BACKPROPAGATION FOR INTERVAL PATTERNS [1]

Radim Bělohlávek

Institute for Research and Applications of Fuzzy Modeling
University of Ostrava
Bráfova 7, 701 03 Ostrava, Czech Republic
e-mail: belohlav@osu.cz
and
Department of Computer Science
Technical University of Ostrava
tř. 17. listopadu, 708 33 Ostrava-Poruba, Czech Republic

Abstract. In [9], the author proposed a modified architecture of multilayer feedforward neural network which is able to work with and to be adapted to interval training patterns. We analyze this architecture, propose a new one based strictly on the interval arithmetic and give a backpropagation like adaptation algorithm. Possible applications of our architecture are discussed. The role of interval networks in approximate reasoning is emphasized.

Key words: Multilayer neural net, steepest descent, backpropagation, interval training patterns.

1 Introduction

Since the backpropagation learning algorithm appeared in the context of artificial neural networks, see [7], (in fact, as revealed later, back propagating errors appeared earlier, see e.g. [1]) multilayer feedforward neural networks became the most frequently used neural network paradigm.

The network consists of a number of nodes performing simple transfer functions connected into a layered architecture. The signal is fed through the network from the input to the output places. Mathematically, the network represents a function composed of simple parametrized functions. Changing appropriately the parameters, the network can be adapted to approximate a function prescribed partially by a training set. This basic idea can be modified in various ways (see [1]).

In the 1960s scientists recognized the need of developing methods capable of effective dealing with imprecise information. The use of not precisely given data is not a mere way to simplify our life when it is not possible to get precise information. Rather, as it becomes clear, the ability of

[1]Supported by grant No. 201/96/0985 of the GA ČR.

processing and use of inexact information plays the crucial role in the way humans cope with the immense complexity of the surrounding world.

A feedforward neural network processing intervals (interval network), a simple kind of indeterminacy, has been described in [8, 9]. An architecture and training algorithm for interval patterns have been developed in [9].

Our paper deals with interval networks. In section 2, we first set the problem. The architecture described in [9] is then analyzed and it is shown that it possesses in some cases undesirable property. The analysis of the architecture based strictly on the interval arithmetic and a steepest descent like training algorithm are covered in section 3. An example and conclusions are presented in sections 4 and 5, respectively.

2 Problem description

The classical multilayer feedforward neural network can be considered as a mapping network assigning n-tupples of real numbers to m-tupples. It is sometimes desirable to have a network which maps tupples of real intervals to tupples of real intervals. There are various reasons for such a network. The inputs to the network may not always be precise numbers, they can be given up to a certain degree of accuracy. An appropriate form for expressing this impreciseness may be intervals. Even the expert knowledge may be expressed in a form of interval values. Another possibility of exploiting the interval network is to reduce the size of a training set for a regular network by appropriate clustering method, e.g. a method which yields rectangular clusters, and adapting the interval networks to this clusters, i.e. interval patterns, which can save time ([9]).

There is a general approach extending a real function $f : \mathcal{R}^n \to \mathcal{R}^m$ which is based on the so called *interval arithmetic*. By this approach we get a function $\overline{f}$ mapping n-tupples of real intervals to m-tupples of subsets of reals as follows

$$\overline{f}(\langle [x_{L1}, x_{R1}], \ldots, [x_{Ln}, x_{Rn}] \rangle) = \langle Y_1, \ldots, Y_m \rangle$$

where

$$Y_j = \{ y_j ; \exists x_i \in [x_{Li}, x_{Ri}] : f(\langle x_1, \ldots x_n \rangle) = \langle y_1, \ldots, y_{j-1}, y_j, y_{j+1}, \ldots y_m \rangle \} \ .$$

This approach can be applied to the multilayer feedforward network. Given a regular network performing a mapping F we get a network performing a mapping $\overline{F}$. A natural property of interval mappings is the monotonicity. An interval mapping is *monotonic* if it preserves the subsethood relation, i.e. if $X_i \subseteq X_i'$ holds for n-tupples of input intervals $\langle X_1, \ldots, X_n \rangle$, $\langle X_1', \ldots, X_n' \rangle$ then we have $Y_j \subseteq Y_j'$ for the corresponding m-tupples of output intervals $\langle Y_1, \ldots, Y_m \rangle$, $\langle Y_1', \ldots, Y_m' \rangle$. The monotonicity of a mapping obtained by interval arithmetic follows directly from definition.

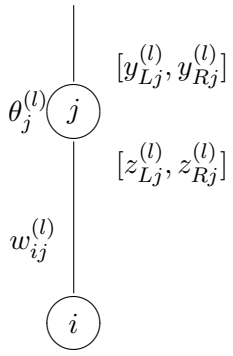

Figure 1: Part of the underlying neural net

The training set in the case of interval information consists of tupples of input-output intervals, i.e.

$$T = \{\langle [x_{1L}^p, x_{1R}^p], \ldots, [x_{mL}^p, x_{mR}^p], [o_{1L}^p, o_{1R}^p], \ldots, [o_{nL}^p, o_{nR}^p]\rangle;\ p \in P\}\ .$$

Given an interval patterns training set T we want to build a network performing approximately the partial mapping T. Our underlying neural network is the usual multilayer feedforward neural network with sigmoidal transfer functions. By interval arithmetic we obtain an interval network. To make the notation clear a part of the network is depicted in Fig. 1. Denote further m_i, $i = 1, \ldots, r$, the number of layers in $i-$th layer. We suppose the number of layers as well as the number of neurons in the respective layers are given, e.g. by an expert, learning thus means appropriate changes of weights and possibly other (nonstructural) parameters.

We firstly describe the feedforward phase. After a tupple of m intervals is presented the network transfers signals layer by layer. In our notation we have $x_{Li} = y_{Li}^{(0)}$ and $x_{Ri} = y_{Ri}^{(0)}$. Suppose we have the output interval signals $[y_{Li}^{(l-1)}, y_{Ri}^{(l-1)}]$ of all the neurons of the $(l-1)$-th layer already at disposal. Then the total input of the j-th neuron in the $l-th$ layer is given by

$$[z_{Lj}^{(l)}, z_{Rj}^{(l)}] = \sum_{i=1}^{m_{l-1}} w_{ij}^{(l)} [y_{Li}^{(l-1)}, y_{Ri}^{(l-1)}] - \theta_j^{(l)}[1,1]$$

($\theta_j^{(l)}$ stands for the threshold) which leads by interval arithmetic to

$$z_{Lj}^{(l)} = \sum_{i=1, w_{ij}^{(l)} \geq 0}^{m_{l-1}} w_{ij}^{(l)} y_{Li}^{(l-1)} + \sum_{i=1, w_{ij}^{(l)} < 0}^{m_{l-1}} w_{ij}^{(l)} y_{Ri}^{(l-1)} - \theta_j^{(l)}$$

and

$$z_{Rj}^{(l)} = \sum_{i=1, w_{ij}^{(l)} \geq 0}^{m_{l-1}} w_{ij}^{(l)} y_{Ri}^{(l-1)} + \sum_{i=1, w_{ij}^{(l)} < 0}^{m_{l-1}} w_{ij}^{(l)} y_{Li}^{(l-1)} - \theta_j^{(l)}\ .$$

The last two formulas can be rewritten as follows,

$$z_{Lj}^{(l)} = \sum_{i=1}^{m_{l-1}} (sgn^+(w_{ij}^{(l)}) w_{ij}^{(l)} y_{Li}^{(l-1)} + sgn^-(w_{ij}^{(l)}) w_{ij}^{(l)} y_{Ri}^{(l-1)}) - \theta_j^{(l)} \tag{1}$$

and

$$z_{Lj}^{(l)} = \sum_{i=1}^{m_{l-1}} (sgn^-(w_{ij}^{(l)}) w_{ij}^{(l)} y_{Li}^{(l-1)} + sgn^+(w_{ij}^{(l)}) w_{ij}^{(l)} y_{Ri}^{(l-1)}) - \theta_j^{(l)} \tag{2}$$

where $sgn^+(x)$ is the signum function and $sgn^- = 1 - sgn^+$.

As a transfer function we use the sigmoidal function $y(x) = \frac{1}{1+e^{-x}}$. As this is an increasing function we have

$$[y_{Lj}^{(l)}, y_{Rj}^{(l)}] = \left[\frac{1}{1+e^{-z_{Lj}^{(l)}}}, \frac{1}{1+e^{-z_{Rj}^{(l)}}} \right] .$$

If an adaptive slope parameter λ is used (see e.g. [6]), i.e. $y(x) = \frac{1}{1+e^{-\lambda x}}$, the monotonicity of y is not disturbed, and the whole machinery inclusive of the adaptation can be sustained. We omit λ for simplicity.

Adapting such a network leads to the minimization process of an appropriate error function for which we choose (as in [9]) the usual squared error

$$E = \sum_{p \in P} E^p$$

where

$$E^p = E_L^p + E_R^p = \frac{1}{2} \sum_{i=1}^{m_r} (y_{Li}^{(r)} - o_{Li}^p)^2 + \frac{1}{2} \sum_{i=1}^{m_r} (y_{Ri}^{(r)} - o_{Ri}^p)^2 \ .$$

In order to minimize the error function one usually uses the steepest descent approach. Following this way we have to determine the weight changes in discrete time t by

$$\Delta w_{ij}^{(l)}(t+1) = \eta d_{ij}^{(l)} + \mu \Delta w_{ij}^{(l)}(t)$$

where $\eta > 0$ is the length of the step (learning rate), μ is a momentum constant, and $d = \langle \ldots, d_{ij}^{(l)}, \ldots \rangle$ is the direction of the steepest descent which coincides with the gradient of E in the case of differentiable E.

From (1) and (2) it is clear that the function is not differentiable in general. Stating this fact, the author in [9] uses a little trick to get differentiable error function. The undifferentiable functions sgn^+ and sgn^- are substituted by their differentiable analogies s^+ and s^-, respectively, where

$$s^+(x) = \frac{1}{1+e^{-x}} \quad \text{and} \quad s^-(x) = \frac{1}{1+e^{x}} \ .$$

In fact, a parameter controlling the slope of s^+ and s^- was added in [9] which is omitted here for simplicity. This approximation leads to an inaccurate interval arithmetic. The resulting network is

again an interval network (call it the *CS* (Continuous Signum) *interval network*) which is no longer monotonic.

Observation 1 *The CS interval network is not monotonic.*

Proof. It is easy to get an example of an CS interval network which is nonmonotonic. The crucial point is the nonmonotonicity of multiplying intervals by weights. An example: Let w be such that $s^+(w) = 0.7$. Then $s^-(w) = 1 - 0.7 = 0.3$. Take intervals $[0, x]$, $[0, y]$, $x < y$. Then by (1) and (2), $w[0, x] = [w \cdot 0.3 \cdot x, w \cdot 0.7 \cdot x] \not\subseteq [w \cdot 0.3 \cdot y, w \cdot 0.7 \cdot y] = w[0, y]$ but $[0, x] \subseteq [0, y]$. □

There are situations (see Conclusion) where a monotonic interval network is needed. There is a possibility to train the network with s^+ and s^- instead of sgn^+ and sgn^- and then, in the consulting mode, replace s^+, s^- by sgn^+, sgn^- to get a monotonic network. However, this approach may lead to unsatisfactory results. Hence, it is desirable to have direct adaptation rule for the net obtained by interval arithmetic. We aim at this problem in the following section.

3 Solution

We first analyze properties of the output functions of our interval network. We denote $w = \langle \ldots, w_{ij}^{(l)}, \ldots \rangle$, a point in the space W of all weight vectors. The first property is the continuity of network output functions despite the discontinuity of the signum functions.

Theorem 2 *For given inputs $y_{Li}^{(0)} = x_{Li}$, $y_{Ri}^{(0)} = x_{Ri}$, the functions $y_{Lj}^{(r)} = y_{Lj}^{(r)}(w)$ and $y_{Rj}^{(r)} = y_{Rj}^{(r)}(w)$, $j = 1, \ldots, m_r$, are continuous.*

Proof. The proof can be done by induction by standard arguments of calculus showing that $\lim_{w \to {}^0w} y_{Lj}^{(l)}(w) = y_{Lj}^{(r)}({}^0w)$ (the same for $y_{Rj}^{(l)}$) for any ${}^0w \in W$, and $l = 0, \ldots, r$. For $l = 0$ we have $y_{Lj}^{(0)} = x_{Lj}$ and $y_{Rj}^{(0)} = x_{Rj}$ which are constant, and, therefore, continuous. Suppose $y_{Lj}^{(l-1)}(w)$ and $y_{Rj}^{(l-1)}(w)$ be continuous. We show that $y_{Lj}^{(l)}(w)$ and $y_{Rj}^{(l)}(w)$ are continuous. From $y_{Lj}^{(l)}(w) = \frac{1}{1+e^{-z_{Lj}^{(l)}(w)}}$ and the continuity of $\frac{1}{1+e^{-x}}$ it follows that it is sufficient to show the continuity of $z_{Lj}^{(l)}(w)$. We have

$$\begin{aligned}\lim_{w \to {}^0w} z_{Lj}^{(l)}(w) &= \lim_{w \to {}^0w} \sum_{i=1, {}^0w_{ij}^{(l)} \neq 0}^{m_{l-1}} (sgn^+(w_{ij}^{(l)}) w_{ij}^{(l)} y_{Li}^{(l-1)}(w) + sgn^-(w_{ij}^{(l)}) w_{ij}^{(l)} y_{Ri}^{(l-1)}(w)) - \theta_j^{(l)} \\ &+ \lim_{w \to {}^0w} \sum_{i=1, {}^0w_{ij}^{(l)} = 0}^{m_{l-1}} (sgn^+(w_{ij}^{(l)}) w_{ij}^{(l)} y_{Li}^{(l-1)}(w) + sgn^-(w_{ij}^{(l)}) w_{ij}^{(l)} y_{Ri}^{(l-1)}(w))\end{aligned}$$

provided both limits exist. The first limit equals to $z_{Lj}^{(l)}({}^0w)$. For the second limit we have

$$\lim_{w\to{}^0w} \sum_{i=1,{}^0w_{ij}^{(l)}=0}^{m_{l-1}} (sgn^+(w_{ij}^{(l)}(w))w_{ij}(w)^{(l)}y_{Li}^{(l-1)}(w) + sgn^-(w_{ij}^{(l)}(w))w_{ij}^{(l)}(w)y_{Ri}^{(l-1)}(w)) = 0$$

which follows from the boundedness of $y_{Li}^{(l-1)}(w)$, $y_{Ri}^{(l-1)}(w)$, $sgn^+(w_{ij}^{(l)}(w))$, $sgn^-(w_{ij}^{(l)}(w))$, and from $\lim_{w\to{}^0w,{}^0w_{ij}^{(l)}=0} w_{ij}^{(l)}(w) = 0$. Hence, $z_{Lj}^{(l)}(w)$ is continuous.
For $y_{Rj}^{(l)}(w)$, the proof goes analogously. □.

We now analyze the differentiability properties of the output functions. As mentioned, the output functions are not differentiable in general. However, a weaker form of differentiability still holds. In the following, we briefly present the results as general as necessary in order to keep the possibility to apply them also to similar situations. Complete proofs can be found in [2]. We need the following concepts.

By a *signature* we mean a tupple $\Sigma = \langle \Sigma_1, \ldots, \Sigma_n \rangle$ where $\Sigma_i \in \{+, -, \pm\}$ for $i = 1, \ldots, n$. If Σ is a signature, then by R^Σ we denote the set

$$R^\Sigma = \{\langle x_1, \ldots, x^n \rangle \in \mathcal{R}^n;\ x_i \geq 0 \text{ for } \Sigma_i = +,\ x_i \leq 0 \text{ for } \Sigma_i = -,\ x_i \in \mathcal{R} \text{ for } \Sigma_i = \pm,\ i = 1, \ldots, n\}\ .$$

It is clear that $\mathcal{R}^\Sigma = \mathcal{R}^n$ if $\Sigma_i = \pm$ for all $i = 1, \ldots, n$. Call a function $f : \mathcal{R}^n \mapsto \mathcal{R}$ Σ-*differentiable in* $a \in \mathcal{R}^n$ if there are $c_1, \ldots, c_n \in \mathcal{R}$ such that $\lim_{u\to 0, u\in\mathcal{R}^\Sigma} \frac{f(x+a)-f(a)-(c_1u_1+\cdots+c_nu_n)}{|u|} = 0$. Denote $grad^\Sigma f(a) = \langle \frac{\partial^{\Sigma_1} f(a)}{\partial^{\Sigma_1} x_1}, \ldots, \frac{\partial^{\Sigma_n} f(a)}{\partial^{\Sigma_n} x_n} \rangle$ ($\frac{\partial^{\Sigma_i} f(a)}{\partial^{\Sigma_i} x_i}$ denotes the (one-side) partial derivative). It can be shown that if f is Σ-differentiable in a, then $\langle c_1, \ldots, c_n \rangle = grad^\Sigma f(a)$. Let Σ be a signature, $a \in \mathcal{R}^n$. A Σ-*neighborhood of* a is a subset $U_\Sigma(a) \subseteq \mathcal{R}^n$ such that there is a (regular) neighborhood $U(a)$ with $U_\Sigma(a) = U(a) \cap (a + \mathcal{R}^\Sigma)$ where $a + \mathcal{R}^\Sigma = \{a + u;\ u \in \mathcal{R}^\Sigma\}$. We say that $f : \mathcal{R}^n \mapsto \mathcal{R}$ *has partial derivatives in* $U_\Sigma(a)$ if for each $b \in U_\Sigma(a)$ it holds: if $b_i \neq a_i$ for some i then $\frac{\partial f(b)}{\partial x_i}$ exists, and if $b_i = a_i$ for some i then $\frac{\partial^{\Sigma_i} f(b)}{\partial^{\Sigma_i} x_i}$ exists. The following proposition can be then proved.

Proposition 3 *Let $f : \mathcal{R}^n \mapsto \mathcal{R}$ have partial derivatives in some $U_\Sigma(a)$ and let the partial derivatives be continuous in $U_\Sigma(a)$. Then f is Σ-differentiable in a.*

Theorem 4 *For a given $w = \langle \ldots, w_{ij}^{(l)}, \ldots \rangle$ let Σ be a signature determined as follows: $\Sigma_{ij}^{(l)} = \pm$ for $w_{ij}^{(l)} \neq 0$, $\Sigma_{ij}^{(l)} \in \{+, -\}$ for $w_{ij}^{(l)} = 0$. Then $y_{Lj}^{(l)}(w)$, $y_{Rj}^{(l)}(w)$, $l = 0, \ldots, r$, $j = 1, \ldots, m_l$, have partial derivatives in some Σ-neighborhood $U_\Sigma(w)$ of w, moreover, the partial derivatives are continuous in $U_\Sigma(a)$. Hence, $y_{Lj}^{(l)}(w)$, $y_{Rj}^{(l)}(w)$, are Σ-differentiable in w.*

Proof. Take a $U_\Sigma(w)$ which does not contain $b = \langle \ldots, b_{ij}^{(l)}, \ldots \rangle$ with $b_{ij}^{(l)} = 0$ and $w_{ij}^{(l)} \neq 0$. The theorem can again be proved by induction. One needs Theorem 2, a form of a theorem concerning Σ-differentiability of composite functions which can be proved (namely, $\frac{\partial^{\Sigma_i} f(g_1(x_1,x_2),g_2(x_1,x_2))}{\partial^{\Sigma_i} x_i} =$

$\frac{\partial f}{\partial y_1}\frac{\partial^{\Sigma_i} g_1}{\partial^{\Sigma_i} x_i} + \frac{\partial f}{\partial y_1}\frac{\partial^{\Sigma_i} g_1}{\partial^{\Sigma_i} x_i}$ for $f(y_1, y_2)$ differentiable, $g_i(x_1, x_2)$ Σ-differentiable) and standard arguments. The point is that for $w_{ij}^{(l)} > 0$ or $w_{ij}^{(l)} = 0$ and $\Sigma_{ij}^{(l)} = +$ ($w_{ij}^{(l)} < 0$ or $w_{ij}^{(l)} = 0$ and $\Sigma_{ij}^{(l)} = +$) we have $\frac{\partial^{\Sigma_{ij}^{(l)}} z_{Lj}^{(l)}(w)}{\partial^{\Sigma_{ij}^{(l)}} w_{ij}^{(l)}} = y_{Li}^{(l-1)}(w)$ $(= y_{Ri}^{(l-1)}(w))$. The Σ-differentiablility of $y_{Lj}^{(l)}(w)$ and $y_{Rj}^{(l)}(w)$ follows from Proposition 3. □

It immediately follows that for a differentiable function $E(x_{L1}, x_{R1}, \ldots, x_{Lm_r}, x_{Rm_r})$ and a given $w \in W$, the function $E(y_{L1}^{(r)}(w), y_{R1}^{(r)}(w), \ldots, y_{Lm_r}^{(r)}(w), y_{Rm_r}^{(r)}(w))$ is Σ-differentiable in w (with Σ as in Theorem 4).

Let $f(x_1, \ldots, x_n) : \mathcal{R}^n \mapsto \mathcal{R}$ have the derivatives $f'^{+}_u(a)$ in $a \in \mathcal{R}^n$ for every $u \in U \subseteq \mathcal{R}^n$. The vector $d \in U$ such that

$$f'^{+}_{\frac{d}{|d|}}(a) = \min\{f'^{+}_{\frac{u}{|u|}}(a);\, u \in U\}$$

is (if exists) called the *direction of the steepest descent in a with respect to U*. Here $f'^{+}_u(a) = \lim_{t\to 0^+} \frac{f(a+tu)-f(a)}{t}$. It can be shown that if f is Σ-differenyiable in a then $f'^{+}_{\frac{u}{|u|}}(a) = \frac{\partial^{\Sigma_1} f(a)}{\partial^{\Sigma_1} x_1}\frac{u_1}{|u|} + \cdots + \frac{\partial^{\Sigma_n} f(a)}{\partial^{\Sigma_n} x_n}\frac{u_n}{|u|}$. The following theorem provides an algorithm for determination of the steepest descent direction in our case.

Theorem 5 *Let $I \subseteq \{1, \ldots, n\}$, $f(x_1, \ldots, x_n) : \mathcal{R}^n \mapsto \mathcal{R}$ be Σ-differentiable in $a \in \mathcal{R}^n$ for each Σ with $\Sigma_i = \pm$ for $i \in I$, $\Sigma_i \in \{+, -\}$ for $i \notin I$. Let $d = \langle d_1, \ldots, d_n\rangle$ be a vector determined as follows:*

For $i \in I$, $d_i = -\frac{\partial f(a)}{\partial x_i}$.

For $i \notin I$, let

$$\begin{aligned}
&d_i = 0 \text{ for } \tfrac{\partial^- f(a)}{\partial^- x_i} \le 0 \text{ and } \tfrac{\partial^+ f(a)}{\partial^+ x_i} \ge 0,\\
&d_i = -\tfrac{\partial^- f(a)}{\partial^- x_i} \text{ for } \tfrac{\partial^- f(a)}{\partial^- x_i} > 0 \text{ and } \tfrac{\partial^+ f(a)}{\partial^+ x_i} \ge 0,\\
&d_i = -\tfrac{\partial^+ f(a)}{\partial^+ x_i} \text{ for } \tfrac{\partial^- f(a)}{\partial^- x_i} \le 0 \text{ and } \tfrac{\partial^+ f(a)}{\partial^+ x_i} > 0,\\
&d_i = -\tfrac{\partial^- f(a)}{\partial^- x_i} \text{ for } \tfrac{\partial^- f(a)}{\partial^- x_i} > 0 \text{ and } \tfrac{\partial^+ f(a)}{\partial^+ x_i} < 0 \text{ and } |\tfrac{\partial^- f(a)}{\partial^- x_i}| > |\tfrac{\partial^+ f(a)}{\partial^+ x_i}|,\\
&d_i = -\tfrac{\partial^+ f(a)}{\partial^+ x_i} \text{ for } \tfrac{\partial^- f(a)}{\partial^- x_i} > 0 \text{ and } \tfrac{\partial^+ f(a)}{\partial^+ x_i} < 0 \text{ and } |\tfrac{\partial^- f(a)}{\partial^- x_i}| < |\tfrac{\partial^+ f(a)}{\partial^+ x_i}|.
\end{aligned}$$

Then d is the direction of the steepest descent in a with respect to $\mathcal{R}^n$.

Proof. Take a signature Σ such that $\Sigma_i = \pm$ for $i \in I$, and for $i \notin I$ take $\Sigma_i = +$ for $d_i \ge 0$, $\Sigma_i = -$ for $d_i < 0$. It holds $d \in \mathcal{R}^\Sigma$. We first show that d is the direction of the steepest descent with respect to $\mathcal{R}^\Sigma$. Consequently, we show that for any other signature Σ', d is better than any other $u \in \mathcal{R}^{\Sigma'}$.

For $u \in \mathcal{R}^\Sigma$ we have $f'^+_{\frac{u}{|u|}} = \frac{\partial^{\Sigma_1} f(a)}{\partial^{\Sigma_1} x_1}\frac{u_1}{|u|} + \cdots + \frac{\partial^{\Sigma_n} f(a)}{\partial^{\Sigma_n} x_n}\frac{u_n}{|u|}$ which is the scalar product $grad^\Sigma f(a) \cdot \frac{u}{|u|}$. From $|\frac{u}{|u|}| = 1$ we have $grad^\Sigma f(a) \cdot \frac{u}{|u|} = \cos\alpha \cdot |grad^\Sigma f(a)|$ where α is the size of the angle between $grad^\Sigma f(a)$ and $\frac{u}{|u|}$, and clearly, it is minimal for $\alpha = \pi$, i.e. for $u = -grad^\Sigma f(a)$. Denote $J = \{i_1, \ldots, i_k\} = \{i;\ \frac{\partial^{\Sigma_i} f(a)}{\partial^{\Sigma_i} x_i} \neq 0, d_i = 0\}$. If $J = \emptyset$ then $-grad^\Sigma f(a) = d$ which means d is the direction of the steepest descent.

Suppose now $J \neq \emptyset$ (hence $\Sigma_i = +$ and $\frac{\partial^{\Sigma_i} f(a)}{\partial^{\Sigma_i} x_i} > 0$ by definition of Σ for $i \in J$). Let $u \in \mathcal{R}^\Sigma$, i.e. $u_i \geq 0$ for $i \in J$. We have to show $grad^\Sigma f(a) \cdot \frac{u}{|u|} \geq grad^\Sigma f(a) \cdot \frac{d}{|d|}$. By the same arguments as above we can show that d is the direction of the steepest descent with respect to $V = \{\langle v_1, \ldots, v_n\rangle \in \mathcal{R}^\Sigma;\ v_i = 0 \text{ for } i \in J\}$ (just consider the restriction of f to the variables x_i, $i \notin J$). It is clear that $\sum_{i\in J} \frac{\partial^{\Sigma_i} f(a)}{\partial^{\Sigma_i} x_i} u_i \geq 0$, hence $grad^\Sigma f(a) \cdot u^* \leq grad^\Sigma f(a) \cdot u$ for u^* such that $u^*_i = 0$ for $i \in J$ and $u^*_i = u_i$ for $i \notin J$. Now, if $grad^\Sigma f(a) \cdot \frac{u}{|u|} < grad^\Sigma f(a) \cdot \frac{d}{|d|}$ then also $grad^\Sigma f(a) \cdot \frac{u^*}{|u|} < grad^\Sigma f(a) \cdot \frac{d}{|d|}$. From $|u^*| \leq |u|$ it follows that $grad^\Sigma f(a) \cdot \frac{u^*}{|u^*|} < grad^\Sigma f(a) \cdot \frac{u^*}{|u|} < grad^\Sigma f(a) \cdot \frac{d}{|d|}$. But $u^* \in V$, a contradiction with the fact that d is the direction of the steepest descent with respect to V.

Take now another signature Σ' which differs from Σ at most in Σ'_i such that $\Sigma_i \neq \pm$. Denote $J = \{i;\ \Sigma'_i \neq \Sigma_i\}$. We may again prove that for d' determined by $d'_i = d_i$ for $i \notin J$, $d'_i = -\frac{\partial^{\Sigma'_i} f(a)}{\partial^{\Sigma'_i} x_i}$ for $i \in J$, $\frac{\partial^- f(a)}{\partial^- x_i} > 0$, $\frac{\partial^+ f(a)}{\partial^+ x_i} < 0$, and $d'_i = 0$ otherwise, d' is the direction of the steepest descent with respect to $\mathcal{R}^{\Sigma'}$. But now, $|d_i| \geq |d'_i|$, $i = 1, \ldots, n$. We have $f'^+_{\frac{d}{|d|}}(a) = grad^\Sigma f(a)\frac{d}{|d|} = -\frac{d\cdot d}{|d|} = -|d|$ $\leq -|d'| = -\frac{d\cdot d'}{|d'|} = grad^{\Sigma'} f(a)\frac{d'}{|d'|} = f'^+_{\frac{d'}{|d'|}}(a)$. The signature Σ' was chosen arbitrarily, hence d is the direction of the steepest descent with respect to $\mathcal{R}$. □

The foregoing theorem assures that if f satisfies the stated conditions the direction of the steepest descent can be relatively easily found. In the worst case we have to compute both of the one-side derivatives. Hence, adapting our net by the steepest descent method is as meaningful as the backpropagation algorithm.

Theorems 4 and 5 can be directly applied to our function E^p to get the direction of the steepest descent in any point $w \in W$. It remains only to derive formulas for respective derivatives. For simplicity we omit the upper indices denoting patterns, i.e. we write e.g. E instead of E^p. Similarly, for a given signature Σ we write ∂ instead of $\partial^{\Sigma^{(l)}_{ij}}$, thus, e.g., $\frac{\partial E}{\partial w^{(l)}_{ij}}$ instead of $\frac{\partial^{\Sigma^{(l)}_{ij}} E}{\partial^{\Sigma^{(l)}_{ij}} w^{(l)}_{ij}}$.

We have

$$\frac{\partial E}{\partial w^{(l)}_{ij}} = \frac{\partial E_L}{\partial w^{(l)}_{ij}} + \frac{\partial E_R}{\partial w^{(l)}_{ij}}.$$

Further we proceed for $\frac{\partial E_L}{\partial w^{(l)}_{ij}}$ only. The formulas for $\frac{\partial E_R}{\partial w^{(l)}_{ij}}$ can be obtained in a similar manner.

$$\frac{\partial E_L}{\partial w^{(l)}_{ij}} = \frac{\partial E_L}{\partial y^{(l)}_{Lj}}\frac{\partial y^{(l)}_{Lj}}{\partial z^{(l)}_{Lj}}\frac{\partial z^{(l)}_{Lj}}{\partial w^{(l)}_{ij}} + \frac{\partial E_L}{\partial y^{(l)}_{Rj}}\frac{\partial y^{(l)}_{Rj}}{\partial z^{(l)}_{Rj}}\frac{\partial z^{(l)}_{Rj}}{\partial w^{(l)}_{ij}}$$

where

$$\frac{\partial y_{Lj}^{(l)}}{\partial z_{Lj}^{(l)}} = y_{Lj}^{(l)}(1 - y_{Lj}^{(l)}) \quad \text{and} \quad \frac{\partial y_{Rj}^{(l)}}{\partial z_{Rj}^{(l)}} = y_{Rj}^{(l)}(1 - y_{Rj}^{(l)}),$$

and

$$\frac{\partial z_{Lj}^{(l)}}{\partial w_{ij}^{(l)}} = \begin{cases} y_{Li}^{(l-1)} \text{ for } w_{ij}^{(l)} > 0 \text{ or } \Sigma_{ij}^{(l)} = + \\ y_{Ri}^{(l-1)} \text{ for } w_{ij}^{(l)} < 0 \text{ or } \Sigma_{ij}^{(l)} = - \end{cases} \quad \text{and} \quad \frac{\partial z_{Rj}^{(l)}}{\partial w_{ij}^{(l)}} = \begin{cases} y_{Ri}^{(l-1)} \text{ for } w_{ij}^{(l)} > 0 \text{ or } \Sigma_{ij}^{(l)} = + \\ y_{Li}^{(l-1)} \text{ for } w_{ij}^{(l)} < 0 \text{ or } \Sigma_{ij}^{(l)} = - \end{cases}$$

Now, for the output layer we have ($l = r$)

$$\frac{\partial E_L}{\partial y_{Lj}^{(l)}} = y_{Lj}^{(l)} - o_{Lj}, \qquad \frac{\partial E_L}{\partial y_{Rj}^{(l)}} = 0,$$

whereas for the inner layer ($l < r$)

$$\frac{\partial E_L}{\partial y_{Lj}^{(l)}} = \sum_{k=1}^{m_{(l+1)}} \left(\frac{\partial E_L}{\partial y_{Lk}^{(l+1)}} \frac{\partial y_{Lk}^{(l+1)}}{\partial z_{Lk}^{(l+1)}} \frac{\partial z_{Lk}^{(l+1)}}{\partial y_{Lj}^{(l)}} + \frac{\partial E_L}{\partial y_{Rk}^{(l+1)}} \frac{\partial y_{Rk}^{(l+1)}}{\partial z_{Rk}^{(l+1)}} \frac{\partial z_{Rk}^{(l+1)}}{\partial y_{Lj}^{(l)}}\right)$$

holds. We have

$$\frac{\partial z_{Lk}^{(l+1)}}{\partial y_{Lj}^{(l)}} = \begin{cases} w_{jk}^{(l+1)} & \text{for } w_{jk}^{(l+1)} > 0 \\ 0 & \text{else} \end{cases} \quad \text{and} \quad \frac{\partial z_{Rk}^{(l+1)}}{\partial y_{Lj}^{(l)}} = \begin{cases} w_{jk}^{(l+1)} & \text{for } w_{jk}^{(l+1)} < 0 \\ 0 & \text{else} \end{cases}$$

Similar computations can be done for $\frac{\partial E_L}{\partial y_{Rj}^{(l)}}$.

To sum up, denote

$$\delta_{Lj}^{L,(l)} = \frac{\partial E_L}{\partial y_{Lj}^{(l)}} y_{Lj}^{(l)}(1 - y_{Lj}^{(l)}), \qquad \delta_{Rj}^{L,(l)} = \frac{\partial E_L}{\partial y_{Rj}^{(l)}} y_{Rj}^{(l)}(1 - y_{Rj}^{(l)}),$$
$$\delta_{Lj}^{R,(l)} = \frac{\partial E_R}{\partial y_{Lj}^{(l)}} y_{Lj}^{(l)}(1 - y_{Lj}^{(l)}), \qquad \delta_{Rj}^{R,(l)} = \frac{\partial E_R}{\partial y_{Rj}^{(l)}} y_{Rj}^{(l)}(1 - y_{Rj}^{(l)}).$$

Then we have

$$\frac{\partial E_L}{\partial w_{ij}^{(l)}} = \delta_{Lj}^{L,(l)} y_{\square i}^{(l-1)} + \delta_{Rj}^{L,(l)} y_{\diamond i}^{(l-1)} \quad \text{and} \quad \frac{\partial E_R}{\partial w_{ij}^{(l)}} = \delta_{Lj}^{R,(l)} y_{\square i}^{(l-1)} + \delta_{Rj}^{R,(l)} y_{\diamond i}^{(l-1)}$$

where

$$\square = \begin{cases} L \text{ for } w_{ij}^{(l)} > 0 \text{ or } \Sigma_{ij}^{(l)} = + \\ R \text{ for } w_{ij}^{(l)} < 0 \text{ or } \Sigma_{ij}^{(l)} = - \end{cases}, \qquad \diamond = \begin{cases} R \text{ for } w_{ij}^{(l)} > 0 \text{ or } \Sigma_{ij}^{(l)} = + \\ L \text{ for } w_{ij}^{(l)} < 0 \text{ or } \Sigma_{ij}^{(l)} = - \end{cases}$$

where for output layer ($l = r$) we have

$$\delta_{Lj}^{L,(l)} = (y_{Lj}^{(l)} - o_{Lj}) y_{Lj}^{(l)}(1 - y_{Lj}^{(l)}), \quad \delta_{Rj}^{L,(l)} = 0, \quad \delta_{Lj}^{R,(l)} = 0, \quad \delta_{Rj}^{R,(l)} = (y_{Rj}^{(l)} - o_{Rj}) y_{Rj}^{(l)}(1 - y_{Rj}^{(l)}),$$

and for inner layer ($l < r$) we have

$$\delta_{Lj}^{L,(l)} = \sum_{k=1}^{m_{l+1}} (\delta_{Lk}^{L,(l+1)} sgn^{+}(w_{jk}^{(l+1)}) + \delta_{Rk}^{L,(l+1)} sgn^{-}(w_{jk}^{(l+1)})) y_{Lj}^{(l)}(1 - y_{Lj}^{(l)})$$

$$\delta_{Rj}^{L,(l)} = \sum_{k=1}^{m_{l+1}} (\delta_{Lk}^{L,(l+1)} sgn^{-}(w_{jk}^{(l+1)}) + \delta_{Rk}^{L,(l+1)} sgn^{+}(w_{jk}^{(l+1)})) y_{Rj}^{(l)}(1 - y_{Rj}^{(l)})$$

$$\delta_{Lj}^{R,(l)} = \sum_{k=1}^{m_{l+1}} (\delta_{Lk}^{R,(l+1)} sgn^{+}(w_{jk}^{(l+1)}) + \delta_{Rk}^{R,(l+1)} sgn^{-}(w_{jk}^{(l+1)})) y_{Lj}^{(l)}(1 - y_{Lj}^{(l)})$$

$$\delta_{Rj}^{R,(l)} = \sum_{k=1}^{m_{l+1}} (\delta_{Lk}^{R,(l+1)} sgn^{-}(w_{jk}^{(l+1)}) + \delta_{Rk}^{R,(l+1)} sgn^{+}(w_{jk}^{(l+1)})) y_{Rj}^{(l)}(1 - y_{Rj}^{(l)}).$$

4 Experimental results

It follows from our experience with the network that adapting to interval training set takes usually more time than adapting to "crisp" training set (by approximately equal numbers of patterns). This is not surprising because intervals carry more information than "crisp" numbers.

On the other hand, even in the case of adapting to "crisp" training set it appears to be convenient if we express our knowledge (training set) in interval terms, e.g. by some method of cluster analysis, roughly adapt the net to the interval training set, and "tune" the net afterwards by adapting to the original crisp training set.

An advantageous property of neural networks documented experimentally is the generalization ability, i.e. "smart" interpolation capability. This property is sustained in our model. We document it by the following example. A network with four hidden neurons has been adapted to the following training set (a form of linear transform has been used for outputs not to be restricted to outputs from $(0, 1)$):

$$\begin{aligned} T \;=\; & \{\langle[-1,1],[-1,1],[-2,2]\rangle, \langle[-1,1],[19,21],[18,22]\rangle, \\ & \langle[9,11],[-1,1],[8,12]\rangle, \langle[9,11],[19,21],[28,32]\rangle\} \ . \end{aligned}$$

The training set represents the interval operation $+$. When adapted, the net was presented the intervals $[4, 6]$ and $[9, 11]$ as inputs. These inputs lie in between the inputs contained in the training set. The interval $[12.2, 16.5]$ has been generated by the net as the corresponding output. Also from other experiments it follows that the quality of generalization capabilities of our model is comparable to that one of classical (crisp) model.

5 Conclusion

We have presented a neural network mapping tupples of intervals to tupples of intervals. The network has been obtained applying interval arithmetic to standard multilayer feedforward neural

net with sigmoidal transfer functions. The approach differs from that one described in [8, 9] in the strict sticking to the interval arithmetic. An important property of the network is the monotonicity. A steepest descent like learning algorithm for the network which makes it possible to learn both from interval and "crisp" data simultaneously has been developed. The algorithm is theoretically well justified. The obtained results are of general nature and are well suitable for design of interval networks induced by other "crisp" neural network architectures.

Concerning possible applications, the network works with very simple kind of indeterminacy—intervals. This allows to express the imprecise knowledge in the form of a training set. Another promising field of application (mentioned in [9]) is the possibility of speeding up the adaptation process to "crisp" training set by rough adapting to interval training set and consecutive "tuning" to the "crisp" training set, see also the previous section.

A promising field of application which is in our opinion the main contribution of our model is the field of approximate reasoning (AR) and the role of neural networks in it. AR aims at methods of modeling of human reasoning. The characteristic property of human reasoning is the effective dealing with the vagueness phenomenon. In the frame of AR, vagueness is modeled mostly by means of fuzzy sets (see e.g. [4, 5] for details)—undoubtedly more adequate model than intervals. Having a linguistic description of some real process in the form of the so called IF-THEN rules (an IF-THEN rule is an expression of the form $\mathsf{IF}\ X\ \mathsf{is}\ \mathcal{A}\ \mathsf{THEN}\ Y\ \mathsf{is}\ \mathcal{B}$, X, Y denoting some state-variables, $\mathcal{A}, \mathcal{B}$ some natural language expressions meaning of which can be modeled by appropriate fuzzy sets) at disposal, the goal is to "deduce" a fuzzy set B' representing a state of Y corresponding to a given observed state of X modeled by a fuzzy set A'. One may want to have a neural network model accomplishing this task. This can be done by the so called extension principle (EP) allowing us to extend the function performed by feedforward neural net to a function mapping fuzzy sets to fuzzy sets. Furthermore, by the so called representation theorem, each fuzzy set can be represented as a collection $\{{}^{\alpha}A;\ \alpha \in K \subseteq [0,1]\}$ of nested (i.e. ${}^{\alpha}A \subseteq {}^{\beta}A$ for $\beta \leq \alpha$) cuts ${}^{\alpha}A$ which are in the most common cases real intervals. In the case of continuous function, the function obtained by EP can be computed cut by cut. Hence, the problem of fuzzifying the standard feedforward neural net reduces to the area of interval networks. However, the condition requiring nested intervals disqualifies the CS interval network which is not monotonic, i.e., as outputs we could obtain collection of nonnested intervals which could not be interpreted as a fuzzy set. The field of fuzzy neural networks in the sense above will be one of the topics of our future research ([2]). In our opinion, it could shift the paradigm of neural nets closer to the idea of adaptive distributed systems capable of processing of indeterminate information.

Needless to say, adaptation procedure has not to be restricted to the most popular backpropagation-like approach. Other methods used in classical neural nets adaptation like genetic algorithms or combination of various approaches can be used.

References

[1] Arbib M.(Ed.): *The Handbook of Brain Theory and Neural Networks.* MIT Press, London, 1995.

[2] Bělohlávek R.: *Neural Networks Processing Indeterminacy.* Prepared doctoral thesis.

[3] Hecht-Nielsen R.: *Neurocomputing*, Addison-Wesley, 1990.

[4] Klir G. J., Yuan B.: *Fuzzy Sets and Fuzzy Logic. Theory and Applications.* Prentice Hall, 1995.

[5] Kruse R., Gebhardt J., Klawonn F.: *Fuzzy-Systeme.* B. G. Teubner, Stuttgart, 1995.

[6] Kufudaki O., Hořejš J.: PAB: Parameters adapting back-propagation, *Neural Network World* **1**(1991), 267-274.

[7] Rumelhart D. E., Hinton G. E., Williams R. J.: Learning representations by back-propagation errors, *Nature* **323**(1986), 533-536.

[8] Šíma J.: The multi-layered neural network as an adaptive expert system with the ability to work with incomplete information and to provide justification of inference, *Neural Network World* **2**(1992), 47-58.

[9] Šíma J.: Generalized back propagation for interval training patterns, *Neural Network World* **2**(1992), 167-173.