A fuzzy backpropagation algorithm

Stefka Stoeva
Bulgarian National Library, V. Levski 88, BG-1504 Sofia, Bulgaria

Alexander Nikov
Technical University of Sofia, K. Ohridski 8, BG-1000 Sofia, Bulgaria

Abstract: This paper presents an extension of the standard backpropagation algorithm (SBP). The proposed learning algorithm is based on the fuzzy integral of Sugeno and thus called fuzzy backpropagation (FBP) algorithm. Necessary and sufficient conditions for convergence of FBP algorithm for single output networks in case of single and multiple training patterns are proved. A computer simulation illustrates and confirms the theoretical results. FBP algorithm shows considerably greater convergence rate in comparison with SBP algorithm. Other advantages of FBP algorithm are that it reaches forward to the target value without oscillations, requires no assumptions about probability distribution and independence of input data. The convergence conditions enable training by automation of weights tuning process (quasi-unsupervised learning) pointing out the interval where the target value belongs to. This supports acquisition of implicit knowledge and ensures wide application, e.g. for creation of adaptable user interfaces, assessment of products, intelligent data analysis, etc.

Keywords: Neural networks, learning algorithm, fuzzy logic, multicriteria analysis.

1. Introduction

Recently the interest in neural networks has grown dramatically: it is expected that neural network models will be useful both as models of real brain functions and as computational devices. One of the most popular neural networks is the layered feedforward neural network with a backpropagation (BP) least mean square learning algorithm[17]. Its topology is shown in Figure 1. The network edges connect the processing units called neurons. With each neuron input there is associated a weight, representing its relative importance in the set of the neuron’s inputs. The inputs’ values to each neuron are accumulated through the net function to yield the net value: the net value is a weighted linear combination of the neuron’s inputs’ values.

For the purpose of multicriteria analysis[13][20][21] a hierarchy of criteria is used to determine an overall pattern evaluation. The hierarchy can be encoded into a hierarchical neural network where each neuron corresponds to a criterion. The input neurons of the network correspond to single criteria. The hidden and output neurons correspond to complex criteria. As evaluation function it can be used the net function of the neurons. However the criteria can be combined linearly when it is assumed that they are independent. But in practice the criteria are correlated to some degree. The linear evaluation function is unable to capture...
relationship between the criteria. In order to overcome this drawback of standard backpropagation (SBP) algorithm we propose a fuzzy extension called fuzzy backpropagation (FBP) algorithm. It determines the net value through the fuzzy integral of Sugeno [3] and thus does not assume independence between the criteria. Another advantage of FBP algorithm is that it reaches always forward to the target value without oscillations and there is no possibility to fall into local minimum. Necessary and sufficient conditions for convergence of FBP algorithm for single output networks in case of single and multiple training patterns are proved. The results of computer simulation are reported and analysed: FBP algorithm shows considerably greater convergence rate in comparison with SBP algorithm.

2. Description of the algorithm

2.1 Standard backpropagation algorithm (SBP algorithm)

First we describe the standard backpropagation algorithm [17] [18] because FBP algorithm proposed by us can be viewed as fuzzy-logic-based extension of the standard one. SBP algorithm is an integrative gradient algorithm designed to minimise the mean square error between the actual output and the desired output by modifying network weights. In the following we use for simplicity a network with one hidden layer but the results are valid also for networks with more than one hidden layers. Let us consider a layered feedforward neural network with \( n_0 \) inputs, \( n_1 \) hidden neurons and single output neuron (cf. Figure 1). The input-output relation of each neuron of the neural network is defined as follows:

**Hidden neurons**

\[
net_i^{(1)} = \sum_{j=1}^{n_1} w_{ij}^{(1)} x_j,
\]

\[
a_i^{(1)} = f\left(net_i^{(1)}\right), \quad i = 1,2,\ldots,n_1
\]

**Output neurons**

\[
net^{(2)} = \sum_{i=1}^{n_1} w_{ji}^{(2)} a_i^{(1)},
\]

\[
a^{(2)} = f\left(net^{(2)}\right),
\]

where \( X = (x_1,x_2,\ldots,x_{n_0}) \) is the vector of pattern’s inputs.

The activation functions \( f\left(net\right) \) can be linear ones.

Analogous to the writing and reading phases, there are also two phases in the supervised learning BP network. There is a learning (training) phase when a training data set is used to determine the weights that define the neural model. So the objective of the BP algorithm is to find the optimal weights to minimise the error between the target value and the actual response. Then the trained neural model will be used later in the retrieving phase to process and evaluate real patterns.

Let the pattern’s output corresponding to the vector of pattern’s inputs \( X \) be called the target value \( t \). Then the learning of the neural network for training pattern \( (X,t) \) is performed in order to minimise the squared error between the target and the actual response:

\[
E = \frac{1}{2} (t - a^{(2)})^2.
\]

The weights are changed according to the following formula:

\[
w_{ji}^{new} = w_{ij}^{old} + \Delta w_{ij}
\]

where

\[
\Delta w_{ij} = \eta \delta_j^{(1)} a_j^{(1)}.
\]

\( \eta \in [0,1] \) denotes the learning rate, \( \delta_j^{(1)} \) is the error signal relevant to the i-th neuron, and \( a_j^{(1)} \) the signal at j-th neuron input.

The error signal is obtained recursively by back propagation.

Output layer

\[
\delta^{(2)} = f'\left(net^{(2)}\right)(t - a^{(2)}),
\]

where \( f'(net) \) is the derivation of the activation function.
Hidden layer

\[ \delta_i^{(1)} = f'(net_i^{(1)}) \delta_i^{(2)} w_{li}^{(2)}, \]

where \( \delta_i^{(2)} \) is determined by formula (1).

When there are more than one training pattern \((X^m, t^m)\), \(m = 1, 2, \ldots, M\), the sum-squared error is defined as

\[ E = \frac{1}{2} \sum_{m=1}^{M} (t^m - a_i^{(2)})^2. \]

### 2.2 Fuzzy backpropagation algorithm (FBP algorithm)

Recently many neuro-fuzzy models have been introduced [2], [6], [8], [10], [11]. The following extension of the standard BP algorithm to fuzzy BP algorithm is proposed. For yielding the net value \( net_i \) the inputs’ values of the \( i \)-th neuron are aggregated. The mapping is mathematically described by the fuzzy integral of Sugeno that relies on psychological background.

Let \( x_j \in [0,1] \subseteq \mathbb{R}, \quad j = 1, \ldots, n_0 \), and all network weights \( w_{ij}^{(l)} \in [0,1] \subseteq \mathbb{R} \), where \( l = 1, 2 \). Let \( P(J) \) be the power set of the set \( J \) of the network inputs (indices), where \( J_j \) is the \( j \)-th input of the \( i \)-th neuron.

Further on for simplicity network inputs with their indices are identified. Let \( g: P(J) \rightarrow [0,1] \) be a function defined as follows:

\[ g(\emptyset) = 0 \]

\[ g\left( \{ J_{j_1} \} \right) = w_{ij}^{(l)}, \quad 1 \leq j \leq n_0 \]

\[ g\left( \{ J_{j_1}, J_{j_2}, \ldots, J_{j_r} \} \right) = w_{ij_1}^{(l)} \lor w_{ij_2}^{(l)} \lor \ldots \lor w_{ij_r}^{(l)}, \quad \text{where} \{ j_1, j_2, \ldots, j_r \} \subseteq \{1, 2, \ldots, n_0 \} \]

\[ g\left( \{ J_{j_1}, J_{j_2}, \ldots, J_{j_r}, \ldots, J_{j_{n_0}} \} \right) = 1, \]

where the notations \( \lor \) and \( \land \) stand for the operations MAX and MIN respectively in the unit real interval \([0,1] \subseteq \mathbb{R}\). It is proved [20] that the function \( g \) is a fuzzy measure on \( P(J) \). Therefore the functional assumes the form of the Sugeno integral over the finite reference set \( J \).

Let \( h: P(I) \rightarrow [0,1] \) be a function, defined in a similar way.

**Hidden neurons**

\[ net_i^{(1)} = \lor_{G \subseteq P(J)} \left[ \land_{p \in G} x_p \right] \land g(G) \]

\[ a_i^{(1)} = f\left( net_i^{(1)} \right), \quad i = 1, 2, \ldots, n_1 \]

**Output neurons**

\[ net^{(2)} = \lor_{G \subseteq P(I)} \left[ \land_{p \in G} a_p^{(1)} \right] \land h(G) \]

\[ a^{(2)} = f\left( net^{(2)} \right) \]

The activation functions \( f(\text{net}) \) are linear ones.

Therefore the activation values \( a_l^{(i)} \in [0,1] \), where \( l = 1, 2 \).

The error signal is obtained by back propagation.
Output layer
\[ \delta^{(2)} = \| y - a^{(2)} \|, \]  
(2)

Hidden layer
\[ \delta^{(i)} = \| y - a^{(i)} \|, \quad i = 1, \ldots, n_1. \]  
(3)

In order to obtain \( w_{ij}^{new} \in [0,1] \) the weights are handled as follows:

**Case \( t = a^{(2)} \):** no adjustment of the weights is necessary

**Case \( t > a^{(2)} \):**
- If \( w_{ij}^{old} < t \) then \( w_{ij}^{new} = 1 \wedge (w_{ij}^{old} + w_y) \)
- If \( w_{ij}^{old} \geq t \) then \( w_{ij}^{new} = w_{ij}^{old} \)

**Case \( t < a^{(2)} \):**
- If \( w_{ij}^{old} > t \) then \( w_{ij}^{new} = 0 \lor (w_{ij}^{old} - w_y) \)
- If \( w_{ij}^{old} \leq t \) then \( w_{ij}^{new} = w_{ij}^{old} \),

where \( \Delta w_{ij} = \eta \delta_i \) is determined by formula (2) resp. (3) and \( \eta \) is the learning rate.

In the following let us denote the iteration steps of the proposed algorithm by \( s \), the corresponding weights by \( w_{ij}^{(l)}(s) \) and the activation values by \( a_i^{(l)}(s) \).

2.3 Necessary and sufficient conditions for convergence of FBP algorithm for single output networks

The following nontrivial necessary and sufficient conditions for convergence of FBP algorithm in case of single output neural networks will be formulated and proved: the interval between the minimum and maximum of the pattern’s inputs shall include the pattern’s output, i.e. the target value.

2.3.1 Convergence condition in case of single training pattern

**Definition 1.** FBP algorithm is convergent to the target value \( t \) if there exists a number \( s_0 \in \mathbb{N}^+ \) such that the following equalities hold:

\[ t = a^{(2)}(s_0) = a^{(2)}(s_0 + 1) = a^{(2)}(s_0 + 2) = \ldots, \]
\[ w_{ij}^{(l)}(s_0) = w_{ij}^{(l)}(s_0 + 1) = w_{ij}^{(l)}(s_0 + 2) = \ldots, \quad l = 1,2. \]

The following lemma concerns the outputs of the neurons at the hidden layer.

**Lemma 1.**

(i) If \( a_i^{(1)}(s) > t \), then \( a_i^{(1)}(s + 1) \geq t \).

(ii) If \( a_i^{(1)}(s) > t \), then \( a_i^{(1)}(s + 1) \leq t \).

The proof of this lemma is similar to that one of the lemma in [19].

The following lemma concerns the output of the output neuron.
Lemma 2.

(i) If $a^{(2)}(s) > t$, then $a^{(2)}(s + 1) \geq t$.

(ii) If $a^{(2)}(s) < t$, then $a^{(2)}(s + 1) \leq t$.

Proof. In the sequel for each set $G \in P(I)$

$$h_s(G) = \bigvee_{p \in G} w^{(2)}_{1p}(s)$$

$$v_s(G) = \bigwedge_{p \in G} a^{(1)}(s)$$

$$a^{(2)}(s) = \bigvee_{G \in P(I)} [v_s(G) \wedge h_s(G)]$$

(i) Since $a^{(2)}(s) > t$, and $P(I)$ is a finite set, there exists a set $G_s \in P(I)$ such that

$$v_s(G_s) \wedge h_s(G_s) = a^{(2)}(s) > t$$

i.e.

$$v_s(G_s) \geq a^{(2)}(s) > t \quad \text{and} \quad h_s(G_s) \geq a^{(2)}(s) > t$$

By Lemma 1. it follows that $v_{s+1}(G_s) \geq t$. Since $G_s$ is a finite set, there exists an input $p_s \in G_s$, such that $w^{(2)}_{1p_s}(s) = h_s(G_s)$. Therefore, the weight $w^{(2)}_{1p_s}(s + 1)$ is produced through the formula:

$$w^{(2)}_{1p_s}(s + 1) = 0 \lor \left[w^{(2)}_{1p_s}(s) - \eta \lceil t - a^{(2)}(s) \rceil\right]$$

$$\geq 0 \lor \left[w^{(2)}_{1p_s}(s) - \lceil t - a^{(2)}(s) \rceil\right]$$

$$= 0 \lor \left(w^{(2)}_{1p_s}(s) - a^{(2)}(s) + t\right)$$

$$\geq 0 \lor \left(a^{(2)}(s) - a^{(2)}(s) + t\right)$$

$$= 0 \lor t$$

$$= t$$

So, $v_{s+1}(G_s) \wedge w^{(2)}_{1p_s}(s + 1) \geq t$. Let

$$a^{(2)}(s + 1) = v_{s+1}(G_{s+1}) \wedge h_{s+1}(G_{s+1})$$

for some set $G_{s+1} \in P(I)$. Then

$$a^{(2)}(s + 1) = v_{s+1}(G_{s+1}) \wedge h_{s+1}(G_{s+1}) \geq v_{s+1}(G_s) \wedge w^{(2)}_{1p_s}(s + 1) \geq t$$

Thus, $a^{(2)}(s + 1) \geq t$.

(ii) Since $a^{(2)}(s) < t$, and $P(I)$ is a finite set, there exists a set $G_s \in P(I)$ such that

$$v_s(G_s) \wedge h_s(G_s) = a^{(2)}(s) < t$$

Let

$$a^{(2)}(s + 1) = v_{s+1}(G_{s+1}) \wedge h_{s+1}(G_{s+1})$$

for some set $G_{s+1} \in P(I)$. Since $G_{s+1}$ is a finite set, there exists an input $p_{s+1} \in G_{s+1}$, such that

$$w^{(2)}_{1p_{s+1}}(s + 1) = h_{s+1}(G_{s+1})$$
Then it holds:
\[ t > a^{(2)}(s) = v_s(G_s) \land h_s(G_s) \]
\[ \geq v_s(G_{s+1}) \land w_{i_p,s+1}^{(2)}(s). \]

If \( v_s(G_{s+1}) \leq a^{(2)}(s) < t \), then by Lemma 1. it follows that \( v_{s+1}(G_{s+1}) \leq t \), and thus \( a^{(2)}(s+1) < t \).

If \( w_{i_p,s+1}^{(2)}(s) \leq a^{(2)}(s) < t \), the weight \( w_{i_p,s+1}^{(2)}(s+1) \) is produced through the formula:
\[
\begin{align*}
   w_{i_p,s+1}^{(2)}(s+1) &= 1 \land \left( w_{i_p,s+1}^{(2)}(s) + [t - a^{(2)}(s)] \right) \\
   &\leq 1 \land \left( w_{i_p,s+1}^{(2)}(s) + [t - a^{(2)}(s)] \right) \\
   &= 1 \land \left( w_{i_p,s+1}^{(2)}(s) + t - a^{(2)}(s) \right) \\
   &= 1 \land t \\
   &= t
\end{align*}
\]

Hence \( a^{(2)}(s+1) \leq t \).

**Theorem 1.** FBP algorithm is convergent to the target value \( t \) iff for the neurons at the input layer the following condition holds:
\[
\exists j' \exists j' \exists a_j^{(0)} \geq t \land a_{j'}^{(0)} \leq t. \tag{4}
\]

**Proof.** The ‘if’ part of Theorem 1. is proved by assuming that condition (4) is not satisfied.

Suppose there is no input \( j' \in J \) such that \( a_{j'}^{(0)} \geq t \). Then for each input \( j \in J \), \( 1 \leq j \leq n_0 \), it holds that \( a_j^{(0)} < t \). Hence for each set \( G \in P(J) \) it holds that
\[ v(G) = \land_{p \in G} a_p^{(0)} < t. \]

So, at each iteration step \( s \) it holds that
\[
\begin{align*}
   a_j^{(1)}(s) &= \lor_{G \in P(J)} [v(G) \land g_a(G)] \\
   &< t \land \lor_{G \in P(J)} g_a(G) \\
   &= t \land 1 = t
\end{align*}
\]

for each \( i, 1 \leq i \leq n_1 \), where \( g_a(G) = \lor_{p \in G} w_p^{(1)}(s) \). Therefore, at each iteration step \( s \) it holds that
\[
\begin{align*}
   a_j^{(2)}(s) &= \lor_{G \in P(J)} [v_s(G) \land h_s(G)] \\
   &= \lor_{G \in P(J)} \left[ \lor_{p \in G} a_p^{(1)}(s) \land h_s(G) \right] \\
   &= \lor_{G \in P(J)} \left[ t \land h_s(G) \right] \\
   &= t \land \lor_{G \in P(J)} h_s(G) \\
   &= t \land 1 = t.
\end{align*}
\]

Thus, in such case FBP algorithm is not convergent to the target value \( t \).
The assumption that there is no input \( j^* \in J \), such that \( a_{j^*}^{(0)} \leq t \), implies in a similar way that at each iteration step \( s \) \( a_{j}^{(2)}(s) > t \) holds. Thus, in this case FBP algorithm is not convergent to the target value \( t \), either.

The "only if" part of Theorem 1. is proved by assuming that FBP algorithm is not convergent to the target value \( t \). Hence, there exists no iteration step \( s_0 \in N^+ \), such that \( a_{j}^{(2)}(s_0) = t \), i.e. at each iteration step \( s \) it holds either \( a_{j}^{(2)}(s) > t \) or \( a_{j}^{(2)}(s) < t \).

If \( a_{j}^{(2)}(1) > t \) is fulfilled, then at each iteration step \( s \), \( a_{j}^{(2)}(s) > t \) is fulfilled, too. The proof is performed through mathematical induction. Let \( a_{j}^{(2)}(1) > t \) be fulfilled indeed. Since FBP algorithm is not convergent, by Lemma 2. it follows that \( a_{j}^{(2)}(2) > t \). Let now at the \( s \)-th iteration step \( a_{j}^{(2)}(s) > t \) be fulfilled. Since FBP algorithm is not convergent, by Lemma 2. it follows that \( a_{j}^{(2)}(s+1) > t \). According to the mathematical induction axiom at each iteration step \( s \), \( a_{j}^{(2)}(s) > t \) is satisfied.

Since \( I \) is a finite set, at each iteration step \( s \) there exists a set \( G_s \in P(I) \) such that

\[
v_s(G_s) \land h_s(G_s) = a_{j}^{(2)}(s) > t,\]

i.e. \( v_s(G_s) > t \) and \( h_s(G_s) > t \).

Let \( s_0 \in N^+ \) be the iteration step, at which the process of adjusting all the weights of the network comes to an end. Then the above inequalities can be fulfilled only if \( h_{s_0}(G_{s_0}) = 1 \), i.e. if set \( G_{s_0} \) is equal to \( I \), \( G_{s_0} = I \), and for each input \( i \), \( 1 \leq i \leq n_1 \), \( a_{j}^{(1)}(s_0) > t \) holds. Since \( J \) is a finite set, for each neuron \( i \), \( 1 \leq i \leq n_1 \), at the hidden layer there exists a set \( G_{s_0} \in P(J) \), such that

\[
v(G_{s_0}) \land g_{s_0}(G_{s_0}) = a_{j}^{(1)}(s_0) > t,\]

i.e. \( v(G_{s_0}) > t \) and \( g_{s_0}(G_{s_0}) > t \). Since the process of adjusting all the weights has already been stopped, the above inequalities can be fulfilled only if \( g_{s_0}(G_{s_0}) = 1 \), i.e. if set \( G_{s_0} \) is equal to \( J \), \( G_{s_0} = J \), and for each input \( j \), \( 1 \leq j \leq n_0 \), \( a_{j}^{(0)}(s_0) > t \) holds. Therefore, condition (4) is not satisfied.

The assumption that \( a_{j}^{(2)}(s) < t \) holds, implies in a similar way that condition (4) is not satisfied in that case, either.
2.3.2 Convergence condition in case of multiple training patterns

**Definition 2.** In case of multiple training patterns \( (X_m, t) \), \( m = 1, \ldots, M \), FBP algorithm is convergent to the target value \( t \) if there exists a number \( s_0 \in \mathbb{N}^+ \) such that following equalities hold:

\[
    t = a_m^{(2)}(s_0) = a_m^{(2)}(s_0 + 1) = a_m^{(2)}(s_0 + 2) = \ldots, \quad m = 1, \ldots, M
\]

\[
    w_{ij}^{(l)}(s_0) = w_{ij}^{(l)}(s_0 + 1) = w_{ij}^{(l)}(s_0 + 2) = \ldots, \quad l = 1, 2.
\]

**Theorem 2.** Let more than one training pattern \( (X_m, t) \), \( m = 1, \ldots, M \), be given. Let \( a_{mj}^{(0)}, a_{mj}^{(0)} \) be inputs of the \( m \)-th pattern, such that Theorem 1. holds, i.e. \( a_{mj}^{(0)} \geq t \) and \( a_{mj}^{(0)} \leq t \). Let \( a_{mj}^{(0)} = \bigwedge_{m=1}^{M} a_{mj}^{(0)} \) and \( a_{mj}^{(0)} = \bigvee_{m=1}^{M} a_{mj}^{(0)} \). Then FBP algorithm is convergent to the target value \( t \) iff the following condition holds:

\[
    a_{mj}^{(0)} \geq t \quad \text{and} \quad a_{mj}^{(0)} \leq t. \tag{5}
\]

**Proof.** Let condition (5) be satisfied. Since the inequalities \( t \leq a_{mj}^{(0)} \leq a_{mj}^{(0)} \) and \( t \geq a_{mj}^{(0)} \geq a_{mj}^{(0)} \) hold for each \( m, n = 1, \ldots, M \), according to Theorem 1. FBP algorithm is convergent to the target value \( t \) for each \( m, n = 1, \ldots, M \).

Let now FBP algorithm is convergent to the target value \( t \) for each \( m, n = 1, \ldots, M \). Then according to Theorem 1. the inequalities \( a_{mj}^{(0)} \geq t \) and \( a_{mj}^{(0)} \leq t \) hold for each \( m = 1, \ldots, M \). Therefore, the condition (5) is satisfied, too.

3. Simulation results

By help of a computer simulation we demonstrate the proposed FBP algorithm using two neural networks with single output neuron in case of single training pattern (cf. Figure 2) and in case of multiple training patterns (cf. Figure 2 and Figure 7). The FBP algorithm is compared with SBP algorithm (cf. Figure 8).

3.1 Single training pattern

In case of single training pattern FBP algorithm was iterated with linear activation function, training accuracy 0.001 (error goal), learning rate \( \eta = 1.0 \) and target value \( t = 0.8 \) (cf. Figure 2). According to Theorem 1. FBP algorithm is convergent to target value \( t \) belonging to the interval between the minimal and maximal input values \([a_{j}^{(0)}, a_{j}^{(0)}] = [0.1, 0.9] \). Figure 3 illustrates that for target values \( t = 0.2, t = 0.5 \) and \( t = 0.8 \) belonging to the interval \([0.1, 0.9] \) the network was trained only for 2 respectively for 3 steps. If the target value \( (t = 0.0 \) and \( t = 1.0) \) is outside the interval \([0.1, 0.9] \) the network output can not reach it and the training process is not convergent. This confirms the theoretical results. Here the final output values are 0.1 and 0.9 for \( t = 0.0 \) and for \( t = 1.0 \) respectively.
Fig. 2. Example neural network with single output neuron in the case of single and multiple training patterns.
3.2 Multiple training patterns

FBP algorithm was iterated with five training patterns (cf. Figure 2), linear activation function, error goal 0.001 and learning rate $\eta=1.0$. The interval between the maximum of minimal input values and minimum of maximal input values of the training patterns is $[a_{m,j}^{(0)}, a_{m,j}^{(0)}] = [0.4, 0.8]$. Thus according to Theorem 2. for the target values $t=0.4$, $t=0.6$ and $t=0.8$ belonging to the interval $[0.4, 0.8]$ the convergence of FBP algorithm is guaranteed. As illustrated in Figure 4, FBP algorithm trains the network only for 10 steps. If the target value ($t=0.0$ and $t=1.0$) is outside the interval $[0.4, 0.8]$ the output value oscillates. There is no convergence outside this interval. This confirms the theoretical result.
The sum-squared error for the targets \( t=0.4 \), \( t=0.6 \) and \( t=0.8 \) reaches the error goal 0.001 only for 2 epochs (cf. Figure 5) where during one epoch all 5 training patterns are learned. For target values outside the interval \([0.4, 0.8]\) the network can not be trained and the final sum-squared error remains constant equal to 0.59 for \( t=0.0 \) and 0.26 for \( t=1.0 \).

From the definition of FBP algorithm (cf. 2.2.) follows that its convergence speed is maximal for the learning rate \( \eta=1.0 \). Figure 6 illustrates this theoretical result. In case of single and multiple training patterns only 3
steps respectively 10 steps are needed for training the network at learning rate $\eta=1.0$. If the learning rate decreases the training steps needed increase. For example for $\eta=0.01$ 589 training steps in case of single training pattern and 870 training steps in case of multiple training patterns are needed.

Fig. 6. Dependence between learning rate and training steps needed for single training pattern (STP) and multiple training patterns (MTP)

3.3 Comparison of FBP algorithm with SBP algorithm

The convergence speed of FBP algorithm was compared with that one of SBP algorithm (cf. Figure 8) for a network with 4 hidden layers (cf. Figure 7), 100 randomly generated training patterns, randomly generated initial network weights, error goal 0.02 and target value $t=0.8$ belonging to the interval $[0,0.8]$. FBP algorithm is significant faster (7 epochs) than SBP algorithm (5500 epochs) for learning rate $\eta=0.01$. SBP algorithm is not convergent for learning rate $\eta>0.01$. On Figure 8 this is shown for $\eta=1.0$. For the same learning rate FBP algorithm is convergent only for 2 epochs.

Numerous other experiments with different neural networks, learning rates and training patterns confirmed the greater convergence speed of FBP algorithm over SBP algorithm. Of course this speed depends also on the initial network weights.
Fig. 7. Example neural network with single output neuron and 4 hidden layers

Fig. 8. Comparison of FBP and SBP algorithms in terms of steps needed for convergence in case of multiple training patterns for two learning rates (LR)
4. Conclusions

In this paper we propose a fuzzy extension of the backpropagation algorithm: fuzzy backpropagation algorithm. This learning algorithm uses as net function the fuzzy integral of Sugeno. Necessary and sufficient conditions for its convergence for single output networks in case of single and multiple training patterns are defined and proved. A computer simulation confirmed these theoretical results.

FBP algorithm has a number of advantages in comparison with SBP algorithm:

1. greater convergence speed implying significant reduction in computation time what is important in case of large sized neural networks;
2. reaches always forward to the target value without oscillations and there is no possibility to fall into local minimum;
3. requires no assumptions about probability distributions and independence of input data (pattern’s inputs);
4. enables the automation of the weights tuning process (quasi-unsupervised learning) by pointing out the interval where the target value belongs to [14][15]. The target values are determined:
   - automatically by computer - in the interval, specified by input data;
   - semi-automatically by experts that define the direction and degree of target value changes. Thus the actual target value is determined by fuzzy sets relevant to predefined linguistic expressions.

Another advantage of FBP algorithm is that it presents a generalisation of the fuzzy perceptron algorithm [19].


The ability of FBP algorithm for quasi-unsupervised learning was successful implemented for creation of adaptable user interface of the information system of Bulgarian parliament [15] and usability evaluation of hypermedia user interfaces [17].

Finally we should say that this study is not yet applied to many complex problems and the results are not compared with different modifications of SBP algorithm. However the current results are encouraging for continuation of our work. We will have to make many experiments and practical implementations of FBP algorithm.

References


Appendix 1: Notation list

\[ x_j \] the j-th pattern’s input

\[ \text{net}_{i}^{(l)} \] net value of the i-th neuron at the l-th layer

\[ a_i^{(l)} \] activation value of the i-th neuron at the l-th layer

\[ w_{ij}^{(l)} \] weight of the j-th input of the i-th neuron at the l-th layer

\[ f(\text{net}) \] activation function

\[ t \] the pattern’s output, i.e. target value

\[ \delta_{i}^{(l)} \] error signal relevant to the i-th neuron at the l-th layer

\[ s \] the s-th iteration step

\[ w_{ij}^{(l)}(s) \] weight \( w_{ij}^{(l)} \) at the s-th step

\[ a_i^{(l)}(s) \] activation value \( a_i^{(l)} \) at the s-th step

\[ \Delta w_{ij} \] weight change

\[ w_{ij}^{old} \] weight \( w_{ij}^{(l)}(s) \)

\[ w_{ij}^{new} \] weight \( w_{ij}^{(l)}(s + 1) \)

\[ \phi \] the empty set

\[ \mathbb{R} \] the set of real numbers

\[ +\mathbb{N} \] the set of positive integers

\[ J \] the set of (indices of) network inputs

\[ I \] the set of (indices of) neurons at the hidden layer

\[ P(I) \] the power set of the set \( I \)

\[ \vee \] max operation in unit interval \([0,1]\)

\[ \wedge \] min operation in unit interval \([0,1]\)

\[ a_m^{(l)} \] activation value \( a_i^{(l)} \) relevant to the m-th pattern

\[ t^m \] the pattern’s output of the m-th pattern

\[ g \] fuzzy measure on the set \( P(J) \)

\[ h \] fuzzy measure on the set \( P(I) \)