{\Delta-1} + \lceil (2^{\Delta-1}-1)/(\Delta-1) \rceil.$$

Repeating this procedure recursively k times:

$$depth_{upper} = k \cdot \{1 + \lceil (\Delta-1)/\log\Delta \rceil\} \tag{4}$$

$$size_{upper} = \{2^{\Delta-1} + \lceil (2^{\Delta-1}-1)/(\Delta-1) \rceil\} \times \sum_{i=0}^{k-1} 2^{i(\Delta-1)}$$

$$= size \cdot (2^{k(\Delta-1)} - 1)/(2^{\Delta-1} - 1)$$

$$\cong 2^{k(\Delta-1)}(1 + 1/\Delta) \approx 2^{k\Delta - k} \quad (5)$$

where the subfunctions depend only on $q = n - k\Delta$ variables. We now generate all the possible subfunctions of q variables with a subnetwork of:

$$depth_{lower} = \lfloor (n - k\Delta)/\Delta \rfloor \cdot \{1 + \lceil (\Delta - 1)/\log\Delta \rceil\} \quad (6)$$

$size_{lower}$

$$= \{2^{\Delta-1} + \lceil (2^{\Delta-1} - 1)/(\Delta-1) \rceil\} \times \sum_{i=1}^{\lfloor n/\Delta \rfloor - k} 2^{2^{n-k\Delta-i\Delta}}$$

$$= size \cdot \left\{ 2^{2^0} + 2^{2^\Delta} + \ldots + 2^{2^{n-(k+1)\Delta}} \right\}$$

$$< (size+1) \cdot 2^{2^{n-(k+1)\Delta}} \quad (7)$$

$$\approx 2^\Delta \cdot 2^{2^{n-k\Delta-\Delta}}. \quad (8)$$

The inequality (7) can be proved by induction; clearly:

$$size \cdot 2^{2^0} < (size+1) \cdot 2^{2^0}.$$

Consider the statement true for α; we prove it for $\alpha + 1$:

$$size \cdot \left\{ 2^{2^0} + 2^{2^\Delta} + \ldots + 2^{2^{\alpha\Delta}} \right\} + size \cdot 2^{2^{(\alpha+1)\Delta}}$$

$$< size \cdot 2^{2^{(\alpha+1)\Delta}} + 2^{2^{(\alpha+1)\Delta}}$$

$$size \cdot \left\{ 2^{2^0} + 2^{2^\Delta} + \ldots + 2^{2^{\alpha\Delta}} \right\} < (size+1) \cdot 2^{2^{\alpha\Delta}}$$

(due to hypothesis), thus:

$$(size+1) \cdot 2^{2^{\alpha\Delta}} < 2^{2^{(\alpha+1)\Delta}}$$

and computing the logarithm of the left side:

$$2^{\alpha\Delta} + \log(size+1)$$

$$= 2^{\alpha\Delta} + \log\{2^{\Delta-1} + \lceil (2^{\Delta-1} - 1)/(\Delta-1) \rceil\}$$

$$< 2^{\alpha\Delta} + \log\{2^{\Delta-1} + 2^{\Delta-1}/\Delta + 1\}$$

$$< 2^{\alpha\Delta} + \Delta$$

$$< 2^{(\alpha+1)\Delta}.$$

From (4) and (6) we can estimate $depth_{BFs}$, and from (5) and (8) $size_{BFs}$ as:

$$depth_{BFs} = \{k + \lfloor (n - k\Delta)/\Delta \rfloor\} \cdot \{1 + \lceil (\Delta-1)/\log\Delta \rceil\}$$

$$= (n/\Delta) \cdot (\Delta/\log\Delta + 1) \quad (9)$$

$$\approx n/\log\Delta = O(n/\log\Delta)$$

$size_{BFs}$

$$= m \cdot size \cdot (2^{k(\Delta-1)} - 1)/(\Delta - 1) + (size+1) \cdot 2^{2^{n-(k+1)\Delta}}$$

$$\approx m \cdot 2^{k\Delta - k} + 2^\Delta \cdot 2^{2^{n-k\Delta-\Delta}} \quad (10)$$

concluding the proof. ❑

Proposition 4 (this paper) *All the critical points of $size_{BFs}(m, n, k, \Delta)$ are relative minimum, and are situated in the vicinity of the parabola $k\Delta \approx n - \log(n + \log m)$.*

Proof To determine the critical points, we equate the partial derivatives to zero. Starting from the approximation of $size_{BFs}$ we compute $\partial size_{BFs}/\partial k = 0$, which gives:

$$m \cdot 2^{k\Delta - k}(\ln 2)(\Delta - 1) +$$

$$+ 2^\Delta \cdot 2^{2^{n-k\Delta-\Delta}}(\ln 2) \cdot 2^{n-k\Delta-\Delta}(\ln 2) \cdot (-\Delta) = 0$$

$$\{m(\Delta-1)/\Delta/(\ln 2)\} \cdot 2^{2k\Delta - k - n} = 2^{2^{n-k\Delta-\Delta}}.$$

Using the following notations:

$$k\Delta = \gamma,$$

$$\beta = m(\Delta - 1)/(\Delta \ln 2),$$

and taking logarithms of both sides:

$$\log\beta + 2\gamma - k - n = 2^{n-\gamma-\Delta} \quad (11)$$

an approximaye solition is:

$$\gamma \approx n - \log(n + \log m).$$

An alternate solution leading to the same result can be obtained by computing with finite differences:

$$size_{BFs}(m, n, k+1, \Delta) - size_{BFs}(m, n, k, \Delta) = 0$$

$$size \cdot \left\{ m \cdot 2^{k\Delta - k} - 2^{2^{n-k\Delta-\Delta}} \right\} = 0$$

$$m \cdot 2^{k\Delta - k} = 2^{2^{n-k\Delta-\Delta}}$$

which—after taking twice the logarithm of both sides, and using the same notations—gives:

$$\log\{\log m + \gamma(1 - 1/\Delta)\} = n - \gamma - \Delta$$

$$\gamma = n - \{\Delta + \log(1 - 1/\Delta)\} -$$

$$- \log\{\gamma + \Delta/(\Delta-1) \cdot \log m\} \quad (12)$$

$$\approx n - \Delta - \log(\gamma + \log m).$$

AN approximate solution is:

$$\gamma \approx n - \log(n + \log m).$$

Starting again from $size_{BFs}$ as given by equation (10), we compute $\partial size_{BFs}/\partial\Delta = 0$. We have:

$$m\, 2^{k\Delta - k}(\ln 2)\, k + 2^\Delta(\ln 2)\, 2^{2^{n-k\Delta-\Delta}} +$$

$$+ 2^\Delta\, 2^{2^{n-k\Delta-\Delta}}(\ln 2)\, 2^{n-k\Delta-\Delta}(\ln 2)(-k) = 0$$

$$mk \cdot 2^{\gamma - k}$$

$$= k(\ln 2) \cdot 2^{n-\gamma} \cdot 2^{2^{n-\gamma-\Delta}} - 2^\Delta \cdot 2^{2^{n-\gamma-\Delta}}$$

$$mk \cdot 2^{\gamma-k} \cdot 2^{\gamma-n}$$

$$= k(\ln 2) \cdot 2^{2^{n-\gamma-\Delta}} - 2^\Delta \cdot 2^{\gamma-n} \cdot 2^{2^{n-\gamma-\Delta}}$$

$$mk \cdot 2^{2\gamma - k - n} = \{k(\ln 2) - 2^{\gamma + \Delta - n}\} \cdot 2^{2^{n-\gamma-\Delta}}$$

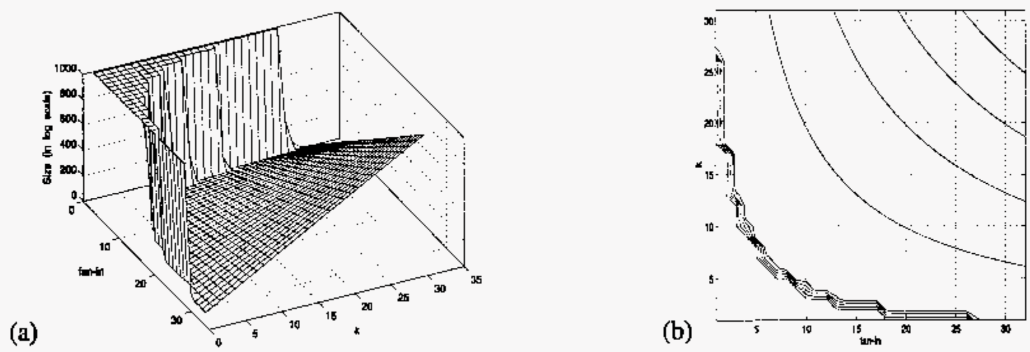

Figure 1: (a) Exact *size* as a function of the *fan-in* Δ and k, for $n = 64$ and $m = 1$; (b) contour plot.

$$(m/\ln 2)\cdot 2^{2\gamma-k-n} = \left\{1 - 2^{\gamma+\Delta-n}/(k\ln 2)\right\}\cdot 2^{2^{n-\gamma-\Delta}}$$

which—by neglecting $2^{\gamma+\Delta}/\{k(\ln 2)\cdot 2^{n}\}$—gives:

$$\log\beta + 2\gamma - k - n \quad = 2^{n-\gamma-\Delta}$$

i.e., the same equation as (11).

These show that the critical points are situated in the (close) vicinity of the parabola $k\Delta \approx n - \log(n + \log m)$. The fact that they are relative minimum has also been proven [7]. ❑

The *size* has been computed for many different values of n, m, Δ and k. One example of those extensive simulations is plotted in Figure 1. From Figure 1(a) it may seem that k and Δ have almost the same influence on $size_{BFs}$. The discrete parabola-like curves (the one closer to the axes is approximately $k\Delta \approx n - \log(n + \log m)$) can be seen in Figure 1(b).

Proposition 5 (this paper) *The absolute minimum* $size_{BFs}$ *is obtained for fan-in* $\Delta = 2$.

Sketch of proof We will analyse only the critical points by using the approximation $k\Delta \approx n - \log n$. Intuitively the claim can be understood if we replace this value in (10):

$$\begin{aligned} size^{*}_{BFs} &\approx m\cdot 2^{n-\log n-k} + 2^{\Delta}\cdot 2^{2^{n-n+\log n-\Delta}} \\ &< m\cdot 2^{n-\log n} + 2^{\Delta}\cdot 2^{2^{\log n}} \\ &= m\cdot 2^{n}/n + 2^{\Delta}\cdot 2^{n}, \end{aligned}$$

which clearly is minimised for $\Delta = 2$. ❑

The detailed proof relies on computing the *size* given by equation (10) $size_{BFs}(n, m, k, \Delta)$ for $k \approx (n - \log n)/\Delta$, and then showing that:

$$size^{*}_{BFs}(n, m, \Delta+1) - size^{*}_{BFs}(n, m, \Delta) > 0.$$

Hence, the function is monotonically increasing and the minimum is obtained for the smallest *fan-in* $\Delta = 2$. Because the proof has been obtained using successive approximations, several simulation results are presented in Table 1. It can be seen that while for relatively small n the *size*-optimal solutions are obtained even for $\Delta = 16$, starting from $n \ge 64$ all the *size*-optimal solutions are obtained for $\Delta = 2$. The other relative minima (on, or in the vicinity of the parabola $k\Delta \approx n - \log n$) are only slightly larger than the absolute minimum. They might be of *practical interest* as leading to networks having fewer layers: $n/\log\Delta$ instead of n. Last, but not least, it is to be mentioned that all these relative minimum are obtained for *fan-ins* strictly lower that linear (as $\Delta \le n - \log n$).

A similar result can be obtained for $\mathbb{F}_{n,m}$, as the first layer is represented by COMPARISONs (*i.e.*, $\mathbb{F}_{n,1}$) which can be decomposed to satisfy the limited *fan-in* condition [4].

Proposition 6 (Lemma 1 from [3]) *The* COMPARISON *of two n-bit numbers can be computed by a* Δ*-ary tree neural network of perceptrons having integer weights and thresholds bounded by* $2^{\Delta/2}$ *for any fan-in in the range* $3 \le \Delta \le n$.

The *size* complexity of the NN implementing one $\mathbb{F}_{n,m}$ function is [3]:

Table 1: Minimum $size_{BFs}$ for $m = 1$ and different values of n.

n	$8 = 2^3$	$16 = 2^4$	$32 = 2^5$	$64 = 2^6$	$128 = 2^7$	$256 = 2^8$	$512 = 2^9$	$1024 = 2^{10}$	$2048 = 2^{11}$
size	110	1470	349,530	1.611×10^{9}	6.917×10^{18}	5.104×10^{38}	2.171×10^{76}	1.005×10^{154}	1.685×10^{307}
Δ	**4**	**8**	**16**	**2**	**2**	**2**	**2**	**2**	**2**
$k\Delta$	4	8	16	58	122	248	504	1014	2038

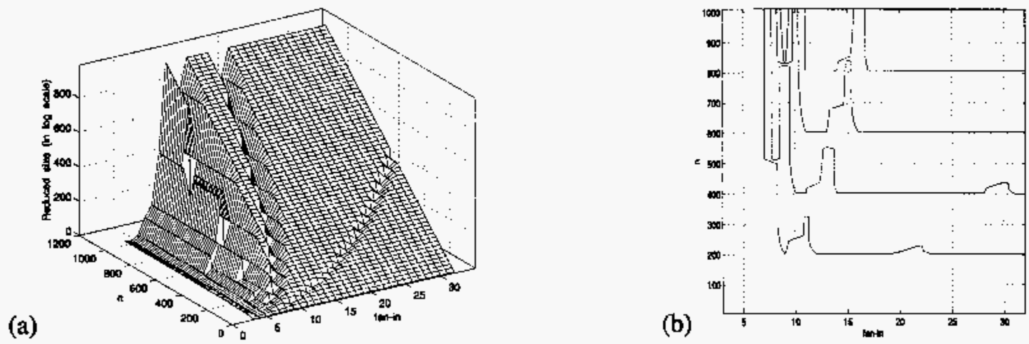

Figure 2: (a) *Size* of NNs implementing $IF_{n,m}$ for $m = 2^{0.99n}$ (almost completely specified BFs); (b) contour plot.

$$size_{IF} = 2nm \cdot \left\{ \frac{1}{\Delta/2} + \ldots + \frac{1}{(\Delta/2)^{depth_{IF}}} \right\}, \quad (13)$$

where $depth_{IF} = \lceil \log n / (\log\Delta - 1) \rceil$. A substantial enhancement is obtained if the *fan-in* is limited. The maximum number of *different BFs* which can be computed in each layer is:

$$(2n/\Delta)\, 2^{\Delta}, \quad \frac{2n/\Delta}{\Delta/2}\, 2^{\Delta(\Delta/2)}, \ldots, \quad \frac{2n/\Delta}{(\Delta/2)^{depth_{IF}-1}} \cdot 2^{\Delta(\Delta/2)^{depth_{IF}-1}}. \quad (14)$$

For large m (needed for achieving a certain precision [10]), and/or large n, the first terms of the sum (13) will be larger than the equivalent ones from (14). This is equivalent to the trick from [12], as the lower levels will compute *all the possible functions* using only limited *fan-in* COMPARISONs. Hence, the optimum *size* becomes:

$$size^{*}_{IF} = 2n \cdot \left\{ \sum_{i=1}^{k} \frac{2^{\Delta(\Delta/2)^{i-1}}}{\Delta(\Delta/2)^{i-1}} + \sum_{i=k+1}^{depth_{IF}} \frac{m}{(\Delta/2)^{i}} \right\}.$$

Following similar steps to the ones used in *Proposition 5*, it is possible to show that the minimum *size* is obtained for $\Delta = 3$. To get a better understanding, we have done simulations by considering that $m = 2^{\varepsilon n}$. Some results can be seen in Figure 2 (for $\varepsilon = 0.99$).

We mention here that similar results ($\Delta = 6\ldots9$), based on closer estimates of *area* and *delay* have been proved for VLSI-efficient implementations of $IF_{n,m}$ functions [5, 6]. Different complexity estimates for COMPARISON can be seen in Table 2. All of these support the claim that small constant *fan-in* NNs can be *size*- and VLSI-optimal, while there are similar small constants relating to our capacity of processing information [16].

4: Conclusions and open problems

In this paper, we have extended a result from Horne & Hush [12] valid for *fan-in* $\Delta = 2$ to arbitrary *fan-ins*, and have shown that the minimum *size* is obtained for small (constant) *fan-ins*. We have also shown that, using their construction, it is possible to obtain 'good' (*i.e.*, relative minimum) solutions for *fan-ins* strictly lower than linear. The same results have been obtained for the *size*-optimal solution from [18]. The main conclusions are that: (*i*) there are interesting *fan-in* dependent *depth-size* (and *area-delay*) tradeoffs; and (*ii*) there are optimal solutions having small constant *fan-in* values. Future work will concentrate on linking these results with the entropy of the data-set, and with principles like "Occam's razor" [27], and "minimum description length", as well as trying to find closer estimates for mixed analog/digital implementations.

References

[1] Abu-Mostafa, Y.S. Connectivity Versus Entropy. In D.Z. Anderson (ed.), *NIPS*, New York, NY: AIP, 1-8, 1988.

[2] Bartlett, P.L. 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. Sys. Eng., Australian Natl. Univ., Canberra, 1996 [short version in M.C. Mozer, M.I. Jordan & T. Petsche (eds.): *NIPS 9*, Cambridge, MA: MIT Press, 1997].

[3] Beiu, V., Peperstraete, J.A., Vandewalle, J. & Lauwereins, R. Area-Time Performances of Some Neural Computations. In P. Borne *et al.* (eds.): *SPRANN'94*, Lille, France: GERF EC, 664-668, 1994.

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

[5] Beiu, V. On the Circuit and VLSI Complexity of Threshold Gate COMPARISON. *Tech. Rep. LA-UR-96-3591*, Los Alamos Natl. Lab., USA, 1996 [to appear in *Neurocomputing*, 1997/8].

[6] Beiu V. Constant Fan-In Digital Neural Networks Are VLSI-Optimal. *Tech. Rep. LA-UR-97-61*, Los Alamos Natl. Lab., USA, 1997; Chapter 12 in S.W. Ellacott, J.C. Mason

Table 2: Different estimates of AT^2 for SRK [21], B_4 and B_log [3, 5], ROS [19], and VCB [23].

Area \ Delay	*Depth*	*Fan-in*	*Length*
Size	$AT^2_{VCB} = O(\sqrt{n})$	$AT^2_{B\_4} = O(n \log^2 n)$	$AT^2_{VCB} = O(n^2 \sqrt{n})$
	$AT^2_{ROS} = O(n/\log n)$	$AT^2_{B\_log} = O[n \log^3 n / \log^2(\log n)]$	$AT^2_{ROS} \cong 3 \cdot n^3/\log n$
	$AT^2_{SRK} = O(n)$	$AT^2_{VCB} = O(n \sqrt{n})$	$AT^2_{B\_log} \cong 4 \cdot n^3/\log n$
	$AT^2_{B\_log} = O[n \log n / \log^2(\log n)]$	$AT^2_{ROS} = O(n^3/\log^3 n)$	$AT^2_{B\_4} \cong 4 \cdot n^3$
	$AT^2_{B\_4} = O(n \log^2 n)$	$AT^2_{SRK} = O(n^3)$	$AT^2_{SRK} \cong 27 n^3/4$
Σ_{NN} *fan-ins*	$AT^2_{VCB} = O(n)$	$AT^2_{B\_4} = O(n \log^2 n)$	$AT^2_{B\_log} \cong 4 n^3$
	$AT^2_{B\_log} = O[n \log^2 n / \log^2(\log n)]$	$AT^2_{B\_log} = O[n \log^4 n / \log^2(\log n)]$	$AT^2_{VCB} \cong 4 n^3$
	$AT^2_{B\_4} = O(n \log^2 n)$	$AT^2_{VCB} = O(n^2)$	$AT^2_{B\_4} \cong 5 n^3$
	$AT^2_{ROS} = O(n^2/\log^2 n)$	$AT^2_{ROS} = O(n^4/\log^4 n)$	$AT^2_{ROS} = O(n^4/\log^2 n)$
	$AT^2_{SRK} = O(n^2)$	$AT^2_{SRK} = O(n^4)$	$AT^2_{SRK} = O(n^4)$
$\Sigma_{NN}(\Sigma_i \lvert w_i \rvert + \lvert \theta \rvert)$	$AT^2_{B\_4} = O(n \log^2 n)$	$AT^2_{B\_4} = O(n \log^2 n)$	$AT^2_{B\_4} = O(n^3)$
	$AT^2_{B\_log} = O[n\sqrt{n} \log n / \log^2(\log n)]$	$AT^2_{B\_log} = O[n\sqrt{n} \log^3 n / \log^2(\log n)]$	$AT^2_{B\_log} = O(n^3 \sqrt{n}/\log n)$
	$AT^2_{ROS} = O(n^2/\log n)$	$AT^2_{ROS} = O(n^4/\log^3 n)$	$AT^2_{ROS} = O(n^4/\log n)$
	$AT^2_{SRK} = O(n^2)$	$AT^2_{SRK} = O(n^4)$	$AT^2_{SRK} = O(n^4)$
	$AT^2_{VCB} = O(n^{1/2} \cdot 2^{\sqrt{n}})$	$AT^2_{VCB} = O(n^{3/2} \cdot 2^{\sqrt{n}})$	$AT^2_{VCB} = O(n^{5/2} \cdot 2^{\sqrt{n}})$

& I.J. Anderson (eds.): *Mathematics of Neural Networks: Models, Algorithms and Applications*, Boston, MA: Kluwer Academic, 89-94, 1997.

[7] Beiu, V. When Constants Are Important. *Tech. Rep. LA-UR-97-226*, Los Alamos Natl. Lab., USA, 1997; in I. Dumitrache (ed.): *Proc. Ctrl. Sys. & Comp. Sci. CSCS-11*, UPBucharest, vol. **2**, 106-111, 1997.

[8] Beiu, V. *VLSI Complexity of Discrete Neural Networks*. Newark, NJ: Gordon & Breach, 1998.

[9] Bruck, J. & Goodmann, J.W. On the Power of Neural Networks for Solving Hard Problems. In D.Z. Anderson (ed.), *NIPS*, New York, NY: AIP, 137-143, 1988 [also in *J. Complexity*, **6**, 129-135, 1990].

[10] Denker, J.S. & Wittner, B.S. Network Generality, Training Required and Precision Required. In D.Z. Anderson (ed.), *NIPS*, New York, NY: AIP, 219-222, 1988.

[11] Hammerstrom, D. The Connectivity Analysis of Simple Association –or– How Many Connections Do You Need. In D.Z. Anderson (ed.), *NIPS*, New York, NY: AIP, 338-347, 1988.

[12] Horne, B.G. & Hush, D.R. On the Node Complexity of Neural Networks. *Neural Networks*, **7**(9), 1413-1426, 1994.

[13] Klaggers, H. & Soegtrop M. Limited Fan-In Random Wired Cascade-Correlation. In D.J. Myers & A.F. Murray (eds.), *MicroNeuro'93*, Edinburgh, UK: UnivEd Tech. Ltd., 79-82, 1993.

[14] Krishnamoorthy, A.V., Paturi, R., Blume, M., Linden, G.D., Linden, L.H. & Esener, S.C. Hardware Tradeoffs for Boolean Concept Learning. *Proc. WCNN'94*, Hills-dale: Lawrence Erlbaum & INNS Press, 551-559, 1994.

[15] Mason, R.D. & Robertson, W. Mapping Hierarchical Neural Networks to VLSI Hardware. *Neural Networks*, **8**(6), 905-913, 1995.

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

[17] Phatak, D.S. & Koren, I. Connectivity and Performances Tradeoffs in the Cascade Correlation Learning Architecture. *IEEE Trans. Neural Networks*, **5**(6), 930-935, 1994.

[18] Red'kin, N.P. Synthesis of Threshold Circuits for Certain Classes of Boolean Functions. *Kibernetika*, **5**, 6-9, 1970 [English translation in *Cybernetics*, **6**(5), 540-544, 1973].

[19] Roychowdhury, V.P., Orlitsky, A. & Siu, K.-Y. Lower Bounds on Threshold and Related Circuits Via Communication Complexity. *IEEE Trans. Information Theory*, **40**(2), 467-474, 1994.

[20] Shannon C. The Synthesis of Two-Terminal Switching Circuits. *Bell Sys. Tech. J.*, **28**, 59-98, 1949.

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

[22] Ullman, J.D. *Computational Aspects of VLSI*. Rockville, MA: Comp. Sci. Press, 1984.

[23] Vassiliadis, S., Cotofana, S. & Berteles, K. 2-1 Addition and Related Arithmetic Operations with Threshold Logic. *IEEE Trans. Comp.*, **45**(9), 1062-1068, 1996.

[24] Walker, M.R., Haghighi, S., Afghan, A. & Akers, L.A. Training a Limited-Interconnect, Synthetic Neural IC. In D.S. Touretzky (ed.), *NIPS 1*, San Mateo, CA: Morgan Kaufmann, 777-784, 1989.

[25] Williamson, R.C. ε-Entropy and the Complexity of Feedforward Neural Networks. In R.P. Lippmann, J.E. Moody and D.S. Touretzky (eds.), *NIPS 3*, San Mateo, CA: Morgan Kaufmann, 946-952, 1990.

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

[27] Zhang, B.-T. & Mühlenbein, H. Genetic Programming of Minimal Neural Networks Using Occam's Razor. *Tech. Rep. GMD 0734*, Schloß Birlinghoven, St. Augustin, Germany, 1993 [also in *Complex Systems*, **7**(3), 199-220, 1993].

Report Number (14) LA-UR--97-3493
CONF-980538--

Publ. Date (11) 199709
Sponsor Code (18) DOE/HR, XF
UC Category (19) UC-905, DOE/ER

DOE