Application of rough fuzzy-neural network in short-term load forecasting

Feng Li
College of electrical engineering Zhejiang University
HangZhou , 310027 China
E-mail: beth_feng@zju.edu.cn

Qiu jia-ju
College of electrical engineering Zhejiang University
HangZhou , 310027 China
E-mail: ee_qiu@zju.edu.cn

***Abstract* – Integrated with rough set theory and fuzzy neural network, this article presents a hybrid model for short-term load forecasting. The genetic algorithm is used to find the minimum reduct which is relevant to electric loads, and the crude domain knowledge extracted from the elementary data set is applied to design the structure and weights of the network. It is testified by the simulation results that the rough fuzzy neural network has better precision and convergence than the traditional fuzzy neural network. Moreover, it becomes easier to understand the transferring way of knowledge in neural network.**

I. INTRODUCTION

Short-term load forecasting plays a key role in power system operation and planning. Its main objective is to extrapolate past load behavior while taking into account the effect of influencing factors. Since the electric load is a function of weather variables and human social activities, the variety of the short-term load is very complex, sometimes it changes evenly, sometimes linearly, and sometimes randomly. In early time regression models [1], time series and expert systems are used generally. Now artificial neural network [2][3] becomes to be the most popular approach for its self-learning, generalization and non-linear mapping abilities. But those algorithms above need a great deal of statistic information and transcendent knowledge, and it is complicated to calculate and train in intricate instances. However there are many factors that influenced the precision of load forecasting directly or indirectly, and it is hard to determine the uncertainty between load and various factors. Therefore, the forecasting process has become even more complex, and more accurate forecasts are needed. Now data mining a newly emerging technology can solve the uncertainty that arises from inexact, noisy, or incomplete information. It provided an effective way for us to solve the difficulties.

Rough set theory as the most typical algorithm of data mining has been applied in expert system, decision support systems, and machine learning. In machine learning rough set is used to extract decision rules from operation data [4]; in neural network, rough set is used in knowledge discovery, data pre-processing [5]-[7] and modelling knowledge-based neural network [8][9]. In order to improve the performance of load forecasting, a novel model integrated with a fuzzy neural network for short-term load forecasting are presented in this article. In this model rough set is applied to extract the relevant domain knowledge to the load, and then the network structure and initial weights are auto-adjusted by the knowledge encoded in the neural network. The simulation results show that the prediction accuracy is improved by applying the method on a real power system.

The article is organized as follows: Section 2 and 3 recall elementary concepts of fuzzy neural network and rough sets; The principle of proposed rough fuzzy neural network is described in section 4; and the experimental results and conclusions are presented respectively in section 5 and 6 respectively.

II. FUZZY NEURAL NETWORK

Neural network is widely used in load forecasting, but there are some disadvantages in conventional neural network as follows: 1) It is hard to determine the structure of network, including the number of neurons and the layers of network. 2) It has bad convergence and tends to get into local minimum. 3) It can't offer a clear concept of the transferring way of knowledge in the neural network. Fuzzy neural network (FNN), integrated with fuzzy theory and neural network, is a more intelligent system that can improve the learning ability and computation speed of the neural network effectively. A brief description of the fuzzy neural network is given in the following parts.

A. Input Vector

The input vector of fuzzy neural network can be either numerical or linguistic. Let the inputs for the i-th sample F_i be mapped to the corresponding three-dimensional feature space of $\mu_{Low}(F_i), \mu_{Medium}(F_i), \mu_{High}(F_i)$ which are denoted by L_i, M_i and H_i respectively. When the input feature is numerical, its membership values for the π–sets is

$$\pi(F_j;c_i,\lambda_i)=\begin{cases}2(1-\dfrac{\|F_j-c_i\|}{\lambda_i})^2, & \dfrac{\lambda_i}{2}\le\|F_j-c_i\|\le\lambda_i\\ 1-2(\dfrac{\|F_j-c_i\|}{\lambda_i})^2, & 0\le\|F_j-c_i\|\le\dfrac{\lambda_i}{2}\\ 0 & \text{otherwise}\end{cases}$$

$j=1\ldots n,\ i=L,M,H$ (1)

Where λ (>0) is the radius of the π-function with c as the central point.

B. Output Vector

The output vector of fuzzy neural network are a series if-then rules, the i-th rule is discribed as

IF A_i and B_i $\rightarrow$ $f(x,y)$

Premise x_0 and y_0 result z

The final defuzzified outputs of fuzzy neural network is

$$y=\sum_{i=1}^{l} f_i \prod_{j=1}^{n}\mu_{R_{ij}} \Big/ \sum_{i=1}^{l}\prod_{j=1}^{n}\mu_{R_{ij}} \qquad (2)$$

$$f_i: X\times Y\rightarrow Z \qquad \mu_{R_{ij}}=\bigvee_{i=1,j=1}^{i=l,j=n}\mu_{A_i}\wedge\mu_{B_j}$$

Where l is the number of rules, $x_1 \ldots x_n$ is the input vector, y is the output vector, $\mu_{R_{ij}}$ denotes the membership of the rule R_{ij} which is the j-th premise of the i-th rule of the rule sets.

III. ROUGH SET THEORY

Rough set theory, presented by Z.Pawlak in 1982, comes to be a new mathematic tool to manage inexact or incomplete information. Here, we recall necessary rough set notions used in this paper.

An information system S is a quadruple $\langle U, A, V, f \rangle$, where $U=\{x_1, x_2, \ldots, x_n\}$ is a finite set of objects, and A is a finite set of attributes. The attributes in A are further classified into two disjoint subsets: the conditional attributes C and the decision attribute D, such that $A = C \cup D$ and $C \cap D = \varnothing$. $V = \bigcup_{a \in A} V_a$ is a set of attribute values, where V_a is the domain of attribute a. f: $U \times Q \rightarrow V$ is an information function that assigns particular values from domains of attributes to objects, such that $f(x_j, a) \in V_a$ for all $x_j \in V_a$ and $a \in A$.

A. Indiscernibility Relation

Given an information system $S = \langle U, A \cup \{d\} \rangle$, let B be a subset of A, and x, y are members of U. A binary relation IND (B) called an indiscernibility relation is defined as

$$\mathrm{IND}(B) = \{(x,y) \in U \times U : f(x,a) = f(y,a) \ \ \forall a \in B\} \quad (3)$$

Objects x and y are indiscernible form each other by attributes from B.

B. Reducts and Core

An attribute $b \in B (\subseteq A)$ is dispensable in B with respect to D, if it satisfied $\mathrm{POS}_B(D) = \mathrm{POS}_{B \setminus \{b\}}(D)$; otherwise b is an indispensable. Set B is called a reduct of A written as RED (A), if (4) is tenable.

$$\mathrm{IND}(B) = \mathrm{IND}(A) \ \text{ and } \ \forall b \in B \ \mathrm{IND}(B - \{b\}) \neq \mathrm{IND}(A) \quad (4)$$

A reduct is a minimal set of attributes from A that preserves the partitioning of the universe, and hence the ability to perform classifications as the whole attribute set A does, and it is usually not unique. The core of A is defined as $\mathrm{CORE}(A) = \cap \mathrm{RED}(A)$.

C. Discernibility Matrix and Discernibility Function

The discernibility matrix of an information system is an $n \times n$ matrix with entries $c_{i,j}$ as given below.

$$c_{ij} = \{a \in A : f(x_i, a) \neq f(x_j, a)\} \quad (5)$$

Discernibilty function f_D is a Boolean function of m Boolean variables $\overline{a_1}, \ldots, \overline{a_m}$ corresponding to the attributes $a_1 \ldots a_m$, respectively, and defined as follows

$$f_D^{x_i} = \wedge \{\vee(c_{ij}) : 1 \leq j \leq n, j \neq i, c_{ij} \neq \varnothing\} \quad (6)$$

D. Dependency Factor

For two attribute sets $B, C \subseteq A$, the dependency factor of B to C is

$$\gamma_C(B) = {card(\mathrm{POS}_C(B))} \Big/ {card(U)} \quad (7)$$

Where card denotes cardinality of the set, and

$\mathrm{POS}_C(B) = \bigcup_{X \in U / IND(B)} \underline{apr}_C(X)$

called a positive region of partition U/B with respect to C, $\underline{apr}_C(X) = \cup\{Y \in U / \mathrm{IND}(C) : Y \subseteq X\}$ is the lower approximation.

E. Decision Rules

Let $S = \langle U, A \cup \{d\} \rangle$ be a decision system and let $V = \cup\{V_a \mid a \in A\} \cup V_d$, decision rules is defined as

$$((a_{i_1} = v_1) \wedge (a_{i_2} = v_2) \wedge \ldots \wedge (a_{i_k} = v_k)) \Rightarrow d = v_d \quad (8)$$

Where $a_{i_1}, \ldots, a_{i_k} \in A, v_j \in V_{i_j}$, $1 \leq j \leq k, v_d \in V_d$, $i_1, \ldots, i_k \in \{1, \ldots, n_A\}$, $i \neq j, i_j \neq i_i$.

IV. ROUGH FUZZY NEURAL NETWORK

There are many influencing factors exist in the information system. Some of them are correlated; and some are independent. If all these factors are used as inputs of neural network directly, it will not only result in complicated structure of network, but long leaning time and inaccurate prediction. Actually we only need a certain factor or some of the factors to achieve the task.

A. Rough Set and Fuzzy Set

Rough set and fuzzy set can both express the heuristic rules of knowledge base effectively. But the ability of knowledge expression and inference of fuzzy set rely on the priori knowledge. Moreover it can't simplify knowledge, while the priori knowledge are always redundant, which would result in enormous structure and bad inference of neural network. Rough set can eliminate the unnecessary knowledge by attribute reduction, which have advantageous to classify and simplify information. But the calculation of rough set is based on the fundamental concepts of upper approximation, lower approximation, positive region, negative region and boundary region that would lead to biases by the problem of boundary simultaneously, and it can be made up by fuzzy neural network. So rough set and fuzzy neural network can work in coordination.

In order to find the factors correlating directly to the load, and improve the performance of load forecasting, rough sets combined with fuzzy neural network are applied to constitute a novel knowledge-based rough fuzzy neural network (RFNN) with three layers for load forecasting, in which rough set is used to deal with the inputs of the rough fuzzy neural network. The best reduct found in the attribute reduction algorithm is applied as the domain knowledge to design the structure and the initial weight of the rough fuzzy neural network.

A. Attribute Reduction Algorithm

Given a decision system $S = \langle U, A \cup \{d\} \rangle$. We use eight kinds of weather factors i.e. clouds, wind speed, wind direction, maximum temperature, minimum temperature, humidity, air pressure and rainfall as the conditional attributes and the historical load on the predicted day d as the decision attribute. The load and weather factors are normalized to the same range of values as follow.

$$X = (x - L_{avg}) / (L_{\max} - L_{\min})$$

where L_{avg}, L_{max} and L_{min} denote the average, maximum and minimum value of the vector x respectively, and the conditional attributes and decision attribute are fuzzified according to their characters.

An $n \times m$ -dimensional pattern $F_i = [F_{i1}, F_{i2}, ..., F_{im}]$ of an attribute i is represented as a $n \times n_k$-dimensional fuzzy vector $[\mu_{L(F_{ij})}(F_i), \mu_{M(F_{ij})}(F_i), ..., \mu_{H(F_{ij})}(F_i)]$, i=1...n, j=1...n_k. n_k is the total number of fuzzy conditional attributes of the attribute set; m and n are the number of attributes in conditional attribute set and objects in universe respectively. Let the attribute-value decision table be consistent, that is, every objects x and y are discernible with respect to the values of their attributes from A.

The absolute distance between each pair of objects is computed along each fuzzy attribute L_j, M_j, ..., H_j for all j (j=1...n_k). Modify (5) to directly handle a real-valued attribute table consisting of fuzzy membership values. We define the individual of discernibility matrix as follow.

$$c_{ij} = \{c \in C : | f(x_i, c) - f(x_j, c) | > Th\} \quad i, j = 1 \ldots n \qquad (9)$$

Where Th ($0.5 \le Th \le 1$) is a threshold depended on the inherent shape of the membership function. Moreover, the type of threshold also enables the discernibility matrix to contain all the representative points [9].

According to (6), it is easy to know that all the reducts of the attribute-value decision table are contained in $f_D^{x_i}$. Whereas we are just interested in those reducts relevant to the task with few attributes and better discrimination. So the best reduct should be found among all the reducts, which are a problem of NP-hard and a process of optimal searching. It is very suitable for using genetic algorithm. Because all the information to differ x_i and x_j are contained in distinguish matrix referring to (5), and the reduct can substitute the whole conditional attribute set and not change the dependent relation and discrimination. Then the best reduct should cover the discernibility matrix as much as possible; And if any element of the reduct were absent, it would not be an exact rule set reasoned from the decision table to make decision; Furthermore a simple and clear decision rule should have few antecedents, i.e. the conditional attributes contained in the reduct should reach the minimum. For the ideas discussed above, we define the fitness function with three terms. The first is number of attributes in the reduct should be the least; the second is the coverage of the reduct over discernibility matrix should be larger; and the last is the approximation of the reduct to the whole conditional attributes should be better. The process of optimum searching is to maximize the forenamed two terms and minimize the last term. The maximizing of the first terms would reduce the third term accordingly. There are five main steps involved in the attribute reduction process.

Step 1. Generate discernibility matrix, calculate the core and remove the items including core.

Step 2. Create random populations of n individuals. Each individual contains all the attributes in core, and is consistent with the definition of reduction. All the individuals are represented as chromosome. Each chromosome is a string of bits, 0 or 1. Where 1 represents the attribute on that bit is included in the reduct, 0 otherwise.

Step 3. Given a fitness function as follow.

$$F = \alpha_1 / L(R) + \alpha_2 \sum_{i=1}^{kl} s_i \Big/ kl - \alpha_3 (\gamma(C) - \gamma(R)) / \gamma(C)$$

$L(R)$ is the numbers of 1 in the string. When there exist an intersection between reduct and a certain individual of discernibility matrix, s_i is set to 1, otherwise 0, and kl is the number of the individuals in the matrix. $\gamma(C)$ is the dependency factors of C to D. $\alpha_1 \ldots \alpha_3$ are positive random weights less than 1. While the fitness function reaches the maximum, the number of attribute in the reduct gets to the optimum. Evaluate the fitness value of each individual according to the given fitness function.

Step 4. Create next generation with larger fitness by using selection, crossover and mutation.

Step 5. If a predefined number of generations are achieved, then stop and return the best individual in current population, else go to step 3. In the genetic process, the optimal individual is saved.

By using the attribute reduction algorithm proposed in the previous section, only relevant factors are saved, which reduces the attribute space substantially.

B. Rough Fuzzy Neural Network

There are two types of neuron in the rough fuzzy neural network, as depicted in Fig.1. One is the traditional neuron that is used to manage the data of historical load; the other is rough neuron that is used to manage the blurry, inexact discrete vectors, such as weather, type of day, random effects. In the following part, we mainly discuss the computation of rough neuron.

1.Input Vector

Let the history load on day d-1, d-2, d-7 and d-14 be the input vector of conventional neuron. The weather factors corresponding to the minimum reduct deduced from attribute reduction algorithm are treat as the input vectors of rough neuron.

2. Structure and Initial Connection Weights of Rough Fuzzy Neural Network

In conventional neural network all the neurons are connected in a fixed way, and the connection weights are initialized as a series random numbers. With the different structure and different connection weights the outputs of the neural network are distinct accordingly, whereas the knowledge-based neural network has a specific topology. During the structure designing of the network, the crude domain knowledge is considered, which are useful for us to understand the transferring way of knowledge in the neural network. Moreover, during learning neural network are always searching for a set of connection weights that minimizes the distances between the target vector and the actual output of the network. So if the connection weights are initialized to those better weights, and an optimum

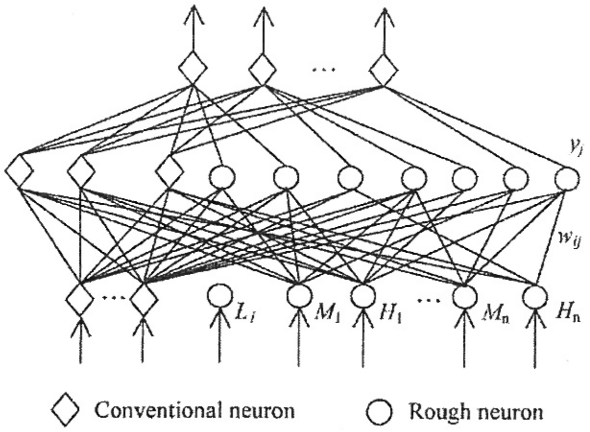

Fig.1. Construction of Rough Fuzzy Neural Networks

network structure is obtained by adjusting the number and the links of nodes, the searching space may be reduced and learning therefore becomes faster. Based on the above discusses, we use the extracted knowledge gained in the attribute reduction to design the rough fuzzy neural network.

1) Structure of the network

Consider an attribute-value decision table according to the reduct; it contains l objects corresponding to the representative points of each class of the decision attribute. l denotes the number of the classes of the decision attribute. This decision table is only used to structure and initial weights designing procedure, during the training phase the network learns from the original $n\times n_k$-dimensional training set. Using (5) and (6) to compute the discernibility matrix (M (S)) and the conjunctive normal forms (CNF) of the discernibility function $f_D^{x_i}$, each normal form corresponds to a decision rule. In this article a rough fuzzy neural network are used to describe a rule accordingly.

a) Neuron of the Input Layer

The input layer consists of n_k attribute values corresponding to the fuzzy conditional attributes.

b) Neuron of the Hidden Layer

The hidden nodes model the disjuncts ($\vee$), each disjunct corresponds to one neuron in the hidden layer. Only those input attributes that appear in a disjunct are connected to the appropriate hidden node. The number of the nodes in the hidden layer equals the number of the conjuncts ($\wedge$).

c) Neuron of the Output layer

Each conjunct ($\wedge$) is modeled at the output layer by joining the corresponding hidden node. The number of the nodes in the output layer equals the number of classes of the decision attribute.

The traditional neurons and the rough neurons are connected with each other in a fixed way as in a traditional neural network.

2) Connection Weights

a) The weight between a hidden node (i) and an output node (k) is $w_{kj}^{1}=\alpha/_{numc}+\varepsilon$, where α is the dependency factor, *numc* (*numc*≥1) is the number of conjuncts in a rule and ε is a small random number.

b) The weight between an attribute a_i (where a corresponding to fuzzy linguistic vectors in a fuzzy set) and a hidden node j is $w_{ja_i}^{0}=\beta/_{numd}+\varepsilon$, where β is the initial weight between the hidden nodes and the output nodes. *numd* (*numd*≥1) is the number of attributes connected by disjuncts.

The sign of the weight is set to positive (negative) if the corresponding entry of M (S) in row k, column a_i is 1 (0). The initial weights of the traditional neuron are set to a set of random numbers in this network. All the neurons are updated by the following equations.

$$w_{ji}^{new}=w_{ji}^{old}+\Delta w_{ji}^{new}$$
$$\Delta w_{ji}^{new}=\eta\cdot out_i\cdot err_i\cdot f'(in_i)+mc\Delta w_{ji}^{old},$$

Where f refers to the neuron activate function; η and mc are the learning rate and momentum factor respectively; *out* and *in* correspond to the output and the input vector accordingly; *err* represents the output error of the neurons.

3) Output Vector

The output vector are a series rules, let i-th rule be

IF influencing factor a_{i1} is v_1

AND influencing factor a_{i2} is v_2

... ...

AND influencing factor a_{ik} is v_k

THEN predicted load is y_i

The predicted loads are gained after the output vector are defuzzified by (2).

V. EXPERIMENTAL EXAMPLE ON LOAD FORECASTING

In this section, we describe the example, where proposed method is applied to real data to forecast the workday load. The historical load and weather of a city in 2001 are used as the training sets. Weather factors on d, d-1, d-2 and d-7 are chose as the conditional attributes; load on d is treated as the decision condition, d is forecasting day. All the attributes are fuzzified according to their characters, Let the load data be divided into l classes, every class corresponds to a rough fuzzy neural network (RFNN); all the outputs of these sub-networks compose the final results. The error function MAPE (mean absolute percentage error) is used to analyze the output error.

$$\mathrm{MAPE}(\%)=\frac{1}{N}\sum_{k=1}^{N}\frac{|t(k)-a(k)|}{t(k)}\times 100$$

where a is the output vector, t is the target vector. The principle of early stopping is used to avoid the generalization depressing caused by over-learning.

TABLE I
THE OUTPUT ERROR OF RFNN IN 24 HOURS

Time	1:00	2:00	3:00	4:00	5:00	6:00
MAPE (%)	2.16	1.38	0.65	0.08	0.03	0.2
Time	7:00	8:00	9:00	10:00	11:00	12:00
MAPE (%)	0.59	3.31	1.03	1.49	0.89	3.11
Time	13:00	14:00	15:00	16:00	17:00	18:00
MAPE (%)	0.34	0.75	1.38	1.60	0.45	0.26
Time	19:00	20:00	21:00	22:00	23:00	24:00
MAPE (%)	0.821	0.137	0.13	0.445	0.74	0.69

The predicted errors in 24 hours of the proposed method in this paper are presented in Table I. The maximum predicted error is about 3.23%, and the minimum is about 0.02%, it satisfied the requirement of load forecasting. In

Table II, the results from testing the proposed method compared with conventional fuzzy neural network (FNN) are presented, where the numbers represent MAPE for five days. The fairly good performance of the maximum predicted error of FNN is 3.271%, however the proposed method has yield a better performance of prediction error lower than 2.832%, the precision has improved at least 14%.

TABLE II
COMPARISON BETWEEN RFNN AND FNN BY MAPE

Date	12~16 Mar.	21~25 May	6~10 Aug	17~21 Dec.
FNN	1.372	1.886	3.271	1.746
RFNN	1.28	1.557	2.732	1.105

TABLE III
EXECUTION TIME OF RFNN AND FNN

Algorithms	RFNN		FNN
	A1	A2	
Execution Time (s)	326	127	648

A1and A2 in TABLE III is referring to the attribute reduction algorithm and the rough neural network training respectively.

We also show the comparison results between FNN and RFNN in Fig.2. From the results, we can see that the proposed method obtained higher prediction accuracy of the prediction errors than traditional FNN, after the input vectors and the link weights are pre-processed by rough sets. The reason is that by using rough set theory the unnecessary information is eliminated, the inherent relations in information are dug out at the same time, and various information resources are utilized sufficiently. Moreover, The redundant of fuzzy neural network and the boundary problem of rough set are conquered by combining rough set with fuzzy neural network, which heightened the accuracy of rule reasoning and improved the accuracy of load forecasting consequently.

In this paper we selected 79 fuzzy attributes including weather factors and electrical load as the elementary elements of decision table. After using the attribute reduction algorithm the total number of the attributes decrease to 27, then the attribute value of input vectors are just correlated directly with the target. Furthermore the antecedents of decision rules have determined the links of the network that is sparser than the full connection of traditional neural network. The structure of neural network

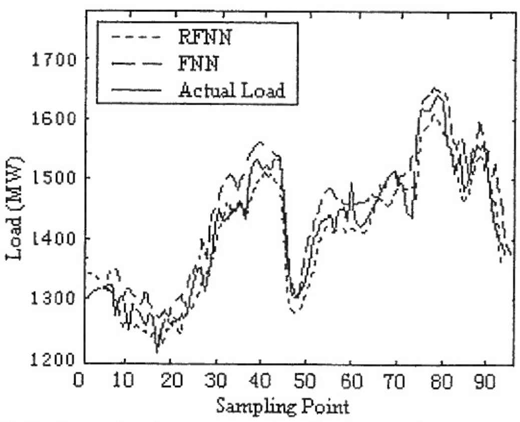

Fig.2. Comparison between actual and load curves forecasted by FNN and RFNN

becomes simpler and clearer, so the probability to immerse in local minimum decreases and the training speed becomes faster, for example, the training epochs decreases from 11826 to 3146 by RFNN. Referring to the comparison of execution time in Table III, the total execution time of proposed method is less than the traditional FNN, although the attribute reduction algorithm has took up most part of the execution time, the time expended on RFNN training has decreased distinctly. As we can see from the results, our method reaches the precision of load forecasting, and obtains higher accuracy than the conventional method.

VI. CONCLUSION

A new hybrid knowledge-based model for short-term load forecasting integrated with rough set theory and fuzzy neural network has been presented in this paper. The discernibility matrix, discernibility function and reducts are used to extract the domain knowledge that is the basis for network structure and initial connection weight designing. For its better performance of optimal searching, genetic algorithm is applied to search the minimum reduct in attribute reduction algorithm. Then the proposed model has been tested and compared with conventional fuzzy neural network in a real power system. The results show that the proposed model provides a more accurate forecast than the conventional one, because it has an optimized network structure and reduced searching space. Moreover, it becomes easier to understand the transferring way of knowledge in the neural network. This paper shows that the proposed method is promising for load forecasting in power system.

The authors gratefully acknowledge the contributions of Ian X. Austan, Ann Burgmeyer, C. J. Essel, and S. H. Gold.

VII. REFERENCES

[1] A.D.Papalexopoulos and T.C.Hesterberg, "A regression-based approach to short term system load forecasting, " IEEE Transaction on Power Systems, vol.5, no.4, pp. 1535-1547, Nov.1990.

[2] S.E.Papadakis, J.B.Theocharis, "A novel approach to short-term load forecasting using fuzzy neural networks, " IEEE Transactions on Power Systems, vol.13, no.2, pp. 480-493, May 1998.

[3] James W. Taylor and Roberto Buizza, "Neural network load forecasting with weather ensemble predictions, " IEEE Transactions on Power Systems, vol.17, no.3, pp. 626-633, August 2002.

[4] R.Slowinski, Ed., Intelligent decision support, "Handbook of applications and advances of the rough sets theory. " Dordrecht: Kluwer, 1992.

[5] A.Czyzewski and A. Kaczmarek, "Speech recognition systems based on rough sets and neural networks, " in Proc.3rd Wkshp. Rough Sets and Soft Computing (RSSC'94), San Jose, CA, 1994, pp. 97-100.

[6] Qiu-Dan Li, Zhong-Xian Chi, Wen-Bing Shi, "Application of rough set theory and artificial neural network for load forecasting, " Proceeding of the First International Conference on Machine Learning and Cybernetics, Beijing, 2002, pp. 1148-1152.

[7] Lambert-Torres, G., "Application of rough sets in power system control centre data mining," Power

Engineering Society Winter Meeting, 2002 IEEE, vol.1, 2002, pp. 627–631.
[8] Mohua Banerjee, Sushmita Mitra, and Sanka.K.Pal, "Rough Fuzzy MLP: knowledge encoding and classification," IEEE Transactions on neural networks, vol.9, no.6, pp. 1203-1217, Nov.1998.
[9] Sankar K. Pal, Sushmita Mitra, "Rough Fuzzy MLP: Modular Evolution, Rule Generation and Evaluation," IEEE Transactions on Knowledge and Data Engineering, vol.15, no.1, pp. 14 –25, Jan. 2003.