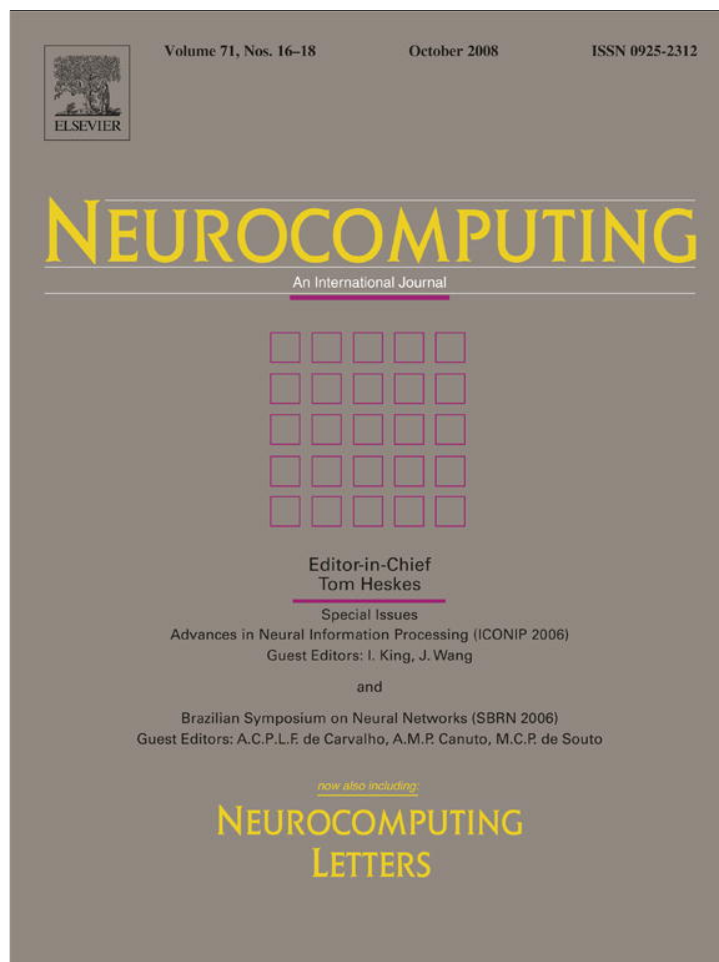


Provided for non-commercial research and education use.
Not for reproduction, distribution or commercial use.



This article appeared in a journal published by Elsevier. The attached copy is furnished to the author for internal non-commercial research and education use, including for instruction at the authors institution and sharing with colleagues.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

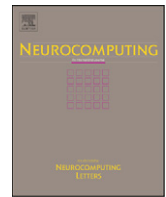
In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier's archiving and manuscript policies are encouraged to visit:

<http://www.elsevier.com/copyright>



Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Letters

Upper bound on pattern storage in feedforward networks

Pramod L. Narasimha^{a,*}, Michael T. Manry^b, Francisco Maldonado^c^a FastVDO Inc., 5840 Banneker Road, Suite 270, Columbia, MD 21044, USA^b Department of Electrical Engineering, University of Texas at Arlington, Arlington, TX 76013, USA^c Williams Pyro, Inc., 200 Greenleaf Street, Fort Worth, TX 76107, USA

ARTICLE INFO

Article history:

Received 14 November 2007

Received in revised form

26 March 2008

Accepted 28 April 2008

Communicated by G.-B. Huang

Available online 17 May 2008

Keywords:

Pattern storage

Memorization

Upper bound

ABSTRACT

An upper bound on pattern storage is stated for nonlinear feedforward networks with analytic activation functions, like the multilayer perceptron and radial basis function network. The bound is given in terms of the number of network weights, and applies to networks having any number of output nodes and arbitrary connectivity. Starting from the strict interpolation equations and exact finite degree polynomial models for the hidden units, a straightforward proof by contradiction is developed for the upper bound. Several networks, trained by conjugate gradient, are used to demonstrate the tightness of the bound for random patterns.

© 2008 Elsevier B.V. All rights reserved.

1. Introduction

Pattern memorization in nonlinear networks has been studied for many decades. The number of patterns that can be memorized has been referred to as the information capacity [1] and storage capacity [19]. Equating network outputs to desired outputs has been referred to as strict interpolation [5,23,9]. It is important to understand the pattern memorization capability of feedforward networks for at least two reasons. First, the capability to memorize is related to the ability to form arbitrary finite shapes in weight space. Second, if a network can successfully memorize many random patterns, we know that the training algorithm is powerful [17].

Upper bounds on the number of distinct patterns P that can be memorized by nonlinear feedforward networks are functions of the number of weights in the network, N_w , and the number of outputs, M [22]. For example, Davis [5] has shown that for any P distinct, complex points there exists a unique $(P - 1)$ degree polynomial, with complex coefficients, that strictly interpolates (memorizes) all the points. In other words, breaking up the complex quantities into separate real and imaginary parts, he has derived a bound for the $M = 2$ case. An upper bound on the number of hidden units in the multilayer perceptron (MLP) for the $M = 1$ case, derived by Elisseeff and Paugam-Moisy [6], agrees with the bound of Davis. They [7] subsequently extended their

result to the case of M larger than one. Suyari and Matsuba [24] have derived the storage capacity of neural networks with binary weights, using minimum distance between the patterns. Cosnard et al. [4] have derived upper and lower bound on the size of nets capable of computing arbitrary dichotomies. Ji and Psaltis [14] have derived upper and lower bounds for the information capacity of two-layer feedforward neural networks with binary interconnections, using an approach similar to that of Baum [2]. Moussaoui and Abbassi [20] and Ma and Ji [18] have pointed out that the information capacity is reflected in the number of weights of the network. Other works by Huang et al. [10–12] give the upper bound on the number of hidden units which comes from the lower bound on memorization.

Unfortunately, most recent research on pattern memorization in feedforward networks focuses on the one output case and does not apply to feedforward networks with arbitrary connectivity between network units. In this paper, partially building upon the work of Davis [5], we investigate an upper bound for $M \geq 1$ and arbitrary hidden unit activation functions. In Section 2, we introduce our notation. A straightforward proof of the upper bound is given in Section 3. Examples which indicate the validity of the bound are presented in Section 4.

2. Notation

Let $\{\mathbf{x}_p, \mathbf{t}_p\}_{p=1}^P$ be the dataset where $\mathbf{x}_p \in \mathbb{R}^N$ is the input vector and $\mathbf{t}_p \in \mathbb{R}^M$ is the desired output vector and P is the number of patterns. The patterns are unique in that none of the input vectors

* Corresponding author.

E-mail addresses: pramod@fastvdo.com, L_pramod79@yahoo.com (P.L. Narasimha).

are the same. Let us consider a feedforward MLP, having N inputs, one hidden layer with h nonlinear units and an output layer with M linear units. For the p th pattern, the j th hidden unit's net function and activation are, respectively,

$$net_{pj} = \sum_{n=1}^{N+1} w_h(j, n) \cdot x_{pn}, \quad 1 \leq p \leq P, \quad 1 \leq j \leq h \quad (1)$$

$$O_{pj} = f(net_{pj}) \quad (2)$$

Here, the activation $f(net)$ is a nonlinear function of the net function. The weight $w_h(j, n)$ connects the n th input to the j th hidden unit. Here the threshold of the j th node is represented by $w_h(j, N + 1)$ and is handled by fixing $x_{p, N+1}$ to one. The i th output for the p th pattern is given by

$$y_{pi} = \sum_{k=1}^{N+1} w_{oi}(i, k) \cdot x_{pk} + \sum_{j=1}^h w_{oh}(i, j) \cdot O_{pj} \quad (3)$$

For the p th pattern, the N input values are x_{pk} ($1 \leq k \leq N$) and the M desired output values are t_{pi} ($1 \leq i \leq M$). w_{oi} denotes weights connecting inputs to outputs and w_{oh} denotes weights connecting hidden units to outputs.

3. An upper bound

In this section, we describe an exact finite degree polynomial model for the memorization process. Then an upper bound on memorization and a proof by contradiction are presented.

3.1. Modeling of memorization

Let us assume that the net functions for different finite inputs can be different but finite. Equating y_{pi} in (3) to t_{pi} , we get

$$t_{pi} = \sum_{k=1}^{N+1} w_{oi}(i, k) \cdot x_{pk} + \sum_{j=1}^h w_{oh}(i, j) \cdot O_{pj} \quad (4)$$

Suppose that a random input vector is put through the network of (3), and that the outputs are called t_{pi} , as in (4). We have then generated a pattern that is memorized by the network. However, the desired outputs are not arbitrarily chosen. We call such patterns trivially memorized (TM) patterns. Clearly, every network has an infinite number of TM patterns. In order to model the memorization process in (4), we can use an exact, finite degree polynomial model for O_{pj} . This is accomplished by using Lagrange polynomials [13]. The $P - 1$ degree Lagrange polynomial that precisely satisfies $\mathcal{L}_j(net_{qj}) = O_{qj}$ is developed as

$$\mathcal{L}_j(net) = \sum_{p=1}^P O_{pj} \cdot l_p^j(net) \quad (5)$$

$$l_p^j(net) = \prod_{k=1, k \neq p}^P \frac{net - net_{kj}}{net_{pj} - net_{kj}} \quad (6)$$

where net is a real variable. Replacing net by net_{qj} in (6), we see that $l_p^j(net_{qj})$ equals $\delta(q - p)$ and that $\mathcal{L}_j(net_{qj})$ is indeed O_{qj} . It is clear that $l_p^j(net)$ is a $P - 1$ degree polynomial and that we can rewrite $\mathcal{L}_j(net)$ as

$$\mathcal{L}_j(net) = \sum_{n=0}^{P-1} a_{jn}(net)^n \quad (7)$$

where a_{jn} are the Lagrange polynomial coefficients. The j th hidden unit's activation function is precisely equal to the interpolation polynomial for our training patterns, so replacing net by net_{pj}

in (7), we get

$$O_{pj} = \sum_{n=0}^{P-1} a_{jn}(net_{pj})^n \quad (8)$$

Substituting Eq. (8) into Eq. (4), we get

$$t_{pi} = \sum_{k=1}^{N+1} w_{oi}(i, k) \cdot x_{pk} + \sum_{j=1}^h w_{oh}(i, j) \sum_{n=0}^{P-1} a_{jn}(net_{pj})^n \quad (9)$$

For the memorized training patterns, note that (9) is exact, rather than being an approximation of (4). Note that the coefficients a_{jn} are completely dependent on the net functions net_{pj} . Therefore, the a_{jn} are dependent on the weights $w_h(j, n)$ in Eq. (1), and are dependent rather than independent variables. Substituting (1) into (9), we get

$$t_{pi} = \sum_{k=1}^{N+1} w_{oi}(i, k) \cdot x_{pk} + \sum_{j=1}^h w_{oh}(i, j) \sum_{n=0}^{P-1} a_{jn} \left(\sum_{m=1}^{N+1} w_h(j, m) \cdot x_{pm} \right)^n \quad (10)$$

In general, the weights are unknown while the training patterns $\{\mathbf{x}_p, \mathbf{t}_p\}_{p=1}^P$ are known. Each ordered pair (p, i) corresponds to a different equation. Taking all values of p and i into account, (10) corresponds to $P \cdot M$ nonlinear equations in N_w unknowns where N_w is, again, the number of network weights and thresholds. For the fully connected MLP,

$$N_w = M(N + 1 + h) + h(N + 1) \quad (11)$$

Note again that N_w does not count the coefficients a_{jn} , since these are completely dependent upon the input weights $w_h(j, n)$ in Eq. (1). After abbreviating the right-hand side of Eq. (10), we have

$$t_{pi} = f_{pi}(\mathbf{w}, \mathbf{A}) \quad (12)$$

where \mathbf{w} is a vector of dimension N_w that contains all the network weights and thresholds, and the matrix \mathbf{A} stores the Lagrange polynomial coefficients a_{jn} . If N_w is equal to $P \cdot M$, a solution that satisfies all the above $P \cdot M$ equations is possible. In other words, for an MLP, the storage capacity, that is the number of patterns that can be memorized by a network with h hidden units, is equal to the total number of weights in the network divided by the number of outputs,

$$P = \frac{N_w}{M} = \frac{M(N + 1 + h) + h(N + 1)}{M} \quad (13)$$

A direct or constructive approach for solving (12) for \mathbf{w} could involve back substitution. In this approach, each equation is solved for one weight or threshold, and the expression is substituted into the remaining equations. The order in which the equations are solved is as follows. For each pattern, we vary i from 1 to M . For each value of i , we solve for one weight that is in the path leading to the i th output. After a weight is solved for $i = M$, we increment p , set $i = 1$, and continue. The first $(N + 1)$ times a given value of i occurs, we solve for one of the bypass weights $w_{oi}(i, k)$ feeding into the i th output. Then weights $w_{oh}(i, j)$ are solved for. Then the input weights $w_h(j, n)$ are solved for.

There are at least three problems with a direct approach for solving for \mathbf{w} . First, the validity of back substitution is questionable as closed form expressions for roots of high degree polynomials do not exist. Second, the complication of the equations increases with each equation. Third, it is not clear how to find the coefficients a_{jn} and the weights simultaneously.

In order to avoid the problems with a constructive proof, we can try a proof by contradiction. Here, we assume that \mathbf{w} and therefore \mathbf{A} are available. Keeping \mathbf{A} constant, we can use back substitution to verify the elements of \mathbf{w} , one at a time.

3.2. Theorem and proof

With this, we can formally state our theorem on the storage capacity of feedforward networks.

Theorem. For a feedforward network with N inputs, M outputs, h hidden units and arbitrary finite hidden activation functions, the number of unique patterns P that can be memorized with no error is less than or equal to number of free parameters of the network, N_w divided by the number of outputs, M .

In other words,

$$P \leq \left\lfloor \frac{N_w}{M} \right\rfloor \quad (14)$$

where $\lfloor \cdot \rfloor$ denotes truncation.

Proof. Let us assume that (1) the network can memorize $(N_w/M) + 1$ patterns and (2) that we know the values of all the weights and the coefficients a_{jn} . Letting the a_{jn} , and x_{pn} , and t_{pi} take on their correct values, our strategy is to solve each equation for one weight, employing back substitution.

The first step in this proof is to explain the type of back substitution used. Without assumption (2), this would be a daunting task. Note that in (10), after multiplying out $(net_{pi})^n$, each weight can occur many times. However, (10) can be simplified by substituting the correct value of the weight for most occurrences of the weight variable. The specific procedure is as follows:

- (i) In the equation to be solved, pick one instance of one variable. After solving for that instance of the variable, replace the other instances with the correct numerical value, available by assumption (2).
- (ii) In the remaining equations, replace all instances of the variable by the expression generated in (i).

Consider two successive examples of (12), to be the following:

$$17 = 5w_1w_2^8 + 4w_3^3w_2^5 + 3w_4^8 \quad (15)$$

$$66 = 8w_1w_2^4 + 3w_6^3w_2^5 + 7w_5^4 \quad (16)$$

Following rule (i), and solving for an occurrence of w_2 in (15), we have

$$w_2 = \frac{17 - 4w_3^3w_2^5 - 3w_4^8}{5w_1w_2^7} \quad (17)$$

If the correct numerical value of w_2 is 1, Eq. (17) becomes

$$w_2 = \frac{17 - 4w_3^3 - 3w_4^8}{5w_1} \quad (18)$$

Following rule (ii), and substituting Eq. (18) into (16), we get

$$8w_1 \left[\frac{17 - 4w_3^3 - 3w_4^8}{5w_1} \right]^4 + 3w_6^3 \left[\frac{17 - 4w_3^3 - 3w_4^8}{5w_1} \right]^5 + 7w_5^4 = 66 \quad (19)$$

At this point, (15) has been solved for w_2 and w_2 has been eliminated from (16). Hence we do not need a closed form expression for roots of Eq. (15), which is eighth degree. There are two cases.

Case 1 (The set of equations in (12) is not solvable for at least one ordered pair (p, i)): If (12) has no solution, assumption (1) is disproved and the theorem is confirmed for the given dataset.

Case 2 (The set of equations in (12) has solutions for all ordered pairs (p, i)): For this case, the proof continues to the second step.

The number of nonlinear equations that need to be solved is $PM = (N_w/M + 1)M = N_w + M$. So, we have $N_w + M$ equations in N_w unknowns.

Case 2.1 (Back substitution eliminates exactly one equation and one unknown at a time): Here, we solve exactly one equation at a time and substitute its solution into the remaining equations. By the end of this back substitution procedure, we will be left with M equations and no unknowns. Hence, the network weights that are already found must satisfy the remaining M equations in order to memorize $P = N_w/M + 1$ distinct patterns. This in general is not possible as we shall see in Case 2.2.

Continuing, equation number N_w may be very complicated, but it will have only one unknown. For this first sub-case, there are M equations left over that cannot be solved. For this sub-case, therefore, the theorem is proved by contradiction.

Case 2.2 (In at least one back-substitution step, more than one equation is solved): Suppose that two or more equations are solved by a single back-substitution step. This means that all unknown weights and thresholds for the two equations have been eliminated, and the right-hand side of (12) is constant for both equations. Also, the desired output t_{pi} for the second equation is TM. We now change t_{pi} on the left-hand side of (12) for the second equation, so t_{pi} is no longer TM. Since all the weights have been solved for in both equations, the change in the second equation's t_{pi} does not necessitate changes in the coefficients a_{jn} . For the second equation solved, we have a constant, t_{pi} , and an equal value of $f_{pi}(\mathbf{w}, \mathbf{A})$ on the right-hand side, in (12). We now change t_{pi} slightly so that the equation is not solved. This change does not invalidate any of the equations already successfully solved. For this second sub-case, there are again M equations left over that cannot be solved. Therefore, the theorem is proved by contradiction. \square

Now that we have proved the theorem, we can investigate the generality of the bound.

3.3. Generality of the bound

Although the type of net function used in Eq. (1) can affect the degree of Eq. (10) and the values of a_{jn} , it has no effect on the structure and validity of the proof. Therefore, we have freedom to rewrite Eq. (1) as

$$net_{pi} = \sum_{k=1}^N [x_{pk} - w_h(j, k)]^2 \quad (20)$$

The theorem is then valid for RBF networks. Note as well that the connectivity of the network has no effect, except to alter the actual value of N_w . The theorem is therefore valid for very general feedforward networks with arbitrary connectivity.

4. Numerical results

In this section, we experimentally demonstrate the tightness of the upper bound. We generated three datasets to observe the memorization.

4.1. Dataset 1

This dataset has 15 inputs, two outputs and $P = 340$ patterns. All the inputs and outputs are Gaussian random numbers with zero mean and unit standard deviation. The number of hidden

units required to memorize all the P patterns according to (13) and the Theorem satisfies

$$h \geq \frac{P \cdot M - M \cdot (N + 1)}{N + M + 1} \tag{21}$$

Plugging in $P = 340$, $N = 15$ and $M = 2$, we get

$$h \geq 36 \tag{22}$$

MLP networks of different sizes were trained using the conjugate gradient algorithm [15–17,3,21,8] on the dataset. Note that this algorithm does not have any user-chosen parameters. A plot of the MSE for different network sizes is shown in Fig. 1. The plot shows the MSE for networks of different sizes starting from zero hidden units up to 50 at intervals of five. It is observed that the error curve changes dramatically from the linear network case (zero hidden units) to the neighborhood of $h = 35$ hidden units. After 40 hidden units, the change in error is negligible. In order to show the fine variation of the MSE around $h = 36$, we have tabulated the values of MSEs for $h = 33, 34, 35, 36, 37, 38$ in Table 1. Due to numerical errors, the MSE values do not go absolutely to zero at $h = 36$ hidden units. However, the relative MSE decrease is pronounced when we go from $h = 35$ to 36. This confirms that the network takes around $h = 36$ hidden units to memorize all the patterns, which agrees with the theorem. Unfortunately, the good performance of conjugate gradient on random data does not extend to the case of correlated data [17].

We repeated the experiment on two other random datasets with bigger dimension outputs.

4.2. Dataset 2

This dataset has $N = 10$ inputs, $M = 3$ outputs and $P = 151$ random patterns (zero-mean, unit variance Gaussian random numbers for inputs and outputs). According to the theorem, we have $h \geq 30$ hidden units. From Fig. 2, we can observe that the decrease in error is negligible after the number of hidden units

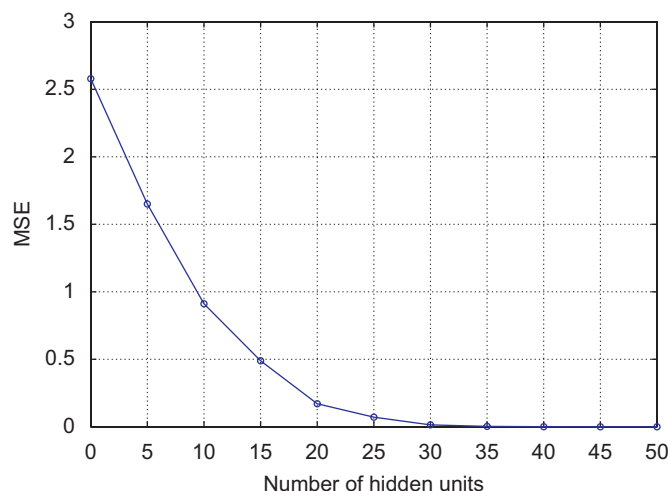


Fig. 2. MSE versus number of hidden units (h) for dataset 2.

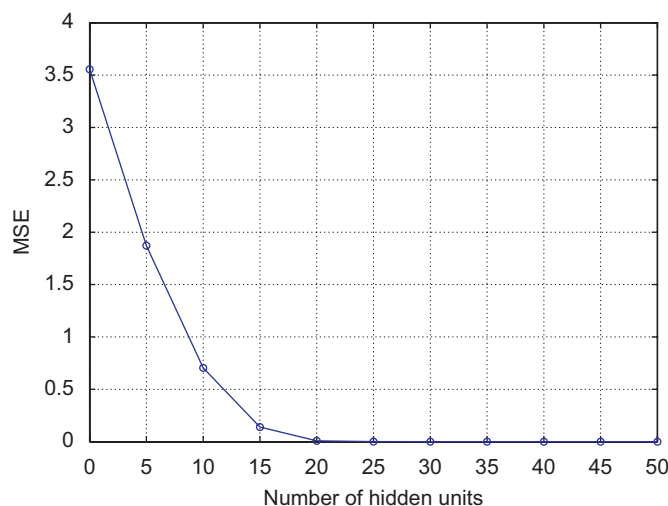


Fig. 3. MSE versus number of hidden units (h) for dataset 3.

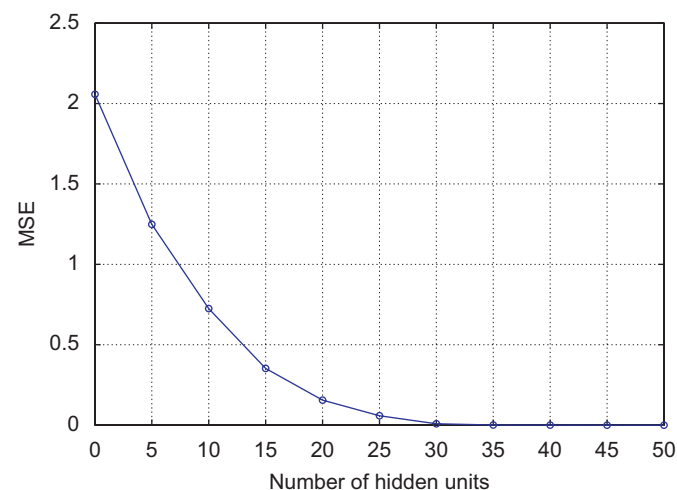


Fig. 1. MSE versus number of hidden units (h) for dataset 1.

Table 1
MSE for networks with h varying around 35 on the dataset 1

h	33	34	35	36	37	38
MSE	0.005630	0.002382	0.002188	0.000780	0.000664	0.000229

reaches the described bound, which again agrees with the theorem.

4.3. Dataset 3

This dataset has $N = 25$ inputs, $M = 5$ outputs and $P = 150$ random patterns. According to the theorem, we have $h \geq 20$ hidden units. Fig. 3 gives the plot of MSE versus number of hidden units. Again the point at which the MSE value becomes small matches with the theoretical bound.

5. Conclusions

We have developed a simple proof of an upper bound on the storage capacity of feedforward networks, with arbitrary finite hidden unit activation functions, including MLP and RBF networks. Since the upper bound is in terms of weights rather than hidden units, it applies even when the network has arbitrary connectivity. The validity and tightness of the upper bound has been demonstrated for a few examples with random patterns.

Many more examples should be tried before we can be confident that the bound is always tight.

References

- [1] Y.S. Abu-Moustafa, J.-M.S. Jacques, Information capacity of hopfield model, *IEEE Trans. Inf. Theory* 31 (4) (1985) 461–464.
- [2] E.B. Baum, On the capabilities of multilayer perceptrons, *J. Complexity* 4 (1988) 193–215.
- [3] C. Charalambous, Conjugate gradient algorithm for efficient training of artificial neural networks, *IEE Proc. Circuits Devices Syst.* 139 (3) (1992) 301–310.
- [4] M. Cosnard, P. Koiran, H. Paugam-Moisy, Bounds on the number of units for computing arbitrary dichotomies by multilayer perceptrons, *J. Complexity* 10 (1994) 57–63.
- [5] P.J. Davis, *Interpolation and Approximation*, Blaisdell Publishing Company, 1963.
- [6] A. Elisseeff, H. Paugam-Moisy, Size of multilayer networks for exact learning: analytic approach, in: *Neural Information Processing Systems*, 1996.
- [7] A. Elisseeff, H. Paugam-Moisy, Size of multilayer networks for exact learning: analytic approach, Technical Report Series (NC-TR-97-002), NEUROCOLT, 1997.
- [8] R. Fletcher, C.M. Reeves, Function minimization by conjugate gradients, *Comput. J.* 7 (1964) 149–154.
- [9] S. Haykin, *Neural Networks: A Comprehensive Foundation*, second ed., Pearson Education, 2004.
- [10] G.-B. Huang, Learning capability and storage capacity of two-hidden-layer feedforward networks, *IEEE Trans. Neural Networks* 14 (2) (2003) 274–281.
- [11] G.-B. Huang, H.A. Babri, Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation function, *IEEE Trans. Neural Networks* 9 (1) (1998) 224–229.
- [12] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, *Neurocomputing* 70 (2006) 489–501.
- [13] H. Jeffreys, B.S. Jeffreys, *Methods of Mathematical Physics*, chap. Lagrange's Interpolation Formula, third ed., Cambridge University Press, Cambridge, England, 1988, p. §9.011, p. 260.
- [14] C. Ji, D. Psaltis, Capacity of two-layer feedforward neural networks with binary weights, *IEEE Trans. Inf. Theory* 44 (1) (1998) 256–268.
- [15] E.M. Johansson, F.U. Dowl, D.M. Goodman, Backpropagation learning for multilayer feed-forward neural networks using the conjugate gradient method, *J. Neural Syst.* 2 (4) (1992) 291–301.
- [16] T. Kim, M.T. Manry, F. Maldonado, New learning factor and testing methods for conjugate gradient training algorithm, in: *International Joint Conference on Neural Networks*, vol. 3, 2003.
- [17] T.-H. Kim, J. Li, M.T. Manry, Evaluation and improvement of two training algorithms, in: *The 36th Asilomar Conference on Signals, Systems, and Computers*, 2002.
- [18] S. Ma, C. Ji, Performance and efficiency: recent advances in supervised learning, *Pro. IEEE* 87 (9) (1999) 1519–1535.
- [19] C. Mazza, On the storage capacity of nonlinear neural networks, *Neural Networks* 10 (4) (1997) 593–597.
- [20] Y.S.A. Moussaoui, H.A. Abbassi, Hybrid hot strip rolling force prediction using a Bayesian trained artificial neural network and analytical models, *Am. J. Appl. Sci.* 3 (6) (2006) 1885–1889.
- [21] W. Murray, *Numerical Methods for Unconstrained Optimization*, Academic Press, New York, 1972.
- [22] P.L. Narasimha, M.T. Manry, F. Maldonado, Upper bound on pattern storage in feedforward networks, in: *Proceedings of International Joint Conference on Neural Networks*, IEEE, Orlando, FL, 2007.
- [23] M.J.D. Powell, Radial basis function approximations to polynomials, in: *Numerical Analysis 1987 Proceedings*, 1988, pp. 223–241.
- [24] H. Suyari, I. Matsuba, New approach to the storage capacity of neural networks using the minimum distance between input patterns, in: *International Joint Conference on Neural Networks*, vol. 1, 1999.



Pramod Lakshmi Narasimha received his B.E. in Telecommunications Engineering from Bangalore University, India. He joined the Image Processing and Neural Networks Lab in the Department of Electrical Engineering (EE) at the University of Texas at Arlington (UTA) as a research assistant in 2002. He received his M.S. degree in 2003 and Ph.D. degree in 2007, both from the EE department at UTA. Currently, he is a Member of Technical Staff at FastVDO Inc. at Maryland. His research interests focus on machine learning, neural networks, estimation theory, pattern recognition and computer vision. He is a member of Tau Beta Pi and a student member of the IEEE.



Michael T. Manry was born in Houston, TX in 1949. He received the B.S., M.S. and Ph.D. in Electrical Engineering in 1971, 1973 and 1976, respectively, from The University of Texas at Austin. After working there for 2 years as an Assistant Professor, he joined Schlumberger Well Services in Houston where he developed signal processing algorithms for magnetic resonance well logging and sonic well logging. He joined the Department of Electrical Engineering at the University of Texas at Arlington in 1982, and has held the rank of Professor since 1993. In summer 1989, Dr. Manry developed neural networks for the Image Processing Laboratory of Texas Instruments in Dallas. His recent work, sponsored by the Advanced Technology Program of the state of Texas, E-Systems, Mobil Research, and NASA has involved the development of techniques for the analysis and fast design of neural networks for image processing, parameter estimation and pattern classification. Dr. Manry has served as a consultant for the Office of Missile Electronic Warfare at White Sands Missile Range, MICOM (Missile Command) at Redstone Arsenal, NSF, Texas Instruments, Geophysics International, Halliburton Logging Services, Mobil Research and Verity Instruments. He is a Senior Member of the IEEE.



Francisco Javier Maldonado Diaz received the bachelor of Industrial Engineering in Electronics from the Instituto Tecnológico de Veracruz in 1985 and a M.S. degree in Electrical Engineering from the Instituto Tecnológico de Chihuahua (ITCH) in 1988. In 1988 he obtained a grant from the National System of Researchers (Mexico) and became part of the Faculty of the ITCH. There he performed research digital systems with applications in control and robotics. In 1997 he obtained a Fulbright-Conacyt scholarship and started his Ph.D. program in EE at the University of Texas at Arlington, both in the Advanced Control and Sensor group at the Automation and Robotics Research Institute (ARRI), and at the Image Processing and Neural Networks Laboratory. During 2001 and 2002 he worked at Williams-Pyro, Inc. (WPI) in Fort Worth, where he developed SBIR projects for NASA, the Navy, NIST, the Air Force and the DOT. After a short interlude as a Professor at the Instituto Tecnológico de Chihuahua during 2003–2005, he returned to WPI as researcher and systems engineer. His areas of interest are artificial neural networks, hardware development for DSP, digital systems, feature extraction algorithms, artificial intelligence and robotics. He is a member of the IEEE Control System Society and the IEEE Instrumentation and Measurement Society. He has been a member and official of the Society of Hispanic Engineers at UTA. He has been a member of HKN since 1998.