Neural Nets with Superlinear VC-Dimension

(Extended Abstract)

Wolfgang Maass
Institute for Theoretical Computer Science, Technische Universitaet Graz
Klosterwiesgasse 32/2, A-8010 Graz, Austria; e-mail: maass@igi.tu-graz.ac.at

The Vapnik-Chervonenkis dimension VC-dimension($\mathcal{N}$) of a neural net $\mathcal{N}$ with n input nodes is defined as the size of the largest set $S \subseteq \mathbf{R}^n$ which is "shattered" by $\mathcal{N}$ in the sense that every function $F : S \to \{0,1\}$ can be computed by $\mathcal{N}$ with some assignment of real numbers to its weights.

The VC-dimension of a neural net $\mathcal{N}$ is an important measure for the expressiveness for $\mathcal{N}$, i.e. for the variety of functions that can be computed by $\mathcal{N}$ with different choices for its weights. In particular it has been shown in Blumer et al. (1987) and Ehrenfeucht et al. (1989) that the VC-dimension of $\mathcal{N}$ essentially determines the number of training examples that are needed to train $\mathcal{N}$ in Valiant's model (Valiant, 1984) for probably approximately correct learning ("PAC-learning").

It has been known for quite a while that the VC-dimension of a neural net with linear threshold gates and w edges (respectively w weights) is at most $O(w \cdot \log w)$. This result, which holds for arbitrary real valued input patterns, was first shown by Cover (1964 and 1968) and later by Baum and Haussler (1989). It has frequently been conjectured that the "true" upper bound is $O(w)$. This conjecture is quite plausible, since a single linear threshold gate with w edges has VC-dimension $w+1$. Furthermore it is hard to imagine that the VC-dimension of a network of linear threshold gates can be larger than the sum of the VC-dimensions of the individual linear threshold gates in the network.

We disprove this popular conjecture by showing that for any depth $d \geq 3$ quite a number of neural nets $\mathcal{N}$ of depth d have a VC-dimension that is superlinear in the number w of edges in $\mathcal{N}$. In particular, we exhibit for arbitrarily large $w \in \mathbf{N}$ neural nets $\mathcal{N}$ of depth 3 (i.e. with 2 hidden layers) with w weights that have VC-dimension $\Omega(w \cdot \log w)$. This shows that the quoted upper bound of $O(w \log w)$ is in fact asymptotically optimal. Our lower bound also shows that the well-known upper bound $2w \log(eN)$ for the VC-dimension of a neural net with w weights and N computation nodes (due to Baum and Haussler, 1989) is asymptotically optimal.

The result of this paper may also be viewed as mathematical evidence for a certain type of "connectionism thesis": that a network of neuron-like elements

M. Marinaro et al. (eds.), *ICANN '94*

is more than just the sum of its elements. We show that in a large neural net a single edge may add more than a constant to the VC-dimension of the neural net: its contribution may increase with the logarithm of the total size of the neural net.The proof of this result relies on a new method that allows us to encode more "program-bits" in the weights of a neural net than previously thought possible. The same result can also be derived for neural nets with other activation functions such as $\sigma(y) = \frac{1}{1+e^{-y}}$.

The proof of our result employs classical circuit construction methods due to Neciporuk (1964) and Lupanov (1972). We refer to Maass (1993) for details of the proofs. Bartlett (1993) has independently derived lower bounds for the VC-dimensions of various neural nets of depth 2 and 3 that are *linear* in the number w of weights.

The *neural nets* which are considered in this paper are feedforward neural nets with linear threshold gates (or simpler: threshold gates), i.e. gates which apply the heaviside activation function to the weighted sum $\sum_{i=1}^{m} \alpha_i y_i + \alpha_0$ of their inputs $y_1, \ldots, y_m$. The parameters $\alpha_1, \ldots, \alpha_m$ and α_0 are the *weights* of such gate. We will consider in this paper only neural nets with boolean inputs and one boolean output.

The *depth* of a neural net is the length of the longest path from an input node to the output node (= output gate). The depth of a gate in a neural net is the length of the longest path from an input node to that gate. We refer to all gates of depth d as "level d" of a neural net.

We will focus our attention on neural nets that are *layered* in the sense that only gates on successive levels are connected by an edge, and input nodes are connected by an edge only with gates on level 1. This is not a serious restriction, since for $i+1 < j$ one may replace an edge between nodes on levels i and j by a path of length $j-i$ (by introducing $j-i-1$ "dummy" gates on the intermediate levels). It is obvious that a layered neural net of depth d has exactly $d-1$ *hidden layers.* One calls a layered neural net *fully connected* if any two nodes on successive levels (as well as any input nodes and any gates on level 1) are connected by an edge.

We use the standard notation $f = \Theta(g)$ for arbitrary functions $f, g : \mathbf{N} \to \mathbf{N}$ to indicate that both $f = O(g)$ and $f = \Omega(g)$.

Theorem 1: *Assume that $(\mathcal{N}_n)_{n \in N}$ is any sequence of fully connected layered neural nets of depth $d \geq 3$. Furthermore assume that $\mathcal{N}_n$ has n input nodes and $\Theta(n)$ gates, of which $\Omega(n)$ gates are on the first hidden layer, and at least $4 \log n$ gates are on the second hidden layer of $\mathcal{N}_n$.*

Then $\mathcal{N}_n$ has $\Theta(n^2)$ edges and VC-dimension$(\mathcal{N}_n) = \Theta(n^2 \log n)$.

The *proof* of Theorem 1 proceeds by "embedding" into the given neural nets $\mathcal{N}_n$ of Theorem 1 for suitable $\tilde{n} \leq n$ the special neural nets $\mathcal{M}_{\tilde{n}}$ that are constructed in the following Theorem 2.

Theorem 2: *Assume that n is some arbitrary power of 2. Then one can construct a neural net $\mathcal{M}_n$ of depth 3 with n input nodes and at most $17n^2$ edges such that VC-dimension$(\mathcal{M}_n) \geq n^2 \cdot \log n$.*

Sketch of the Proof of Theorem 2: We assume that n is of the form $2^{n'}$ for some non-zero $n' \in \mathbf{N}$. This implies that n is even and that $\log n \in \mathbf{N}$. We construct a neural net $\mathcal{M}_n$ with $2n + \log n$ binary inputs and $O(n^2)$ weights that shatters the following set $S \subseteq \{0,1\}^{2n+\log n}$ of size $n^2 \cdot \log n$:

$$S := \{\underline{e}_p \underline{e}_q \underline{\tilde{e}}_m : p, q \in \{1, \ldots, n\} \text{ and } m \in \{1, \ldots, \log n\}\},$$

where $\underline{e}_r \in \{0,1\}^n$ denotes the r-th unit vector of length n ($r = 1, \ldots, n$), and $\underline{\tilde{e}}_m \in \{0,1\}^{\log n}$ denotes the m-th unit vector of length $\log n$ ($m = 1, \ldots, \log n$). Thus each $\underline{u} \in S$ contains exactly three 1's.

Fix any map $F : S \to \{0,1\}$. One can encode F by a function $g : \{\underline{e}_1, \ldots, \underline{e}_n\}^2 \to \{0,1\}^{\log n}$ where the m-th output bit of $g(\underline{e}_p, \underline{e}_q)$ equals 1 if and only if $F(\underline{e}_p \underline{e}_q \underline{\tilde{e}}_m) = 1$. One can show that for any function $g : \{\underline{e}_1, \ldots, \underline{e}_n\}^2 \to \{0,1\}^{\log n}$ there exist for $k = 1, \ldots, 4$ functions $g_k : \{\underline{e}_1, \ldots, \underline{e}_n\}^2 \to \{0,1\}^{\log n}$ such that $g_k(\cdot, \underline{e}_q)$ is $1-1$ for every fixed $q \in \{1, \ldots, n\}$, and such that for all $p, q \in \{1, \ldots, n\}$:

$$g(\underline{e}_p, \underline{e}_q) = \begin{cases} g_1(\underline{e}_p, \underline{e}_q) \oplus g_2(\underline{e}_p, \underline{e}_q), & \text{if } p \leq \frac{n}{2} \\ g_3(\underline{e}_p, \underline{e}_q) \oplus g_4(\underline{e}_p, \underline{e}_q), & \text{if } p > \frac{n}{2}. \end{cases}$$

The symbol $\oplus$ denotes here the bit-wise exclusive OR (i.e. parity) on bit-strings of length $\log n$. In the following we write $(\underline{x})_j$ for the j-th coordinate of some vector $\underline{x}$, and $\mathrm{bin}(i)$ for the binary string that represents the number $i-1$.

The neural net $\mathcal{M}_n$ computes F in the following way. The output gate on level 3 is an OR of $4 \log n$ threshold gates. These threshold gates consist of $\log n$ blocks of 4 threshold gates, such that for any $b \in \{1, \ldots, \log n\}$ some threshold gate in the b-th block outputs 1 for network input $\underline{e}_p \underline{e}_q \underline{\tilde{e}}_m$ if and only if $m = b$ and $(g(\underline{e}_p, \underline{e}_q))_b = 1$ (i.e. $F(\underline{e}_p \underline{e}_q \underline{\tilde{e}}_m) = 1$). More precisely, the a-th threshold gate in block b outputs 1 if and only if the a-th one of the following 4 conditions is satisfied:

$$\begin{array}{llllllll}
(1) & m = b & \wedge & p \leq \frac{n}{2} & \wedge & (g_1(\underline{e}_p, \underline{e}_q))_b = 1 & \wedge & (g_2(\underline{e}_p, \underline{e}_q))_b = 0 \\
(2) & m = b & \wedge & p \leq \frac{n}{2} & \wedge & (g_1(\underline{e}_p, \underline{e}_q))_b = 0 & \wedge & (g_2(\underline{e}_p, \underline{e}_q))_b = 1 \\
(3) & m = b & \wedge & p > \frac{n}{2} & \wedge & (g_3(\underline{e}_p, \underline{e}_q))_b = 1 & \wedge & (g_4(\underline{e}_p, \underline{e}_q))_b = 0 \\
(4) & m = b & \wedge & p > \frac{n}{2} & \wedge & (g_3(\underline{e}_p, \underline{e}_q))_b = 0 & \wedge & (g_4(\underline{e}_p, \underline{e}_q))_b = 1.
\end{array}$$

The subconditions involving $g_1, \ldots, g_4$ are tested with the help of $8n$ threshold gates on level 1 that use weights $w_{k,i,j} \in \{1, \ldots, n\}$, which are defined by the condition $w_{k,i,j} = r \Leftrightarrow g_k(\underline{e}_r, \underline{e}_j) = \mathrm{bin}(i)$. There are $4n$ threshold gates $G^+_{k,i}(\underline{e}_p, \underline{e}_q)$ on level 1 that output 1 if and only if $\sum_{r=1}^{n} r \cdot (\underline{e}_p)_r \geq \sum_{j=1}^{n} w_{k,i,j} \cdot (\underline{e}_q)_j$ (i.e. $p \geq w_{k,i,q}$), and $4n$ threshold gates $G^-_{k,i}(\underline{e}_p, \underline{e}_q)$ on level 1 that output 1 if and only if $\sum_{r=1}^{n} r \cdot (\underline{e}_p)_r \leq \sum_{j=1}^{n} w_{k,i,j} \cdot (\underline{e}_q)_j$ (i.e. $p \leq w_{k,i,q}$); for $k = 1, \ldots, 4$ and $i = 1, \ldots, n$. These are the only weights in the neural net $\mathcal{M}_n$ which depend on the function $F : S \to \{0,1\}$. By definition one has that for each k, i at least one of the two gates $G^+_{k,i}(\underline{e}_p, \underline{e}_q)$, $G^-_{k,i}(\underline{e}_p, \underline{e}_q)$ outputs 1. Furthermore for any $k \in \{1, \ldots, 4\}$ and any $\underline{e}_p \underline{e}_q \underline{\tilde{e}}_m \in S$ there is exactly one $i \in \{1, \ldots, n\}$ such that both of these gates output 1. This index i is characterized by the equality $g_k(\underline{e}_p, \underline{e}_q) = \mathrm{bin}(i)$. Hence one can check whether $(g_k(\underline{e}_p, \underline{e}_q))_b = 1$ by testing

whether $\sum_{\substack{i=1,\ldots,n \\ \text{with } (\text{bin}(i))_b=1}} G^+_{k,i}(\underline{e}_p, \underline{e}_q) + G^-_{k,i}(\underline{e}_p, \underline{e}_q) \geq \frac{n}{2} + 1$, and one can check whether $(g_k(\underline{e}_p, \underline{e}_q))_b = 0$ by testing whether $\sum_{\substack{i=1,\ldots,n \\ \text{with } (\text{bin}(i))_b=0}} G^+_{k,i}(\underline{e}_p, \underline{e}_q) + G^-_{k,i}(\underline{e}_p, \underline{e}_q) \geq \frac{n}{2} + 1$. Furthermore the sums on the left hand side of both inequalities can only assume the values $\frac{n}{2}$ or $\frac{n}{2} + 1$. Therefore one can test the AND of two subconditions of this type and of the subconditions "$m = b$" and "$p \leq \frac{n}{2}$" ("$p > \frac{n}{2}$") by a *single* threshold gate on level 2 of $\mathcal{M}_n$. Hence one can test each of the conditions (1), ..., (4) by a separate threshold gate in the b-th block on level 2.

Altogether the constructed layered neural net $\mathcal{M}_n$ consists of $2n+\log n$ input nodes, $8n + 8\log n + 3$ computation nodes, and $16n^2 + (8\log n + 2)n + 16\log n$ edges. Obviously the nodes and edges of $\mathcal{M}_n$ are independent of the given function $F : S \to \{0, 1\}$, whereas the weights on $8n^2$ of the edges depend on F (these weights range over $\{1, \ldots, n\}$).

Since the function $F : S \to \{0, 1\}$ that is computed by $\mathcal{M}_n$ was choosen arbitrarily, the construction implies that the set S is shattered by $\mathcal{M}_n$. Hence VC-dimension$(\mathcal{M}_n) \geq |S| = n^2 \cdot \log n$. We refer to (Maass, 1993) for details of this proof. ∎

References

Bartlett, P. L. (1993). *Lower bounds on the Vapnik-Chervonenkis dimension of multi-layer threshold networks*, Proc. of the 6th Annual ACM Conference on Computational Learning Theory, 144 - 150

Baum, E. B., Haussler, D. (1989). *What size net gives valid generalization?*, Neural Computation, **1**, 151 - 160

Blumer, A., Ehrenfeucht, A., Haussler, D., Warmuth, M. K. (1989). *Learnability and the Vapnik-Chervonenkis dimension*, J. of the ACM, **36**(4), 929 - 965

Cover,T. M. (1964). *Geometrical and statistical properties of linear threshold devices*, Stanford PH. D. Thesis 1964, Stanford SEL Technical Report No. 6107-1, May 1964

Cover,T. M. (1968). *Capacity problems for linear machines*, in: Pattern Recognition, L. Kanal ed., Thompson Book Co., 283 - 289

Ehrenfeucht, A., Haussler, D., Kearns, M., Valiant, L. (1989). *A general lower bound on the number of examples needed for learning*, Information and Computation, **82**, 247 - 261

Lupanov, O. B. (1972). *On circuits of threshold elements*, Dokl. Akad. Nauk SSSR, **202**, 1288 - 1291; engl. transl. in: Sov. Phys. Dokl., **17**, 91 - 93

Maass W. (1993). *Neural nets with superlinear VC-dimension*, IIG-Report 366 of the Technische Universität Graz, June 1993

Neciporuk, E. I. (1964). *The synthesis of networks from threshold elements*, Probl. Kibern. No. 11, 49 - 62; engl. transl. in: Autom. Expr., **7**(1), 35 - 39

Valiant, L. G. (1984). *A theory of the learnable*, Comm. of the ACM,**27**, 1134 - 1142