A Novel Fuzzy BP Learning Algorithm for Four-layer Regular Fuzzy Neural Networks[1]

Liu Puyin[2] Yang Wenqiang

(*Department of Mathematics, National University of Defense Technology, Changsha 410073, China*)

Abstract–The general fuzzy numbers are approximately represented as polygonal fuzzy numbers, which can be determined by finite nested closed interval. Based on interval arithmetic the input–output (I/O) relationship of a four-layer feedforward regular fuzzy neural network (FNN) is analyzed systematically. By introducing semi-jump function 'Lor' a group of partial derivative formulas are established for the error function of the four-layer regular FNN, in which the maximum operator '$\vee$' and minimum operator '$\wedge$' are included. A BP learning algorithm for fuzzy weights of the regular FNN is developed. To speed the convergence of the algorithm the learning constant is updated in each iteration step. Our experimental results show that the novel fuzzy BP algorithm can train a regular FNN efficiently to realize a family of fuzzy inference rules approximately, and to finish a uncomplete fuzzy inference rule table to demonstrate the FNN trained by our learning scheme having strong generalization capability.

Keywards–Polygonal line operator; Fuzzy arithmetic; Regular fuzzy neural network; Fuzzy BP algorithm.

I. Introduction

The fuzzy neural networks (FNN's), organic combinations of fuzzy logic and neural networks have attracted much attention in recent years (see, e.g., [1–3, 5–14, 17]). They combine the natural language description of fuzzy logic and the learning properties of artificial neural networks. Among those being used most, regular FNN's whose input, output signals and connection weights are fuzzy numbers, internal operations are based on Zadeh's principle can process fuzzy information directly. And they have found useful in many applied areas, such as system identification and system modeling[13,14], system control[2] and image processing[12,16] and so on.

One important problem related to regular FNN's is to develop some learning algorithm of fuzzy weights. Since the inputs, connection weights and the thresholds related to regular FNN's are fuzzy numbers, naturally it is much more difficult to develop their learning algorithms than conventional neural networks. This facts leads to lacking systematic achievements in the field. The basic processes to deal with the learning for regular FNN's are similar ones with conventional neural networks, that is, define a suitable error function, and develop some iteration schemes for fuzzy weights and thresholds. Since we do not have the calculus for fuzzy numbers, the conventional learning algorithms for multi-layer neural networks cannot be directly fuzzified. To apply BP type learning schemes in developing learning algorithms of regular FNN's the fuzzy weights and thresholds must be restricted as some special fuzzy sets. For instance, Buckley et al apply direct fuzzification to develop the fuzzy delta rule[3]. However, it cannot be used, practically because of the lack of theoretic support. By restricting fuzzy weights and fuzzy thresholds to be real numbers, Ishibuchi et al propose a fuzzy BP algorithm for real weights and thresholds, based on finite level sets of the fuzzy sets related[7]. Also Ishibuchi et al choose specifically triangular and trapezoidal fuzzy numbers as the fuzzy weights and thresholds of regular FNN's[8,9]. They utilize finite parameters related to those fuzzy numbers to develop learning algorithms, and the regular FNN's can be trained to realize a family of fuzzy inference rules approximately. In order to call off constraint conditions for fuzzy weights, Dunyak et al present a transformation which does not simplify the representation of fuzzy weights[5].

A general fuzzy number cannot be determined by a finite parameter collection. This is a main reason that causes the difficulties for developing FNN learning algorithms. So the results obtained so far in the area hold only in some special cases. Further an indispensable step to build these learning algorithms is to differentiate $\vee - \wedge$ functions by using the unit step function, i.e., for the given real constant a, define

$$\frac{\partial(x\vee a)}{\partial x}=\begin{cases}1, & x\geq a,\\ 0, & x<a;\end{cases}\quad \frac{\partial(x\wedge a)}{\partial x}=\begin{cases}1, & x\leq a,\\ 0, & x>a.\end{cases}$$

Above representations are only valid for special case $x \neq a$. And if $x = a$, they are no longer valid. Based on these two derivative formulas, the chain rules for differentiation of composition functions are only in form, without rigorous mathematical sense.

[1]Project supported by grants from National Natural Science Foundation of China (No. 60375023; No. 69974041).

[2]Email address: liupuyin@public.cs.hn.cn.

This paper develops a novel fuzzy BP algorithm of regular FNN's with general fuzzy numbers, based on rigorous mathematical sense. To this end, we at first expresses a general fuzzy number approximately as a polygonal fuzzy number which can be determined by finite $\alpha-$level sets, by introducing the polygonal line operator. Then we employ the interval arithmetic to analyze the I/O relationship of the regular FNN's. For given fuzzy pattern pairs for training the regular FNN, a suitable error function for testing the closeness between the real outputs and the desired ones is established. By the semi-jump function, a group of derivative formulas of the error function are developed. Thus, a novel fuzzy BP type learning algorithm for the FNN is built. Finally some simulation examples demonstrate the effectiveness of our results.

II. Polygonal Line Operator

In the section let us first introduce some basic notations and terminologies used in the paper. $\mathbb{R}$ means the set of all real numbers, and $\mathbb{N}$ the collection of all natural numbers. By $\mathcal{F}_0(\mathbb{R})$ we denote the set of all fuzzy numbers with the following conditions: for each $\widetilde{A}\in \mathcal{F}_0(\mathbb{R})$, it follows that

(i) $\forall\alpha \in (0,1]$, $\alpha-$cut $\widetilde{A}_\alpha\subset \mathbb{R}$ is a nonempty bounded and closed interval;

(ii) $\mathrm{supp}(\widetilde{A})\overset{\triangle}{=}\overline{\{x \in \mathbb{R}|\widetilde{A}(x)>0\}}\neq\emptyset$ is bounded;

(iii) If $\mathrm{supp}(\widetilde{A}) = [a_0^1, a_0^2]$, $\ker(\widetilde{A}) = [e_0^1, e_0^2]$, then $\widetilde{A}(\cdot)$ is strictly increasing on $[a_0^1, e_0^1]$, and strictly decreasing on $[e_0^2, a_0^2]$.

Write the support $\mathrm{supp}(\widetilde{A})$ as $\widetilde{A}_0$. Thus, $\widetilde{A}_0= [a_0^1, a_0^2]$, $\ker(\widetilde{A}) = [e_0^1, e_0^2]$, and by [4], we can easily prove that $\widetilde{A}(\cdot)$ is right continuous and strictly increasing on $[a_0^1, e_0^1)$, left continuous and strictly decreasing on $(e_0^2, a_0^2]$, and $\widetilde{A}(x) \equiv 1$ for $x \in [e_0^1, e_0^2]$.

By $d_H(A, B)$ for $A, B \subset \mathbb{R}$ ($A \neq \emptyset$, $B \neq \emptyset$) we denote the Hausdorff metric between A, B. So

$$[a, b], [c, d] \subset \mathbb{R} \Longrightarrow d_H([a, b],\ [c, d]) = |a-c| \vee |b-d|.$$

If $\widetilde{A}$, $\widetilde{B}\in \mathcal{F}_0(\mathbb{R})$, define metric $D(\widetilde{A}, \widetilde{B})$ as follows[4]: $D(\widetilde{A}, \widetilde{B}) = \bigvee_{\alpha\in[0,1]} \{d_H(\widetilde{A}_\alpha, \widetilde{B}_\alpha)\}$. It may easily be proved by [4] that $(\mathcal{F}_0(\mathbb{R}), D)$ is a metric space. Also the Euclidean metric $d_E\big([a, b],\ [c, d]\big)$ is as follows:

$$d_E\big([a, b],\ [c, d]\big)= \big\{(a-c)^2 + (b-d)^2\big\}^{\frac{1}{2}}.$$

For given intervals $[a, b]$, $[c, d] \subset \mathbb{R}$, we can imply

$$\begin{aligned} d_H\big([a, b],\ [c, d]\big) &\leq d_E\big([a, b],\ [c, d]\big) \\ &\leq \sqrt{2}\cdot d_H\big([a, b],\ [c, d]\big), \end{aligned}$$

that is, the metrics d_E and d_H are equivalent.

Next let us prove that each fuzzy number in $\mathcal{F}_0(\mathbb{R})$ can be represented as a polygonal fuzzy number approximately. To this end, for any $\widetilde{A}\in \mathcal{F}_0(\mathbb{R})$, we choose $\gamma \in \mathbb{N}$, and partition $[0, 1]$ into γ equal parts: $0 < 1/\gamma < \cdots < (\gamma-1)/\gamma < 1$. Let $\widetilde{A}_{i/\gamma}= [a_i^1, a_i^2]$ for each $i \in \{0, 1, ..., \gamma\}$. Link by line the points $(a_0^1, \widetilde{A}(a_0^1)), ..., (a_\gamma^1, \widetilde{A}(a_\gamma^1))$, $(a_\gamma^2, \widetilde{A}(a_\gamma^2)), ...,$ and $(a_0^2, \widetilde{A}(a_0^2))$, successively. And a polygonal line denoted by $\widetilde{tA}(\cdot)$ is established. We call $\widetilde{tA}$ a polygonal fuzzy number with respect to $\widetilde{A}$, whose membership curve is shown in Fig. 1. We can show, $\ker(\widetilde{A}) = \ker(\widetilde{tA}) = [a_\gamma^1, a_\gamma^2]$, $\mathrm{supp}(\widetilde{A}) = \mathrm{supp}(\widetilde{tA}) = [a_0^1, a_0^2]$. Moreover

$$\begin{cases} a_0^1 \leq a_1^1 \leq \cdots \leq a_\gamma^1 \leq a_\gamma^2 \leq a_{\gamma-1}^2 \leq \cdots \leq a_0^2; \\ \forall i \in \{0, 1, ..., \gamma\},\ \widetilde{A}_{i/\gamma}= (\widetilde{tA})_{i/\gamma}. \end{cases}$$

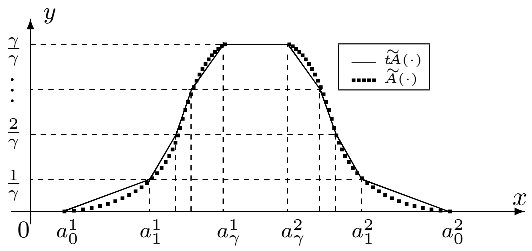

Fig. 1. Illustration of polygonal fuzzy operator

Let $\gamma \in \mathbb{N}$, define the operator $Z_\gamma :\ \mathcal{F}_0(\mathbb{R}) \longrightarrow \mathcal{F}_0(\mathbb{R})$ as follows:

$$\forall\ \widetilde{A}\in \mathcal{F}_0(\mathbb{R}),\ Z_\gamma(\widetilde{A}) =\widetilde{tA}\ .$$

$Z_\gamma(\cdot)$ is called $\gamma-$polygonal line operator.

Theorem 1. *Let $\widetilde{A}$, $\widetilde{B}\in \mathcal{F}_0(\mathbb{R})$, $\gamma \in \mathbb{N}$. And for $i = 0, 1, ..., \gamma$, let $\widetilde{A}_{i/\gamma}= [a_i^1, a_i^2]$, $\widetilde{B}_{i/\gamma}= [b_i^1, b_i^2]$. Then the following conclusions hold:*

(i) $Z_\gamma(\widetilde{A}) \subset Z_\gamma(\widetilde{B}) \Longleftrightarrow [a_i^1, a_i^2] \subset [b_i^1, b_i^2]$ for each $i \in \{0, 1, ..., \gamma\}$;

(ii) $\widetilde{A}\subset\widetilde{B}\Longleftrightarrow \forall\gamma \in \mathbb{N},\ Z_\gamma(\widetilde{A}) \subset Z_\gamma(\widetilde{B})$;

(iii) $m \in \mathbb{N}, \Longrightarrow D(Z_m(\widetilde{A}),\ Z_m(\widetilde{B})) \leq D(\widetilde{A}, \widetilde{B})$. Moreover $\lim_{m\to+\infty} D(\widetilde{A}, Z_m(\widetilde{A})) = 0$.

By Theorem 1, each fuzzy number in $\mathcal{F}_0(\mathbb{R})$ can be represented as a polygonal fuzzy number approximately. Thus, we can utilize level sets of fuzzy numbers to build the learning algorithm of regular FNN's.

III. Regular Fuzzy Neural Networks

We propose a four layer feedforward regular FNN whose topological architecture is shown in Fig. 2. The neurons in the input layer, the second hidden layer and output layer are linear. And each neuron in the first hidden layer has the transfer function $\sigma(\cdot)$

and a fuzzy threshold. The input–output (I/O) relationship of the FNN demonstrated in Fig. 2 can be represented as follows:

$$\widetilde{Y}=\sum_{k=1}^{q}\widetilde{W}_k\cdot\Big(\sum_{j=1}^{p}\widetilde{V}_{jk}\cdot\sigma\big(\langle\widetilde{\mathbf{X}},\ \widetilde{\mathbf{U}}(j)\rangle+\widetilde{\Theta}_j\big)\Big). \quad (1)$$

Specifically, if let $\widetilde{V}_{jk},\ \widetilde{\Theta}_j\in\mathcal{F}_0(\mathbb{R})$ in (1) be $v_{jk},\ \theta_j\in\mathbb{R}$, respectively, and $\widetilde{\mathbf{U}}(j)=(\widetilde{U}_{1j},...,\widetilde{U}_{dj})\in\mathcal{F}_0(\mathbb{R})^d$ be a vector $\mathbf{u}(j)\in\mathbb{R}^d$, then we obtain a simpler form of the I/O relationship:

$$\widetilde{Y}=\sum_{k=1}^{q}\widetilde{W}_k\cdot\Big(\sum_{j=1}^{p}v_{jk}\cdot\sigma\big(\langle\mathbf{u}(j),\ \widetilde{\mathbf{X}}\rangle+\theta_j\big)\Big). \quad (2)$$

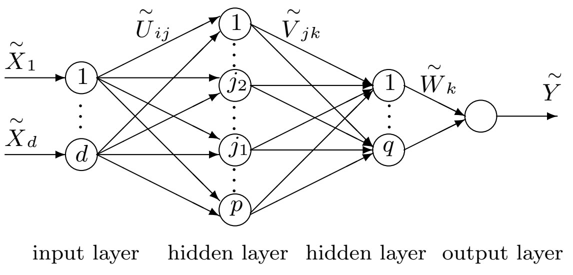

Fig. 2. Four-layer regular feedforward FNN

Next let us account for the extension operations and extended inner product $\langle\cdot,\cdot\rangle$ in (1) (2) as follows. For $\widetilde{A},\widetilde{B}\in\mathcal{F}_0(\mathbb{R}),\alpha\in\mathbb{R}$, we define[15]

$$\begin{aligned}(\widetilde{A}+\widetilde{B})(x)&=\bigvee_{x_1+x_2=x}\{\widetilde{A}(x_1)\wedge\widetilde{B}(x_2)\},\\(\widetilde{A}-\widetilde{B})(x)&=\bigvee_{x_1-x_2=x}\{\widetilde{A}(x_1)\wedge\widetilde{B}(x_2)\},\\(\widetilde{A}\cdot\widetilde{B})(x)&=\bigvee_{x_1\cdot x_2=x}\{\widetilde{A}(x_1)\wedge\widetilde{B}(x_2)\},\\(\alpha\cdot\widetilde{A})(x)&=\begin{cases}\widetilde{A}\big(\frac{x}{\alpha}\big), & \alpha\neq 0,\\ \chi_{\{0\}}, & \alpha=0,\end{cases}\end{aligned}$$

Define extended inner product: $\langle\widetilde{\mathbf{X}},\widetilde{\mathbf{Y}}\rangle=\sum_{i=1}^{d}\widetilde{X}_i\cdot\widetilde{Y}_i$ for $\widetilde{\mathbf{X}}=(\widetilde{X}_1,...,\widetilde{X}_d),\widetilde{\mathbf{Y}}=(\widetilde{Y}_1,...,\widetilde{Y}_d)$. When $\widetilde{\mathbf{X}}$ and $\widetilde{\mathbf{Y}}$ degenerate as the vectors in $\mathbb{R}^d$, $\langle\widetilde{\mathbf{X}},\widetilde{\mathbf{Y}}\rangle$ is the inner product in $\mathbb{R}^d$. If $f:\mathbb{R}\longrightarrow\mathbb{R}$ is a continuous function f can be extended as $f:\mathcal{F}_0(\mathbb{R})\longrightarrow\mathcal{F}_0(\mathbb{R})$ as follows[15]:

$$\forall\ \widetilde{X}\in\mathcal{F}(\mathbb{R}),\ f(\widetilde{X})(y)=\bigvee_{f(x)=y}\{\widetilde{X}(x)\}.$$

By [11, 13], The regular FNN's defined as (2) can be universal approximators of a class of fuzzy function. So in the following let us analyze the I/O relationship of (2) and develop a novel BP type learning algorithm. Choose $\gamma\in\mathbb{N}$, let $\alpha_{k'}=k'/\gamma$ ($k'=0,1,...,\gamma$). For $i=1,...,d;\ k=1,...,q$, let $(\widetilde{W}_k)_{\alpha_{k'}}=[w^1_{k(k')},\ w^2_{k(k')}]$ and $(\widetilde{X}_i)_{\alpha_{k'}}=[x^1_{i(k')},\ x^2_{i(k')}]$. If $\widetilde{\mathbf{X}}=(\widetilde{X}_1,...,\widetilde{X}_d)\in\mathcal{F}_0(\mathbb{R})^d$, we denote

$$\begin{aligned}X^1_{j(k')}&=\sum_{i=1}^{d}\big(u_{ij}x^1_{i(k')}\wedge u_{ij}x^2_{i(k')}\big)+\theta_j,\\X^2_{j(k')}&=\sum_{i=1}^{d}\big(u_{ij}x^1_{i(k')}\vee u_{ij}x^2_{i(k')}\big)+\theta_j,\\Y^1_{k(k')}&=\sum_{j=1}^{p}\big(v_{jk}\sigma(X^1_{j(k')})\wedge v_{jk}\sigma(X^2_{j(k')})\big),\\Y^2_{k(k')}&=\sum_{j=1}^{p}\big(v_{jk}\sigma(X^1_{j(k')})\vee v_{jk}\sigma(X^2_{j(k')})\big),\end{aligned}\quad (3)$$

Then by the interval arithmetic[4], $\alpha_{k'}$−cut of $\widetilde{Y}$ are represented as follows:

$$\begin{aligned}\widetilde{Y}_{\alpha_{k'}}&\triangleq\sum_{k=1}^{q}\big[Z^1_{k(k')},\ Z^2_{k(k')}\big]=\sum_{k=1}^{q}\big[w^1_{k(k')},\ w^2_{k(k')}\big]\cdot\\&\cdot\Big(\sum_{j=1}^{p}v_{jk}\cdot\sigma\Big(\sum_{i=1}^{d}u_{ij}\cdot\big[x^1_{i(k')},\ x^2_{i(k')}\big]+\theta_j\Big)\Big)\\&=\sum_{k=1}^{q}\big[w^1_{k(k')},\ w^2_{k(k')}\big]\cdot\big[Y^1_{k(k')},\ Y^2_{k(k')}\big],\end{aligned}\quad (4)$$

thus, $Z^1_{k(k')}$ and $Z^2_{k(k')}$ can be respectively expressed as

$$\begin{aligned}Z^1_{k(k')}&=w^1_{k(k')}Y^1_{k(k')}\wedge w^2_{k(k')}Y^1_{k(k')}\wedge\\&\quad\wedge w^1_{k(k')}Y^2_{k(k')}\wedge w^2_{k(k')}Y^2_{k(k')},\\Z^2_{k(k')}&=w^1_{k(k')}Y^1_{k(k')}\vee w^2_{k(k')}Y^1_{k(k')}\vee\\&\quad\vee w^1_{k(k')}Y^2_{k(k')}\vee w^2_{k(k')}Y^2_{k(k')}.\end{aligned}$$

IV. Learning Algorithm

Two key steps to design learning algorithm for the fuzzy weights and thresholds of regular FNN's as (2) are to derive the error function of fuzzy outputs from output neuron and the corresponding desired values (fuzzy targets)[7−10,14], and to define the derivatives of the error function in which fuzzy operators '$\vee$' or '$\wedge$' are included. To this end, let us introduce a function which is called semi-jump function to calculate the derivative operations of $\vee-\wedge$ functions[17]. Let

$$\mathrm{lor}(\cdot):\mathbb{R}\longrightarrow\mathbb{R},\ \ \mathrm{lor}(x)=\begin{cases}1, & x>0,\\ \frac{1}{2}, & x=0,\\ 0, & x<0.\end{cases}$$

Using the semi-jump function 'lor$(\cdot)$' we can establish the derivative operation laws of $\vee-\wedge$ functions.

Proposition 1[17]. *Let the real functions $f,\ g$ be differentiable on $\mathbb{R}$, $h_1=f\vee g,\ h_2=f\wedge g$. Then $h_1,\ h_2$*

are almost everywhere differentiable on $\mathbb{R}$, *moreover*

$$\frac{\mathrm{d}h_1(x)}{\mathrm{d}x} = \frac{\mathrm{d}(f(x)\vee g(x))}{\mathrm{d}x}$$
$$= \mathrm{lor}(f(x)-g(x))\frac{\mathrm{d}f(x)}{\mathrm{d}x} + \mathrm{lor}(g(x)-f(x))\frac{\mathrm{d}g(x)}{\mathrm{d}x};$$
$$\frac{\mathrm{d}h_2(x)}{\mathrm{d}x} = \frac{\mathrm{d}(f(x)\wedge g(x))}{\mathrm{d}x}$$
$$= \mathrm{lor}(f(x)-g(x))\frac{\mathrm{d}g(x)}{\mathrm{d}x} + \mathrm{lor}(g(x)-f(x))\frac{\mathrm{d}f(x)}{\mathrm{d}x}.$$

Specifically, if $a\in\mathbb{R}$, *the following facts hold:*

$$\frac{\mathrm{d}(a\vee f(x))}{\mathrm{d}x} = \mathrm{lor}(f(x)-a)\frac{\mathrm{d}f(x)}{\mathrm{d}x},$$
$$\frac{\mathrm{d}(a\wedge f(x))}{\mathrm{d}x} = \mathrm{lor}(a-f(x))\frac{\mathrm{d}f(x)}{\mathrm{d}x}.$$

A. Derivatives of Error Function

For $\big\{\big((\widetilde{X}_1(l),...,\widetilde{X}_d(l));\widetilde{O}(l)\big),\ l=1,...,L\big\}$, a given family of fuzzy pattern pairs, we can train the regular FNN in Fig. 2, where $\big(\widetilde{X}_1(l),...,\widetilde{X}_d(l)\big)$ is the input, and $\widetilde{O}(l)$ is the corresponding output. Let $\widetilde{O}(l)_{\alpha_{k'}} = \big[o^1_{k'}(l),\ o^2_{k'}(l)\big]$, and $\big(\widetilde{X}_i(l)\big)_{\alpha_{k'}} = \big[x^1_{i(k')}(l),\ x^2_{i(k')}(l)\big]$. To measure the difference between the desired and real outputs, we define the following error function:

$$E = \frac{1}{2}\sum_{l=1}^{L}\Big[\sum_{k'=0}^{\gamma} d_E\Big(\widetilde{Y}(l)_{\alpha_{k'}},\ \big(\widetilde{O}(l)\big)_{\alpha_{k'}}\Big)^2\Big]. \qquad (5)$$

where $\widetilde{Y}(l)_{\alpha_{k'}} = \sum_{k=1}^{q}\big[Z^1_{k(k')}(l),\ Z^2_{k(k')}(l)\big]$, $Z^1_{k(k')}(l)$ and $Z^2_{k(k')}(l)$ are defined respectively by letting $x^1_{i(k')} = x^1_{i(k')}(l)$ and $x^2_{i(k')} = x^2_{i(k')}(l)$ in (3) (4). We write all adjustable parameters related to the FNN in (2) as a vector $\mathbf{w}=(w_1,...,w_N)^{\mathrm{T}}$, that is

$$\mathbf{w} = \big(u_{11},...,u_{dp},v_{11},...,v_{pq},\theta_1,...,\theta_p,$$
$$w^1_{1(0)},...,w^1_{q(\gamma)},w^2_{q(\gamma)},...,w^2_{1(0)}\big)^{\mathrm{T}}.$$

We can build the partial derivative formulas related to the error function E.

Theorem 2. *Let the transfer function* $\sigma:\mathbb{R}\longrightarrow\mathbb{R}$ *be continuously differentiable and non-negatively increasing. Then the error function defined by (5) is differentiable a.e. with respect to* $\mathbf{w}$ *in* $\mathbb{R}^N$. *Further, for* $r=1,2;\ k=1,...,q;\ k'=0,1,...,\gamma;\ j=1,...,p;\ i=1,...,d$, *if let*

$$\Delta^1_{k'}(l) = \sum_{k=1}^{q} Z^1_{k(k')}(l) - o^1_{k'}(l),$$
$$\Delta^2_{k'}(l) = \sum_{k=1}^{q} Z^2_{k(k')}(l) - o^2_{k'}(l);$$

and

$$D^1_{k(k')}(l) = w^1_{k(k')}Y^1_{k(k')}(l)\wedge w^1_{k(k')}Y^2_{k(k')}(l)$$
$$-w^2_{k(k')}Y^1_{k(k')}(l)\wedge w^2_{k(k')}Y^2_{k(k')}(l);$$
$$D^2_{k(k')}(l) = w^1_{k(k')}Y^1_{k(k')}(l)\vee w^1_{k(k')}Y^2_{k(k')}(l)$$
$$-w^2_{k(k')}Y^1_{k(k')}(l)\vee w^2_{k(k')}Y^2_{k(k')}(l);$$
$$A^r_{k(k')}(l) = \mathrm{lor}\big((-1)^r D^1_{k(k')}(l)\big)\big(Y^1_{k(k')}(l)\mathrm{lor}\big(w^r_{k(k')}\big)$$
$$+Y^2_{k(k')}(l)\mathrm{lor}\big(-w^r_{k(k')}(l)\big)\big);$$
$$B^r_{k(k')}(l) = \mathrm{lor}\big((-1)^{r+1} D^2_{k(k')}(l)\big)\big(Y^1_{k(k')}(l)\mathrm{lor}\big(-w^r_{k(k')}\big)$$
$$+Y^2_{k(k')}(l)\mathrm{lor}\big(w^r_{k(k')}(l)\big)\big);$$
$$U^r_{k(k')}(l) = \mathrm{lor}\big((-1)^r w^2_{k(k')}\big)\mathrm{lor}\big(D^1_{k(k')}(l)\big)$$
$$+\mathrm{lor}\big((-1)^r w^1_{k(k')}\big)\mathrm{lor}\big(-D^1_{k(k')}(l)\big);$$
$$V^r_{k(k')}(l) = \mathrm{lor}\big((-1)^{r+1} w^1_{k(k')}\big)\mathrm{lor}\big(D^2_{k(k')}(l)\big)$$
$$+\mathrm{lor}\big((-1)^{r+1} w^2_{k(k')}\big)\mathrm{lor}\big(-D^2_{k(k')}(l)\big);$$
$$C^r_{ijk(k')}(l) = v_{jk}\Big\{\mathrm{lor}\big((-1)^r v_{jk}\big)\sigma'\big(X^2_{j(k')}(l)\big)\cdot$$
$$\cdot\Big(\sum_{t=1}^{2} x^{3-t}_{i(k')}(l)\mathrm{lor}((-1)^{t+1}u_{ij})\Big)$$
$$+\mathrm{lor}\big((-1)^{r+1} v_{jk}\big)\sigma'\big(X^1_{j(k')}(l)\big)\cdot$$
$$\cdot\Big(\sum_{t=1}^{2} x^{3-t}_{i(k')}(l)\mathrm{lor}((-1)^{t}u_{ij})\Big)\Big\};$$
$$H^1_{k(k')}(l) = \Delta^1_{k'}(l)U^1_{k(k')}(l) + \Delta^2_{k'}(l)V^1_{k(k')}(l);$$
$$H^2_{k(k')}(l) = \Delta^1_{k'}(l)U^2_{k(k')}(l) + \Delta^2_{k'}(l)V^2_{k(k')}(l),$$

we obtain the following partial derivative formulas:

(i) $\displaystyle\frac{\partial E}{\partial\theta_{jk}} = \sum_{l=1}^{L}\sum_{k'=0}^{\gamma}\sum_{k=1}^{q}\Big\{H^1_{k(k')}(l)\cdot$
$$\cdot\big(\sum_{t=1}^{2}\sigma'(X^{3-t}_{j(k')}(l))\mathrm{lor}((-1)^{t+1}v_{jk})\big)$$
$$+H^2_{k(k')}(l)\big(\sum_{t=1}^{2}\sigma'(X^{3-t}_{j(k')}(l))\mathrm{lor}((-1)^{t}v_{jk})\big)\Big\}.$$

(ii) $\displaystyle\frac{\partial E}{\partial u_{ij}} = \sum_{l=1}^{L}\sum_{k'=0}^{\gamma}\sum_{k=1}^{q}\Big\{\Delta^1_{k'}(l)\cdot$
$$\cdot\Big(U^1_{k(k')}(l)C^2_{ijk(k')}(l) + U^2_{k(k')}(l)C^1_{ijk(k')}(l)\Big)$$
$$+\Delta^2_{k'}(l)\Big(V^1_{k(k')}(l)C^2_{ijk(k')}(l) + V^2_{k(k')}(l)C^1_{ijk(k')}(l)\Big)\Big\};$$

(iii) $\displaystyle\frac{\partial E}{\partial v_{jk}} = \sum_{l=1}^{L}\sum_{k'=0}^{\gamma}\Big\{H^1_{k(k')}(l)\cdot$
$$\cdot\big(\sum_{t=1}^{2}\sigma(X^{3-t}_{j(k')}(l))\mathrm{lor}((-1)^{t}v_{jk})\big)$$
$$+H^2_{k(k')}(l)\big(\sum_{t=1}^{2}\sigma(X^{3-t}_{j(k')}(l))\mathrm{lor}((-1)^{t+1}v_{jk})\big)\Big\};$$

(iv) $\displaystyle\frac{\partial E}{\partial w^r_{k(k')}} = \sum_{l=1}^{L}\big\{A^r_{k(k')}(l)\Delta^1_{k'}(l) + B^r_{k(k')}(l)\Delta^2_{k'}(l)\big\}$
for $r=1,2$.

Proof The proof of (iv) is easy. Considering the following facts we can prove (i)–(iii), respectively:

$$\frac{\partial Z^r_{k(k')}(l)}{\partial u_{ij}} = \frac{\partial Z^r_{k(k')}(l)}{\partial Y^1_{k(k')}} \frac{\partial Y^1_{k(k')}(l)}{\partial u_{ij}} + \frac{\partial Z^r_{k(k')}(l)}{\partial Y^2_{k(k')}} \frac{\partial Y^2_{k(k')}(l)}{\partial u_{ij}};$$

$$\frac{\partial Z^r_{k(k')}(l)}{\partial v_{jk}} = \frac{\partial Z^r_{k(k')}(l)}{\partial Y^1_{k(k')}} \frac{\partial Y^1_{k(k')}(l)}{\partial v_{jk}} + \frac{\partial Z^r_{k(k')}(l)}{\partial Y^2_{k(k')}} \frac{\partial Y^2_{k(k')}(l)}{\partial v_{jk}};$$

$$\frac{\partial Z^r_{k(k')}(l)}{\partial \theta_j} = \frac{\partial Z^r_{k(k')}(l)}{\partial Y^1_{k(k')}} \frac{\partial Y^1_{k(k')}(l)}{\partial \theta_j} + \frac{\partial Z^r_{k(k')}(l)}{\partial Y^2_{k(k')}} \frac{\partial Y^2_{k(k')}(l)}{\partial \theta_j}.$$

where $r = 1, 2$. Easily we can show that E is differentiable a.e. with respect to $\mathbf{w}$ in $\mathbb{R}^N$. Thus theorem is proved. □

B. Learning Algorithm

By the partial derivatives determined by Theorem 2, we can develop a novel fuzzy BP algorithm. Let the learning constant η change as the time step iterates, i.e. suppose $\eta = \eta[t] = \rho(E[t])$, η is defined by the error function $E[t]$ in each iteration: $\rho(E[t]) = \rho_0 E[t] / \|\nabla E(\mathbf{w})\|^2$, where

$$\nabla E(\mathbf{w}) = \big(\partial E/\partial w_1, ..., \partial E/\partial w_N\big)$$

is a gradient vector, ρ_0 is a given constant. we obtain the following iteration scheme:

$$\begin{cases} u_{ij}[t+1] = u_{ij}[t] - \rho_0 \cdot E[t] \cdot \dfrac{\partial E[t]}{\partial u_{ij}[t]} \Big/ \|\nabla E(\mathbf{w})\|^2; \\ v_{jk}[t+1] = v_{jk}[t] - \rho_0 \cdot E[t] \cdot \dfrac{\partial E[t]}{\partial v_{jk}[t]} \Big/ \|\nabla E(\mathbf{w})\|^2; \\ \theta_j[t+1] = \theta_j[t] - \rho_0 \cdot E[t] \cdot \dfrac{\partial E[t]}{\partial \theta_j[t]} \Big/ \|\nabla E(\mathbf{w})\|^2; \\ w^r_{k(k')}[t+1] = w^r_{k(k')}[t] - \dfrac{\rho_0 E[t] \partial E[t]}{\partial w^r_{k(k')}[t]} \Big/ \|\nabla E(\mathbf{w})\|^2, \end{cases} \tag{6}$$

where $r = 1, 2$. Let ρ_0 ba a small positive number, e.g., $\rho_0 = 0.01$.

Algorithm 1 Novel type fuzzy BP algorithm.

Step 1. Randomly choose initial values: $u_{ij}[0]$, $v_{jk}[0]$, $\theta_j[0]$ and $w^r_{k(k')}[0]$ $(r = 1, 2)$, and let $t = 0$;

Step 2. Calculate following partial derivatives: $\partial E[t]/\partial u_{ij}[t]$, $\partial E[t]/\partial v_{jk}[t]$, $\partial E[t]/\partial \theta_j[t]$, and $\partial E[t]/\partial w^r_{k(k')}[t]$;

Step 3. According to iteration scheme (6) update the parameters u_{ij}, v_{jk}, θ_j and $w^r_{k(k')}$;

Step 4. For any $k \in \{1, ..., q\}$, we re-array the set $\{w^r_{k(k')} | r = 1, 2;\ k' = 0, 1, .., \gamma\}$ increasingly, i.e.

$$w^1_{k(0)} \le w^1_{k(1)} \le \cdots \le w^1_{k(\gamma)} \le w^2_{k(\gamma)} \le \cdots \le w^2_{k(0)}.$$

Step 5. Discriminate whether $|E[t]| < \varepsilon$? If yes go to Step 6, otherwise let $t = t + 1$ go to Step 2;

Step 6. Output all parameters.

V. Simulations

In the section, the proposed fuzzy BP algorithm is demonstrated by computer simulations on a few of numerical examples. In the following simulations, the upper-bound of iteration steps M is assumed to be 2×10^4, and the error bound $\varepsilon = 0.1$. In the hidden layers of the regular FNN as in Fig. 2 let $p = q = 5$. Choose $L = 4$. The activation function is chosen as $\sigma : \mathbb{R} \longrightarrow \mathbb{R}_+$, defined as follows: $\forall x \in \mathbb{R}$, $\sigma(x) = 1/(1 + \mathrm{e}^{-x})$, i.e. σ is the Sigmoidal function. Obviously it is a increasing, non-negative and differentiable function.

A. Learning capability

To examine the learning capability of a regular FNN, let us now study a simulation example related to digital image enhancement. Suppose [0, G] to be a grey degree interval, in which includes all grey levels of the digital images related, and here we choose $G = 255$. By some fuzzy sets, such as, 'Bright ($\widetilde{B}_R$)', 'Brighter ($\widetilde{B}_r$)', 'Medium ($\widetilde{M}_e$)', 'Darker ($\widetilde{D}_r$)' and 'Dark ($\widetilde{D}_R$)' we can describe the gray levels of the images[12]. To develop some efficient image filters based on fuzzy inference, the first step is to design a family of suitable fuzzy rules. For example, the following 'IF–THEN' rules are usual cases in designing fuzzy filters[12,16] :

$$\begin{array}{l} \text{IF } x_1 \text{ is } \widetilde{B}_R \text{ AND } x_2 \text{ is } \widetilde{B}_R \text{ THEN } y \text{ is } \widetilde{B}_R; \\ \text{IF } x_1 \text{ is } \widetilde{B}_R \text{ AND } x_2 \text{ is } \widetilde{D}_R \text{ THEN } y \text{ is } \widetilde{M}_e; \\ \text{IF } x_1 \text{ is } \widetilde{D}_R \text{ AND } x_2 \text{ is } \widetilde{B}_R \text{ THEN } y \text{ is } \widetilde{M}_e; \\ \text{IF } x_1 \text{ is } \widetilde{D}_R \text{ AND } x_2 \text{ is } \widetilde{D}_R \text{ THEN } y \text{ is } \widetilde{D}_R. \end{array} \tag{7}$$

The antecedent and consequent fuzzy sets '$\widetilde{D}_R$', '$\widetilde{M}_e$' and '$\widetilde{B}_R$' are fuzzy numbers defined on [0, G].

Let us now employ the regular FNN's defined as in Fig. 2 to realize the fuzzy IF–THEN inference rules defined in (7), approximately. Also we present the numerical comparison of our model with other fuzzified neural networks developed in [7, 9]. To this end we assume that the input related is a two dimensional variable (x_1, x_2), and the output is an one dimensional variable y. The membership curves of fuzzy numbers $\widetilde{B}_R$, $\widetilde{M}_e$, $\widetilde{D}_R$ are shown in (a) of Fig. 3, respectively at the bottom of this page. Choose

$$\begin{array}{l} \big\{\big((\widetilde{B}_R,\ \widetilde{B}_R),\ \widetilde{B}_R\big),\ \big((\widetilde{B}_R,\ \widetilde{D}_R),\ \widetilde{M}_e\big), \\ \quad \big((\widetilde{D}_R,\ \widetilde{B}_R),\ \widetilde{M}_e\big),\ \big((\widetilde{D}_R,\ \widetilde{D}_R),\ \widetilde{D}_R\big)\big\}, \end{array} \tag{8}$$

as the training patterns for designing learning algorithms of the regular FNN's as (2) and those in [7], [9].

The regular FNN defined as (2) can be trained by using Algorithm 1, and corresponding to the input patterns in the training set (8), we can get the actual outputs after 20000 iteration steps, as shown (b) of Fig. 3 at the bottom of this page. Ishibuchi et al use the symmetric triangular fuzzy number weights and thresholds in [9] to build up a regular FNN, whose adjustable parameters are only two kinds of parameters

— two endpoints of the $\alpha-$cuts. The BP type learning algorithms are used to train this FNN model, also it can realize the inference rules in (7) approximately through the training pattern set (8). The corresponding actual and desired outputs are shown in (c) of Fig. 3 at the bottom of this page, after 20000 iteration steps. In [7] Ishibuchi et al take the real numbers as the connection weights and thresholds to build a FNN model. Similarly with the convenient BP algorithm they develop a learning algorithm of the FNN. By iterating 20000 steps, we can get the actual outputs to approximate the rules in (7), as shown in (d) of Fig. 3 at the bottom of this page. By comparing (b) (c) (d) of Fig. 3, we can easily find that the regular FNN as (2) gives the best results, i.e. the error of the regular FNN is significantly lower than those Ishibuchi's FNN models[7, 9].

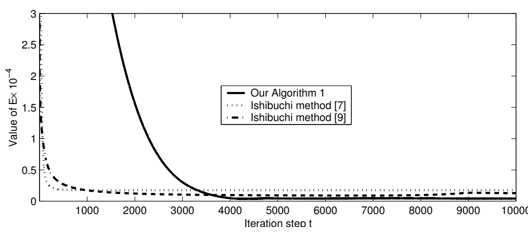

Fig. 4. Error curves of different FNN models

By Fig. 4 we show the curves of the error function defined by (5), corresponding to above three FNN models, i.e. the regular FNN defined as (2) and two Ishibuchi's models[7, 9]. Through Fig. 4 we can find that despite larger error at the beginning of iteration, the square error of our regular FNN is lowest, and corresponding to Ishibuchi's two FNN models, the square errors are both larger. So our result is also the best.

B. Generalization capability

Let us proceed to study fuzzy rules to show the generalization capability of the regular FNN's as in Fig. 2. The fuzzy rules in (7) are shown as Fig. 5, a fuzzy rule table, in which only four rules out of 25 fuzzy IF—THEN rules are presented and others are missing. Now let us complete the rule table by assigning one of the five fuzzy sets as '$\widetilde{B}_R$' '$\widetilde{B}_r$' '$\widetilde{M}_e$' '$\widetilde{D}_r$' and '$\widetilde{D}_R$' to the consequent of each missing rule. For example, when we choose the input (x_1, x_2) to be $(\widetilde{D}_R, \widetilde{D}_r)$, $(\widetilde{D}_R, \widetilde{M}_e)$, $(\widetilde{D}_r, \widetilde{B}_r)$, and $(\widetilde{M}_e, \widetilde{B}_R)$, $(\widetilde{B}_R, \widetilde{B}_r)$, respectively, the corresponding outputs of the regular FNN as (2) are respectively shown in Fig. 7, from which we can obtain their respective linguistic values: '$\widetilde{D}_R$' '$\widetilde{D}_r$' '$\widetilde{M}_e$' '$\widetilde{B}_r$' and '$\widetilde{B}_R$'. Similarly we can complete the other missing rules, as shown in Fig. 6. Obviously, these consequents conform to inference sense in (7). So the regular FNN's as (2) possesses strong generalization capability, which is advantageous over that of Ishibuchi's model in [9], since the similar rule table is completed based on nine fuzzy rules.

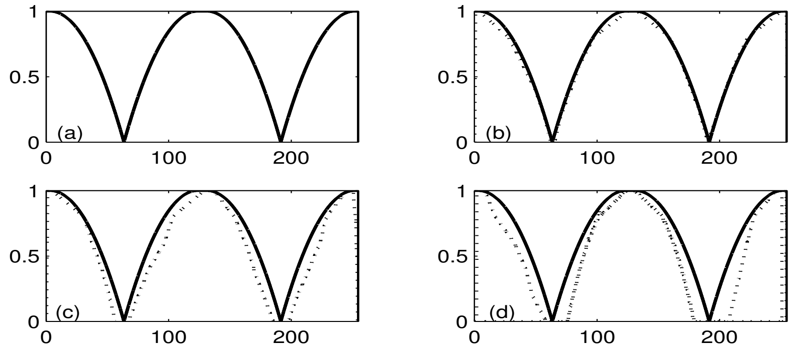

Fig. 3. Membership curves of fuzzy numbers 'Dark' 'Medium' 'Bright' : (a) desired curves; (b) desired curves (—) and actual output curves by our model (···); (c) desired curves (—) and actual output curves by Ishibuchi model in [9] (···); (d) desired curves (—) and actual output curves by Ishibuchi model in [7] (···).

$x_2 \backslash x_1$	$\widetilde{D}_R$	$\widetilde{D}_r$	$\widetilde{M}_e$	$\widetilde{B}_r$	$\widetilde{B}_R$
$\widetilde{D}_R$	$\widetilde{D}_R$				$\widetilde{M}_e$
$\widetilde{D}_r$					
$\widetilde{M}_e$					
$\widetilde{B}_r$					
$\widetilde{B}_R$	$\widetilde{M}_e$				$\widetilde{B}_R$

Fig. 5. Uncomplete fuzzy rule table

$x_2 \backslash x_1$	$\widetilde{D}_R$	$\widetilde{D}_r$	$\widetilde{M}_e$	$\widetilde{B}_r$	$\widetilde{B}_R$
$\widetilde{D}_R$	$\widetilde{D}_R$	$\widetilde{D}_r$	$\widetilde{D}_r$	$\widetilde{D}_r$	$\widetilde{M}_e$
$\widetilde{D}_r$	$\widetilde{D}_r$	$\widetilde{D}_r$	$\widetilde{D}_r$	$\widetilde{M}_e$	$\widetilde{B}_r$
$\widetilde{M}_e$	$\widetilde{D}_r$	$\widetilde{D}_r$	$\widetilde{M}_e$	$\widetilde{B}_r$	$\widetilde{B}_r$
$\widetilde{B}_r$	$\widetilde{D}_r$	$\widetilde{M}_e$	$\widetilde{B}_r$	$\widetilde{B}_r$	$\widetilde{B}_r$
$\widetilde{B}_R$	$\widetilde{M}_e$	$\widetilde{B}_r$	$\widetilde{B}_r$	$\widetilde{B}_r$	$\widetilde{B}_R$

Fig. 6. Fuzzy rule table completed by FNN (2)

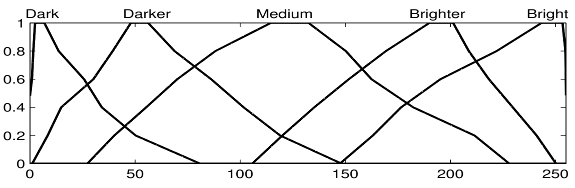

Fig. 7 Actual outputs of FNN (2)

V. Conclusions

By the polygonal line operator we can express a general fuzzy number approximately as a polygonal fuzzy number. Thus, with regard to the endpoints of finite level sets of fuzzy number weights in a regular FNN we can develop BP learning algorithm for the FNN. In the paper the partial derivative formulas of the error function are built based on rigorous mathematical sense through introducing semi-jump function. To accelerate convergence of learning algorithm we take the learning constant as a function of the iteration step t. Some simulations demonstrate the regular FNN's based on the novel fuzzy BP algorithm can realize a given fuzzy rule family approximately, and have strong generalization capability. Also, through the polygonal line operator we can employ genetic algorithm to build learning schemes for regular FNN's[1], which is a meaningful subject in future research.

References

[1] R. A. Aliev and B. Fazlollahi, et al, Genetic algorithm-based learning of fuzzy neural networks. Part I: feed-forward fuzzy neural networks, *Fuzzy Sets and Systems,* **118**(2001) 351–358.

[2] J. J. Buckley and Y. Hayashi, Can neural nets be universal approximators for fuzzy functions? *Fuzzy Sets and Systems*, **101**(1999) 323–330.

[3] J. J. Buckley and Y. Hayashi, Direct fuzzification of neural networks, in *Proc. of 1st Asian Fuzzy Sys. Symp.,* vol. 1 (1993) 560–567.

[4] P. Diamond and P. Kloeden, Metric spaces of fuzzy sets, Singapore: *World Scientific Publishing*, 1994.

[5] J. Dunyak and D. Wunsch, "Fuzzy number neural networks," *Fuzzy Sets and Systems,* **108**(1999) 49–58.

[6] T. Feuring and W. M. Lippe, The fuzzy neural network approximation lemma, *Fuzzy Sets and Systems*, **102**(1999) 227–236.

[7] H. Ishibuchi, R. Fujioka, and H. Tanaka, Neural networks that learn from fuzzy if–then rules, *IEEE Trans. on Fuzzy Systems,* **1**(1993) 85–97.

[8] H. Ishibuchi, K. Kwon and H. A. Tanaka, Learning algorithm of fuzzy neural networks with triangular fuzzy weights, *Fuzzy sets and Systems*, **71**(1995) 277–293.

[9] H. Ishibuchi and M. Nii, Numerical analysis of the learning of fuzzified neural networks from fuzzy if–then rules, *Fuzzy Sets and Systems,* **120**(2001) 281–307.

[10] Zhenquan Li, V. Kecman and A. Ichikawa, Fuzzified neural network based on fuzzy number operations, *Fuzzy Sets and Systems*, **130**(2002) 291–304.

[11] Puyin Liu, Analyses of regular fuzzy neural networks for approximation capability, *Fuzzy Sets and Systems,* **114**(2000) 329–338.

[12] Puyin Liu and Hongxing Li, Image restoration techniques based on fuzzy neural networks, *Science in China, Series F,* **45**(2002)(4) 273–285.

[13] Puyin Liu and Hongxing Li, *Fuzzy Neural Network Theory and Application*, Singapore: *World Scientific Publishing*, 2004.

[14] Puyin Liu and Hongxing Li, Efficient learning algorithms for three-layer regular feedforward neural networks, *IEEE Trans. on Neural Networks,* 2004, 15(3).

[15] H. T. Nguyen, A note on the extension principle for fuzzy set, *J. Math. Anal. Appl.*, **64**(1978) 369–380.

[16] F. Russo, Hybrid neuro-fuzzy filter for impulse noise removal, *Pattern Recognition,* **32**(1999) 1843–1855.

[17] X. H. Zhang, C. C. Huang and S. H. Tan, et al, The min-max function differentiation and training of fuzzy neural networks, *IEEE Trans. on Neural Networks*, **7**(1996) 1139–1150.