Reliable Computing **4**: 235–239, 1998.

Universal Approximation Theorem for Interval Neural Networks

MARK R. BAKER
Geo Media Research and Development, 6040 S. Strahan Rd., El Paso, TX 79932, USA

and

RAJENDRA B. PATIL
Computing Research and Applications (CIC-3), MS-M986, Los Alamos National Laboratory, Los Alamos, NM 87545, USA
Current address: General Electric, Corporate R & D, P. O. Box 8, Information Technology Laboratory (ITL), Bldg. K-1, Room 5C16A, Schenectady, NY 12301, USA, e-mail: patil@crd.ge.com

(Received: 15 March 1995; accepted: 25 December 1997)

Abstract. One of the main computer-learning tools is an (artificial) neural network (NN); based on the values $y^{(p)}$ of a certain physical quantity y at several points $x^{(p)} = (x_1^{(p)}, \dots, x_n^{(p)})$, the NN finds a dependence $y = f(x_1, \dots, x_n)$ that explains all known observations and predicts the value of y for other $x = (x_1, \dots, x_n)$. The ability to describe an arbitrary dependence follows from the *universal approximation* theorem, according to which an arbitrary continuous function of a bounded set can be, within a given accuracy, approximated by an appropriate NN.

The measured values of y are often only known with interval uncertainty. To describe such situations, we can allow interval parameters in a NN and thus, consider an *interval* NN. In this paper, we prove the universal approximation theorem for such interval NN's.

Neural networks. At present, one of the main computer learning tools is an (artificial) *neural network* that simulates, on the level of neurons, human ability to learn (see, e.g., [2]). One of the most widely used neural networks is based on a neuron which takes n input signals $x_1, \dots, x_n$ and returns the output $y = f(w_1 \cdot x_1 + \cdots + w_n \cdot x_n + w_0)$, where w_i are real numbers called *weights*, and $f(x)$ is, usually, either a so-called *logistic* function $f_0(x) = 1 / (1 + \exp(-x))$ (for *non-linear* neurons), or an identity function $f(x) = x$ (for *linear* neurons). To form a *neural network*, we send the output of these neurons as inputs to other neurons. Two types of neural architecture are most widely used: 3-layer and 4-layer. In both architectures:

- the first layer (called *input layer*) inputs the signals x_i;
- each intermediate layer consists of non-linear neurons that process the outputs of the previous layer:
  - the second layer consists of K non-linear neurons, each of which ($k = 1, \dots, K$) transforms the inputs $x_1, \dots, x_n$ into an output

$$y_k = f_0(w_{k1} \cdot x_1 + \cdots + w_{kn} \cdot x_n + w_{k0});$$

- in a 4-layer network, the third layer consists of L non-linear neurons, each of which transforms the inputs $y_1, ..., y_K$ into a signal

$$z_l = f_0(w'_{l1} \cdot y_1 + \cdots + w'_{lK} \cdot y_K + w'_{l0}).$$

- the last layer (called *output layer*) contains a single linear neuron that combines the results of the previous layer into an output y:
  - for a 3-layer network, we get $y = W_1 \cdot y_1 + \cdots + W_K \cdot y_K + W_0$;
  - for a 4-layer network, we get $y = W_1 \cdot z_1 + \cdots + W_L \cdot y_L + W_0$.

For each set of weights w_{ki}, w'_{lk}, W_j, these formulas lead to a function that describes y in terms of $x_1, ..., x_n$. For a 3-layer network, we have

$$y = f_{NN}(x_1, ..., x_n) = \sum_{k=1}^{K} W_k \cdot f_0 \left(\sum_{i=1}^{n} w_{ki} \cdot x_i + w_{k0} \right) + W_0. \tag{1}$$

For a 4-layer network, we have

$$y = f_{NN}(x_1, ..., x_n) = \sum_{l=1}^{L} W_l \cdot f_0 \left(\sum_{k=1}^{K} w'_{lk} \cdot y_k + w_{l0} \right) + W_0, \tag{2}$$

where

$$y_k = f_0 \left(\sum_{i=1}^{n} w_{ki} \cdot x_i + w_{k0} \right). \tag{3}$$

We are usually given some *patterns* $(x_1^{(p)}, ..., x_n^{(p)}, y^{(p)})$, and we want to find the weights for which, for each pattern, $y^{(p)} \approx f_{NN}(x_1^{(p)}, ..., x_n^{(p)})$. Finding such weights is called *learning*.

The fact that such weights always exist (for appropriate K and/or L) follows from the result known as the *universal approximation theorem* [3]–[5], [7], [8]: For every continuous function $f(x_1, ..., x_n)$ on a compact set $M \subset R^n$ and for every $\varepsilon > 0$, there exist weights for which for every $(x_1, ..., x_n) \in M$, $|f(x_1, ..., x_n) - f_{NN}(x_1, ..., x_n)| \le \varepsilon$. This result is true both for 3-layer and for 4-layer networks.

Interval neural networks. From the mathematical viewpoint, a neural network is an extrapolation tool. For example, in geophysics, if we measure the density d at several depths and locations, we can use a neural network to extrapolate this data and thus, to predict the density at all possible depths and locations.

The data from which we extrapolate is often known with a reasonable inaccuracy: instead of a single value of density, we usually have an *interval* $\mathbf{d} = [\underline{d}, \overline{d}]$ of possible values of density. It is therefore desirable to predict, for all other depths and locations, not just a single value, but the *interval* of possible values of density (so that we not only have the prediction, but also the accuracy of this prediction). In other words, we want to extrapolate the *interval-valued* function $\mathbf{d}(x_1, ..., x_n) = [\underline{d}(x_1, ..., x_n), \overline{d}(x_1, ..., x_n)]$.

From the mathematical viewpoint, an interval-valued function can be viewed as two real-valued functions $\underline{d}$ and $\overline{d}$. Therefore, we can use two different neural networks to describe these two functions (a similar idea was proposed in [9]). The disadvantage of this approach is that, even if take the patterns from the functions $\underline{f}$ and $\overline{f}$ for which always $\underline{f}(x_1, \ldots, x_n) \leq \overline{f}(x_1, \ldots, x_n)$, due to approximate character of neural network approximations ($\underline{f}_{NN} \approx \underline{f}$ and $\overline{f}_{NN} \approx \overline{f}$), we sometimes end up with $\underline{f}_{NN}(x_1, \ldots, x_n) > \overline{f}_{NN}(x_1, \ldots, x_n)$, in which case

$$[\underline{f}_{NN}(x_1, \ldots, x_n), \overline{f}_{NN}(x_1, \ldots, x_n)]$$

is *not* a meaningful interval.

To solve this problem, a notion of an *interval neural network* was invented [6] (see also [10]), in which, instead of two different neural networks, we have a single network, but with interval-valued weights $\mathbf{w}_{ki}$, $\mathbf{w}'_{lk}$, and $\mathbf{W}_j$. For a 3-layer network, we have

$$\begin{aligned} \mathbf{y} &= \mathbf{f}_{NN}(x_1, \ldots, x_n) = [\underline{f}_{NN}(x_1, \ldots, x_n), \overline{f}_{NN}(x_1, \ldots, x_n)] \\ &= \sum_{k=1}^{K} \mathbf{W}_k \cdot f_0 \left(\sum_{i=1}^{n} \mathbf{w}_{ki} \cdot x_i + \mathbf{w}_{k0} \right) + \mathbf{W}_0, \end{aligned} \tag{4}$$

where all the operations with intervals are understood in the sense of interval arithmetic (see, e.g., [1]).

For a 4-layer network, we have

$$\begin{aligned} \mathbf{y} &= \mathbf{f}_{NN}(x_1, \ldots, x_n) = [\underline{f}_{NN}(x_1, \ldots, x_n), \overline{f}_{NN}(x_1, \ldots, x_n)] \\ &= \sum_{l=1}^{L} \mathbf{W}_l \cdot f_0 \left(\sum_{k=1}^{K} \mathbf{w}'_{lk} \cdot \mathbf{y}_k + \mathbf{w}_{l0} \right) + \mathbf{W}_0, \end{aligned} \tag{5}$$

where

$$\mathbf{y}_k = f_0 \left(\sum_{i=1}^{n} \mathbf{w}_{ki} \cdot x_i + \mathbf{w}_{k0} \right). \tag{6}$$

With these formulas, for all possible weight combinations, and for all possible values $x_1, \ldots, x_n$, we get an interval.

There exists methods of training such a network, which are experimentally rather successful [6]. However, in contrast to real-valued neural networks, we did not have a universal approximation theorem that would guarantee that such weights can be found. This theorem is proven in this paper.

THEOREM. *For every continuous interval-valued function*

$$\mathbf{f}(x_1, \ldots, x_n) = [\underline{f}(x_1, \ldots, x_n), \overline{f}(x_1, \ldots, x_n)]$$

on a compact set $M \subset R^n$ and for every $\varepsilon > 0$, there exist weights for which for every $(x_1, \ldots, x_n) \in M$,

$$|\underline{f}(x_1, \ldots, x_n) - \underline{f}_{NN}(x_1, \ldots, x_n)| \leq \varepsilon \quad \textit{and}$$
$$|\overline{f}(x_1, \ldots, x_n) - \overline{f}_{NN}(x_1, \ldots, x_n)| \leq \varepsilon.$$

Comment. In other words, *4-layer interval neural networks are universal approximators for interval-valued functions.*

We could not prove a similar result for 3-layer networks; moreover, since learning algorithms often fail for 3-layer neural networks, we conjecture that 3-layer networks may not be universal approximators after all.

Proof. We will approximate the given interval-valued function by a 4-layer neural network in which the third layer has two neurons, and in which all the weights are real numbers (degenerate intervals) except for the weight $\mathbf{W}_2 = [0, 1]$. To be more precise, we want to use a neural network for which

$$\mathbf{f}_{NN}(x_1, \ldots, x_n) = [1, 1] \cdot z_1(x_1, \ldots, x_n) + [0, 1] \cdot z_2(x_1, \ldots, x_n),$$

where $z_j = f_0(u_j)$, and

$$u_j(x_1, \ldots, x_n) = \sum_{k=1}^{K} w'_{jk} \cdot f_0\left(\sum_{i=1}^{n} w_{ki} \cdot x_i + w_{k0}\right) + w'_{j0}.$$

To guarantee that this network produces an ε approximation to the original interval-valued function, it is sufficient to guarantee that z_1 is $(\varepsilon / 3)$-close to $\underline{f}$, and z_2 is $(\varepsilon / 3)$-close to $\Delta f = \overline{f} - \underline{f} + \varepsilon / 3$. Indeed, if these closeness conditions are satisfied, then, from $\Delta f \geq \varepsilon / 3$, and from the fact that z_2 is $(\varepsilon / 3)$-close to Δf, we can conclude that $z_2 \geq 0$ and therefore, that $\underline{f}_{NN} = z_1$ and $\overline{f}_{NN} = z_1 + z_2$.

- $\underline{f}_{NN} = z_1$ is $(\varepsilon / 3)$-close to $\underline{f}$, and therefore, ε-close too.
- Since z_1 is $(\varepsilon / 3)$-close to $\underline{f}$, and z_2 is $(\varepsilon / 3)$-close to $\Delta f = \overline{f} - \underline{f} + \varepsilon / 3$, we can conclude that $z_1 + z_2$ is $(2\varepsilon / 3)$-close to $\underline{f} + \Delta f = \overline{f} + \varepsilon / 3$. Therefore, the sum $\overline{f}_{NN} = z_1 + z_2$ is ε-close to $\overline{f}$.

The function $f_0(x)$ is uniformly continuous, hence, there exists a $\delta > 0$ such that if $|x - x'| \leq \delta$, then $|f_0(x) - f_0(x')| \leq \varepsilon / 3$. Therefore, to guarantee that $z_1 = f_0(u_1)$ is $(\varepsilon / 3)$-close to $\underline{f}$, it is sufficient to guarantee that u_1 is δ-close to the function $F_1(x_1, \ldots, x_n) = f_0^{-1}(\underline{f}(x_1, \ldots, x_n))$. Similarly, to guarantee that $z_2 = f_0(u_2)$ is $(\varepsilon / 3)$-close to Δf, it is sufficient to guarantee that u_2 is δ-close to the function $F_2(x_1, \ldots, x_n) = f_0^{-1}(\Delta f(x_1, \ldots, x_n))$.

The expressions for u_1 and u_2 are exactly the expressions for the 3-layer neural network, so, the existence of the weights for which u_1 is δ-close to $F_1(x_1, \ldots, x_n)$ follows from the universal approximation theorem for 3-layer real-valued neural networks. Similarly, we can prove the existence of weights that compute the desired u_2. □

Acknowledgements. The authors are thankful to Hung T. Nguyen, Vladik Kreinovich, and Věra Kůrková for valuable discussions.

References

1. Alefeld, G. and Herzberger, J.: *Introduction to Interval Computations*, Academic Press, NY, 1983.
2. Haykin, S.: *Neural Networks: A Comprehensive Foundation*, Macmillan College Publishing Company, New York, NY, 1994.
3. Hecht-Nielsen, R.: Kolmogorov's Mapping Neural Network Existence Theorem, in: *Proceedings of First IEEE International Conference on Neural Networks*, San Diego, CA, 1987, pp. 11–14.
4. Hornik, K., Stinchcombe, M., and White, H.: Multilayer Feedforward Neural Networks Are Universal Approximators, *Neural Networks* **2** (1989), pp. 359–366.
5. Hornik, K.: Approximation Capabilities of Multilayer Feedforward Neural Networks. *Neural Networks* **4** (1991), pp. 251–257.
6. Ishibuchi, H. and Tanaka, H.: An Architecture of Neural Networks with Interval Weights and Its Application to Fuzzy Regression, *Fuzzy Sets and Systems* **57** (1993), pp. 27–39.
7. Kůrková, V.: Kolmogorov's Theorem Is Relevant, *Neural Computation* **3** (1991), pp. 617–622.
8. Kůrková, V.: Kolmogorov's Theorem and Multilayer Neural Networks, *Neural Networks* **5** (1992), pp. 501–506.
9. Nesterov, V. M.: Interval Analogues of Hilbert's 13th Problem, in: *Abstracts of the Int'l Conference Interval'94, St.Petersburg, Russia, March 7–10, 1994*, pp. 185–186.
10. Patil, R. B.: Interval Neural Networks, in: *Extended Abstracts of APIC'95: International Workshop on Applications of Interval Computations, El Paso, TX, Febr. 23–25, 1995*, Reliable Computing (1995), Supplement, p. 164.