PAPER

Neuro-fuzzy Recognition System for Detecting Wave Patterns Using Wavelet Coefficients

Sung Hoon Jung† *and* **Doo Sung Lee**†, *Nonmembers*

SUMMARY Recognition of specified wave patterns in one-dimensional signals is an important task in many application areas such as computer science, medical science, and geophysics. Many researchers have tried to automate this task with various techniques, recently the soft computing algorithms. This paper proposes a new neuro-fuzzy recognition system for detecting one-dimensional wave patterns using wavelet coefficients as features of the signals and evolution strategy as the training algorithm of the system. The neuro-fuzzy recognition system first trains the wavelet coefficients of the training wave patterns and then evaluates the degree of matching between test wave patterns and the training wave patterns. This system was applied to picking first arrival events in seismic data. Experimental results with three seismic data showed that the system was very successful in terms of learning speed and performances.
key words: Neuro-fuzzy, Pattern Recognition, Evolution Strategy, Wavelets

1. Introduction

Recognition of specified wave patterns in one-dimensional signals is one of the important tasks in signal processing areas, such as voice recognition in computer science, wave detection in medical science, and event picking in geophysics. To automate this process many methods have been introduced by many researchers with various techniques, especially with soft computing recently [1]–[9].

This paper proposes a new neuro-fuzzy approach for detecting specified one-dimensional wave patterns. Our neuro-fuzzy recognition system consists of three main modules: a neuro-fuzzy decision module, an evolution strategy learning module, and a wavelet feature extraction module. The neuro-fuzzy decision module that plays a major role in our system evaluates the degree of matching between the test wave patterns and the training wave patterns. Of course, the neuro-fuzzy decision module must be trained with the training wave patterns prior to the decisions. The neuro-fuzzy decision module has neural network structures and their link weights between the input layer and hidden layer, and hidden layer and output layer have fuzzy membership functions [10]. The evolution strategy learning module trains the neuro-fuzzy decision module by optimizing the training parameters of the neuro-fuzzy decision module. This training method prevents the neuro-fuzzy decision module from getting stuck at a local optimum. Performances of recognition systems greatly depend on which features are used. We have used the wavelet coefficients, calculated by the wavelet feature extraction module, as the features of the signals.

The overall application procedures of the neuro-fuzzy recognition system are as follows; 1) defines the training patterns, 2) normalizes the training patterns and transforms into wavelet coefficients, 3) trains the neuro-fuzzy decision module with the wavelet coefficients of training wave patterns, 4) finally, decides how much the test patterns are matched to training wave patterns by the neuro-fuzzy decision module.

We apply our system to picking the first arrival events in seismic data. This picking process is a very laborious and time consuming task in geophysics [7]–[9]. McCormack et al. [7] and Veezhinathan et al. [8] have used back-propagation neural networks (BPNN) as processing algorithms. While McCormack et al. [7] have used the seismic data itself as the features of the data, Veezhinathan et al. [8] used four signal attributes of the seismic data as features †. Thus, the method of Veezhinathan et al. is faster in training than that of McCormack et al. These two methods takes much time to train because they used BPNN. Moreover, their methods can get stuck at a local optimum in training. Chu and Mendel [9] introduced a new method using fuzzy logic systems. Their method produced nearly the same results as previous works in spite of fast training. This method can also fall into a local optimum because they used a back-propagation algorithm in training.

Our system was tested with three seismic data. Experimental results showed that our method was excellent than previous methods in terms of learning speed and recognition results. This paper is organized as follows. Section 2 describes the structure and operations of our system. The picking first arrival events of seismic data is described in section 3. In section 4, we show the experimental results with discussions. This paper concludes in section 5.

†The authors are with the Faculty of School of Information and Computer Engineering, Hansung Univ., Seoul, 136-792, Korea

†The four signal attributes are the maximum amplitude, mean power level, power ratio and envelope slope.

2. Neuro-fuzzy Recognition System

Our neuro-fuzzy recognition system (NFRS) consists of three main modules; a neuro-fuzzy decision module (NFDM), an evolution strategy learning module (ESLM), and a wavelet feature extraction module (WFEM). Figure 1 shows the structure of NFRS. The overall recognition sequences of the NFRS is given in Algorithm 1. Each module has its own functions for recognition processing. The NFDM as a major module of the NFRS makes a decision how much the test wave are matched to the training wave patterns. Of course, the NFDM must be trained with the training wave patterns prior to the decisions. The function of the ESLM is to train the NFDM by optimizing the parameters of the NFDM. The WFEM performs the wavelet transforms and produces the wavelet coefficients, which is used as the features of the waves. The wavelet coefficients represent the features of the waves in a multiresolution, shape decomposition fashion [11], [12]. Original waves are successively decomposed into components of lower resolution, while the high frequency components are not analyzed any further [13]. By this multiresolution decomposition, final wavelet coefficients have an average value of the waves at the lowest resolution, an average value and a difference value at the 2nd lowest resolution, and so on. As a result, the wavelet coefficients have the informations of the shape as well as frequency of the waves in a multiresolution fashion [13]. Based on the above observation, we concluded that using wavelet coefficients as features of the waves for detection of a specified wave are more useful than using the raw waves or using other features extracted from the raw waves. Moreover, the features can be enhanced by weighting of the wavelet coefficients. For example, denosing effects are achieved by weighting the high frequency of the wavelet coefficients [14], [15].

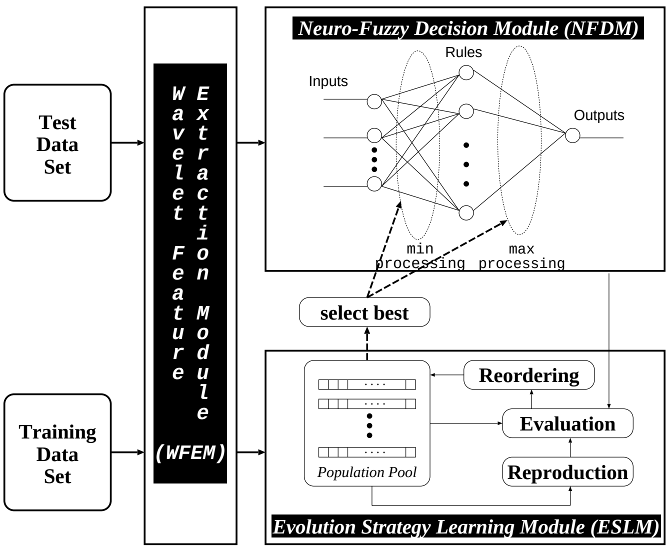


Fig. 1 Structure of neuro-fuzzy recognition system (NFRS)

Algorithm 1 Operations of NFRS

// PP : population pool composed of N parents and N offsprings//
// f : the fitness of the best individual//
// f_t : fitness for terminating the training //

initialize each module and set training data
  initialize NFDM
  initialize ESLM
  set training data and normalize
  calculate wavelet coefficients of all normalized training data
train the NFDM with ESLM
  do
    evaluate all N parents
    mutate all N parents and generate N offsprings
    evaluate all N offsprings and insert them to PP
    select good N individuals from the PP and replace them to parents
  until $f < f_t$
set the parameters of the best individual to NFDM
apply each test data to the NFDM and make a decision
  read the test data and normalize
  calculate wavelet coefficients of the normalized test data
  input the wavelet coefficients to the NFDM
  decide how much the test data is matched to the training data

---

For wavelet transforms, some wavelets such as Haar, B-Spline, Coiflet, and Daubachie wavelets are available [13], [15]. For the WFEM, we used Daubachie's order-4 wavelet because this wavelet shows considerably good performance in denosing and compression of data. We used the order-4 wavelet because it was found from experiments that the performance of our neuro-fuzzy system was decreased as the order of Daubachie's wavelet increase. This may be because as the order of Daubachie's wavelet, the local properties of data are more and more destroyed.

The neural network structures of the NFDM are composed of inputs, rules (in other words, hidden), and output layers. We call the links between the inputs and the rules layers *input links* and the links between the rules and the outputs layers *output links*. Each link of the input and the output has a weight represented by one linguistic term (for example, *NB* (Negative Big), *NS* (Negative Small), and so on) as a fuzzy membership function. The input and output links represent conditional and conclusion parts of fuzzy rules, respectively. For example, let us consider a simple structure, which has only two input nodes, x_1 and x_2; two rule nodes, h_1 and h_2; and one output node, y_1. Let the input links be $W_{i,j}$, $i \in \{x_1, x_2\}$, $j \in \{h_1, h_2\}$ and output links be $V_{j,k}$, $j \in \{h_1, h_2\}$, $k \in \{y_1\}$. Let the weights of input and output links be represented by three linguistic terms: *NB* (Negative Big), *ZO* (Zero), and *PB* (Positive Big). If W_{x_1,h_1} is *NB*, W_{x_2,h_1} is *ZO*, and V_{h_1,y_1} is *NB*; W_{x_1,h_2} is *ZO*, W_{x_2,h_2} is *PB*, and V_{h_2,y_1} is *ZO*, then the input and output links represent the following two fuzzy rules.

$$r_1:\ if\ x_1\ is\ NB\ and\ x_2\ is\ ZO,\ then\ y_1\ is\ NB$$
$$r_2:\ if\ x_1\ is\ ZO\ and\ x_2\ is\ PB,\ then\ y_1\ is\ ZO$$

Operations of the NFRS are as follows. Input values are applied to input links (more specifically, linguistic terms of input links). The matching rates between input values and linguistic terms of input links are used to generate the outputs of rule nodes. With the matching rates from all input nodes, rule nodes do $t-norm$ operation and generate real values, which are used as $\alpha - cut$ values for output links. These $\alpha - cut$ values are applied to the linguistic terms of the output links. Output nodes do $t-conorm$ operation with the $\alpha-cut$ linguistic terms. Since the output of $t-conorm$ operation is fuzzy value, the output nodes do defuzzification for generating crisp values. In our experiments, we used minimum and maximum operations as $t-norm$ and $t-conorm$ operations, respectively. The simple minimum and maximum operations have been widely used in fuzzy reasoning.

With the simple neuro-fuzzy structure, previously mentioned in this section, we describe the neuro-fuzzy operations in detail. Let membership functions of the linguistic terms of an input link ij and an output link jk be $\mu_{ij}(\cdot)$ and $\mu_{jk}(\cdot)$, respectively. Then the outputs of rule nodes are given as:

$$h_1^o = \min\{\mu_{x_1,h_1}(x_1), \mu_{x_2,h_1}(x_2)\} \quad (1)$$
$$h_2^o = \min\{\mu_{x_1,h_2}(x_1), \mu_{x_2,h_2}(x_2)\}, \quad (2)$$

where h_1^o and h_2^o are outputs of the nodes h_1 and h_2, respectively. As described before, these values are used as $\alpha - cut$ for output links.

The membership functions of linguistic terms of output links are cut according to the $\alpha - cut$ values. With these cutting membership functions, output nodes do the max operation and do defuzzification with the result of max operation. For defuzzification, we used level grading defuzzification method [16]. This defuzzification is useful for the NFRS because this method is effective where linguistic terms are not equally distributed. Initially, the linguistic terms of the NFRS are equally distributed, but they are different each other after training. Since this defuzzification incorporates the max operation into the defuzzification [16], output nodes do only this defuzzification without max operation.

This defuzzification [16] is defined as:

$$y_1^o = \frac{\sum_{p=1}^{2} \mu^{-1}_{h_p,y_1}(1)\ m_c(h_p^o)}{\sum_{p=1}^{2} m_c(h_p^o)}, \quad (3)$$

where y_1^o is the output of node y_1 and $m_c(\cdot)$ is the measure of certainty. In [16], the measure of certainty is defined as:

$$m_c(\alpha) = \frac{\alpha}{\mu_{\tilde{A}}^{-1}(\alpha)_{max} - \mu_{\tilde{A}}^{-1}(\alpha)_{min} + 1}, \quad (4)$$

where α is a $\alpha - cut$ value, $\tilde{A}$ is a linguistic term, and $\mu_{\tilde{A}}^{-1}(\alpha)_{max}$ and $\mu_{\tilde{A}}^{-1}(\alpha)_{min}$ are maximum and minimum values of the function $\mu_{\tilde{A}}^{-1}(\alpha)$, respectively. Finally, the outputs of the NFRS is given as:

$$y_1^o = \frac{\sum_{p=1}^{2} \frac{\mu^{-1}_{h_p,y_1}(1)\ h_p^o}{\mu^{-1}_{h_p,y_1}(h_p^o)_{max} - \mu^{-1}_{h_p,y_1}(h_p^o)_{min} + 1}}{\sum_{p=1}^{2} \frac{h_p^o}{\mu^{-1}_{h_p,y_1}(h_p^o)_{max} - \mu^{-1}_{h_p,y_1}(h_p^o)_{min} + 1}}. \quad (5)$$

The performance of a neuro-fuzzy system is strongly dependent on the number of hidden nodes (in other words, the number of fuzzy rules). If the number of hidden nodes is too small, then training time will be short but the training can be poor by under fitting. Contrarily, if the number of nodes is too large, then training can be good but the training time will be large. In order to automatically decide the number of fuzzy rules, we devised an algorithm that is described in Algorithm 2. In this algorithm, we assume that linguistic terms are equally distributed within the range of each input and output. With the first training data, the first rule is generated as follows. Let the first training data be given as $t_1 = (x_{11}, x_{21}, y_{11})$ and the membership functions of three lingustic terms be represented as $\mu_{NB}(\cdot), \mu_{ZO}(\cdot)$, and $\mu_{PB}(\cdot)$, then the first fuzzy rule r_1 are represented as

Algorithm 2 Setting of the number of rules

```
    // r_j : jth rule //
    // k : the number of generated rules //
    // t_i : ith training data //
    // N : the total number of training data //
    // m_{i,j} : matching rate between t_i and r_j//
    // m^d : matching decision value//
    // M(·,·) : matching function //
    // R(t_j) : rule generation function with training data t_j//
R(t_1) ▷ make first rule for first trainig data
k ← 1
for i = 2 to N ▷ test for each training data
    for j = 1 to k ▷ test for each generated rule
        m_{i,j} = M(t_i, r_j)
    end for
    If max(m_{i,1}, ..., m_{i,k}) < m^d then
        k ← k + 1
        R(t_i) ▷ generate kth rule r_k
    end if
end for
```

$$W_{x_1,h_1} = \arg\max\{\mu_{NB}(x_{11}), \mu_{ZO}(x_{11}), \mu_{PB}(x_{11})\}$$
$$W_{x_2,h_1} = \arg\max\{\mu_{NB}(x_{21}), \mu_{ZO}(x_{21}), \mu_{PB}(x_{21})\}$$
$$V_{h_1,y_1} = \arg\max\{\mu_{NB}(y_{11}), \mu_{ZO}(y_{11}), \mu_{PB}(y_{11})\}.$$

For the other training data, a new fuzzy rule based on the matching rate with previous rules may or may not be generated. In Algorithm 2, the matching function $M(\cdot,\cdot)$ calculates the matching rate between a training data and the generated rule. Let, for example, the second training data be given $t_2 = (x_{12}, x_{22}, y_{12})$, then the matching rate m_{21} is calculated as $M(t_2, r_1) = \min\{\mu_{W_{x_1,h_1}}(x_{12}), \mu_{W_{x_2,h_1}}(x_{22}), \mu_{V_{h_1,y_1}}(y_{12})\}$. If the m_{21} is less than the matching decision value m^d, then a new rule is generated. Otherwise, no new rule is generated for the second training data and third training data will be processed. If the number of generated rules is greater than one, then the maximum value of all matching rates must be compared to the m^d. From this algorithm, we can get fuzzy rules as well as the number of fuzzy rules. In some cases, these fuzzy rules can be used as initial fuzzy rules; however, using these rules may sometimes make ESLM fall into a local optimum. We used the algorithm only for getting the number of fuzzy rules in this paper.

The NFDM must be trained by a learning algorithm before it makes a decision. We used the evolution strategy as learning method of the NFDM. Figure 1 shows the overall block diagram of the ESLM. In order to apply evolution strategy, the optimized parameters and the format of individuals must be defined first. The optimized parameters are:

- linguistic terms of the weights of input and output links (in other words, rules)
- center and standard deviations of the linguistic terms [†].

Those parameters are embedded into the format of individuals. With the simple structure described above, the format of individuals is depicted in Figure 2. The

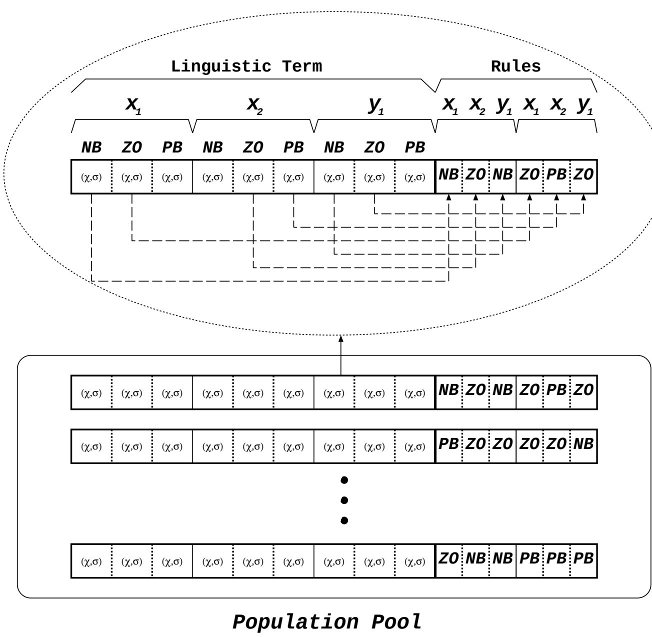


Fig. 2 Individual representation in population pool

linguistic terms in Figure 2 indicate the center and deviation (χ, σ) of each linguistic term of all inputs and outputs. Initially, the centers and deviations (χ, σ) of all individuals are distributed uniformly. Thus, the centers and deviations in same linguistic term have same initial values. During the processing of learning, these values of each individual are updated according to evolution processing. Therefore, although two inputs have same linguistic term, the centers and deviations of those two inputs may be different from each other after training. While the values of linguistic terms of all individuals have real number and are equally distributed initially, the rules of all individuals have integer values and are randomly set within the number of linguistic terms. For example, if linguistic terms *NB*, *ZO*, and *PB* are represented as 0, 1, and 2, respectively, then the rules are set to a random integer number from 0 to 2. As we already mentioned before, we did not use the fuzzy rules generated by the Algorithm 2 as initial fuzzy rules because we observed from experiments that using the generated fuzzy rules as initial fuzzy rules made ESLM fall into a local optimum as we expected.

The ESLM keeps $2N$ individuals (N for parents, N for offsprings) in population pool during run. The number N, of course, must be set by users prior to run. The ESLM evaluates the fitness of each individual in the population pool. Algorithm 3 shows the evaluation procedure of the ESLM. To evaluate the fitness of each

Algorithm 3 Evaluation procedure
// N : the number of training data //
// P : the number of individuals //
// O : the number of outputs of NFDM //
// NFDM^i : NFDM whose links are set by ith individual //
// o_j^k : kth actual output of NFDM^i when jth training data are applied//
// t_j^k : kth target output of jth training data //
// e_i : total sum square error of ith individual//
// f_i : fitness of ith individual //
for $i = 1$ **to** P
  set NFDM^i
  $e_i \leftarrow 0$
  for j=1 **to** N
    apply the wavelet coefficients of jth normalized training data to the NFDM^i
    measure the actual output o_j of NFDM^i
    calculate error, $e_j = \frac{1}{2}\sum_{k=1}^{O}(t_j^k - o_j^k)^2$
  end for
  obtain the mean square error of ith individual, $e_i = \sum_{j=1}^{N} e_j / N$
  set the fitness of ith fitness to $f_i = \frac{1}{1+e_i}$
end for

individual, the ESLM feeds inputs of the wavelet coefficients of a normalized training data to the NFDM, and calculates the errors between actual outputs of NFDM and target outputs of the training data. A mean square error (MSE) for the individual is calculated from the errors for all training data. This mean square error is used for calculating the fitness of the individual. Finally, the

[†]We used bell-shaped fuzzy sets as each linguistic term.

fitness of ith individual is given by:

$$f_i = \frac{1}{1+e_i} = \frac{1}{1+1/2N\sum_{j=1}^{N}\sum_{k=1}^{O}(t_j^k - o_j^k)^2}. \quad (6)$$

From the equation 6, it is obvious that the maximum fitness is one. As shown in Algorithm 1, after evaluation the individuals of parents are mutated with Gaussian noise with zero mean and constant standard deviation to generate offsprings. The ESLM evaluates the fitness of the offsprings and rearranges the parents and offsprings in descending order of the fitness to select next generations. If the best individual, which is always at the first in population pool, has MSE less than a predefined error ϵ, then the training is finished. Since we used the MSE as a parameter of the fitness function, the termination condition of training can also be exposed by the terminating fitness of the training. i.e., $f_t = \frac{1}{1+\epsilon}$.

After training, the parameters of the best individual is set to NFDM. The NFDM estimates matching rates between test patterns and training patterns by fuzzy reasoning. There are many reasoning methods. We have used the max-min inference and the level grading defuzzification method [16]. Therefore, the rule nodes of our NFDM play a role of min-processing and output nodes play a role of max-processing [†]. The fuzzy values of output nodes are converted to crisp values (in other words, matching rates) by level grading defuzzification. The NFRS can use these matching rates for various purposes dependent on application areas.

3. Picking First Arrival Events

In this paper, the NFRS is applied to picking the first arrival events in seismic data. This process has been known as a very tedious and time consuming task in geophysics because the amount of seismic data is usually large and the seismic data contains a lot of noise. In order to automate this processing, some researchers have devised algorithms based on the artificial intelligence techniques, such as neural networks and fuzzy logic [7]–[9].

McCormack et al. [7] applied a back-propagation neural network (BPNN) approach to the recognition. They used a two-layer network containing close to 5,000 input neurons and two output neurons. They represented seismic data as binary image that is a 2-D image with only two grey levels (0 and 1). After training using the binary image, BPNN decides the first arrival events. Since the number of input nodes are too large, their method needs huge training time. Veezhinathan et al. [8] proposed a neural network based recognition method with four signal attributes of the seismic data as features. Their method was faster in training neural networks than that of McCormack et al. [7] because they used only four signal attributes, instead of the data itself. Using fuzzy logic systems, Chu and Mendel [9] introduced a new method for the picking first arrivals. Their systems called back-propagation fuzzy logic systems (BPFLS) were trained by updating parameters such as input and output regional centers of membership functions with back-propagation of errors. They used four signal attributes proposed by McCormack et al. [7] and another new attribute called distance to the guiding function as features. The fifth attribute is a piecewise linear function with four points that are selected by human as first arrival events. Their method produced nearly the same results as the BPNN with faster training. These methods can get stuck at a local optimum because they employed the back-propagation algorithm.

In previous works, a training pattern is composed of three peaks whose center peak was first arrival and the others are not. Since the first and third peaks include the informations of no first arrival patterns, there are no specific training patterns for no first arrival ones. In their methods, as a result, the output values of all training patterns are one. On the other hand, we used two types of training patterns. One is for first arrival patterns and the other is for no first arrival patterns. The training patterns for first arrival ones consist of only one peak that is first arrival. While the training patterns for first arrival ones are given by a user, the training patterns for no first arrival ones are automatically generated by shifting the start and end array index of the first arrival peak to upward or downward. Of course, if the inputs of a training pattern are first arrival, then the output value of the training pattern is given as one. Otherwise, the output value is given as zero. For simplicity, we call the training patterns whose output values are one *one output training patterns (OOTPs)* and those whose output values are zero *zero output training patterns(ZOTPs)*, respectively. The shifting values and the number of ZOTPs per one OOTP are a user parameter. Training patterns are represented as training windows in seismic traces. Since the first arrival events occur within specific duration in traces according to the minimum and maximum speed of seismic wave, the NFRS scans only within the specific duration (we call this duration *searching windows* in traces). The NFRS evaluates the matching rates of test patterns (we call the test patterns *testing windows* in traces) one by one by shifting one array index from the start array index to the end array index of the searching window of a trace. Finally, the NFRS selects the test pattern with a maximum matching rate as a first arrival pattern for a trace. In figure 3, the numbers of testing windows for the trace 3 are some examples of matching rates. Figure 3 shows the training windows, searching windows, and testing windows.

[†]The min-processing and max-processing can be regarded as activation functions in neural networks.

In Figure 3, only some testing windows are depicted

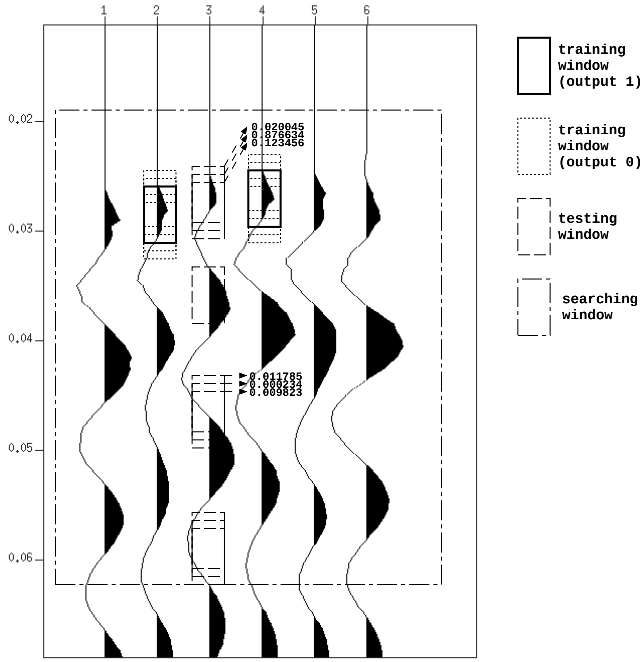


Fig. 3 Training window, searching windows, and testing windows

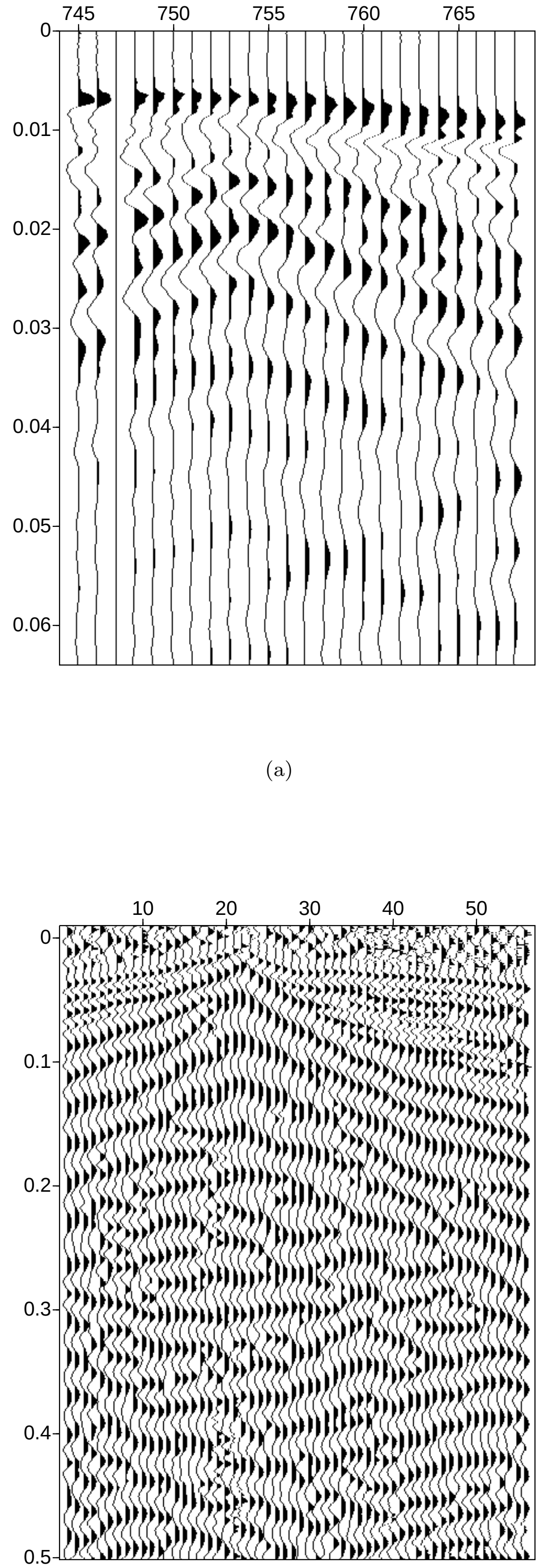
for simplicity. The number of ZOTPs per one OOTP is four, two of them are upward and two of them are downward.

4. Experimental Results

Our system was tested with three seismic records, the first one is relatively low noise and the others are considerably noisy. Figure 4 (a) and (b) show the two seismic records for test. Table 1 shows the parameters of NFRS for test. The number of training data is set to 15, three of them are OOTPs and the others are ZOTPs. The number of linguistic terms is 5 and the number of rules are the same as that of training data. The training of NFDM by ESLM is finished when the MSE of best individual is below 0.00001. The number of individuals of ESLM is set to 20. The standard deviations for mutation of parents in ESLM are set to (max-min) * 0.02 for centers and deviations of bell-type

Table 1 Parameters of NFRS for test 1 and 2

Parameters	Values
# of training data	15 (3: OOTPs, 12: ZOTPs)
# of linguistic terms	5 (*NB, NS, ZO, PS, PB*)
# of rules	15
ϵ	0.001
# of individuals	20
Mutation	individuals : σ = (max-min) * 0.02 rules : $\sigma = 0.5$



(b)

Fig. 4 Seismic records for (a) test 1 and (b) test 2 membership functions † and set to 0.5 for rules. Therefore, offsprings are mutated by adding Gaussian noise with zero mean and the standard deviation.

Table 2 shows the training data and searching windows for test 1 and 2. In Table 2, the '#' means trace

Table 2 Training data and searching windows (time unit: sec.)

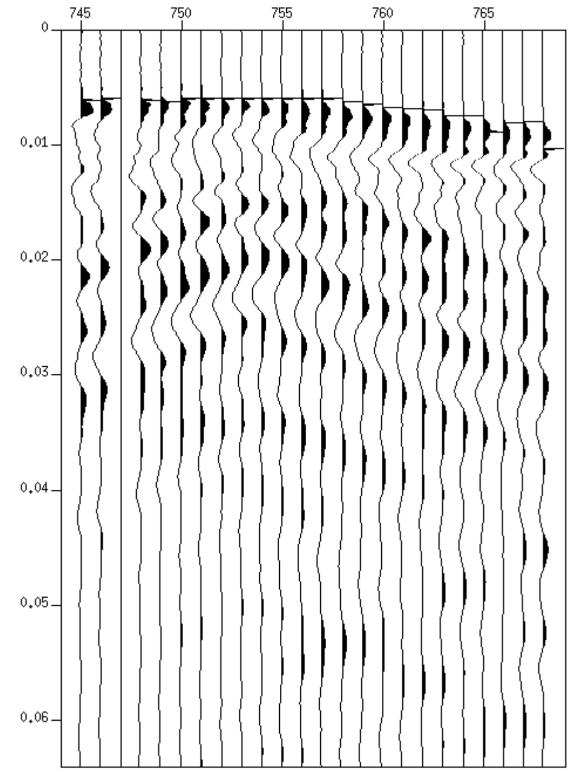
OOTPs	test 1		test 2	
	#	time	#	time
first	745	start : 0.006 end : 0.009	1	start : 0.03464 end : 0.04464
second	755	start : 0.006 end : 0.009	22	start : 0.004 end : 0.014
third	768	start : 0.008 end : 0.011	56	start : 0.0367 end : 0.0467
searching length	0.003		0.03	

sequence numbers selected by users, the start and end are the beginning and end of the training windows, and the start time of searching windows is the same as that of training windows and the end time of searching windows is calculated as start time plus searching length. For example, the start time of a searching window at the first trace of test 2 is 0.03464 and the end time of the searching window is 0.06464 (0.03464 + 0.03). With three searching windows at three training traces, the NFRS interpolates the searching windows at each trace.

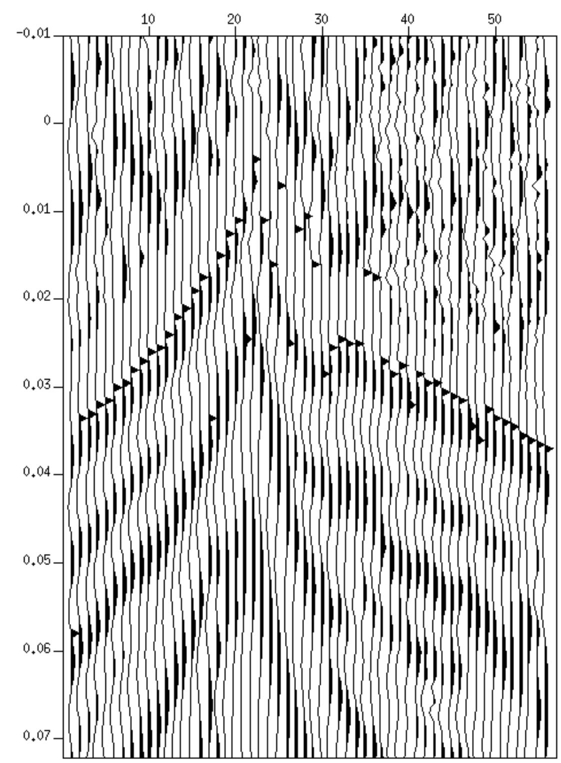
We select three OOTPs, which are located at first, middle, and the last traces in the records by giving the start and end of each training windows. The four ZOTPs at each training trace are automatically selected by the NFRS (the shifting array value is one). All training patterns are first normalized and then transformed to wavelet coefficients by WFEM to extract features. After training, the NFRS decides a first arrival pattern by observing the maximum output value in a searching window.

The ESLM finished about 200 $\sim$ 300 generations for test 1 and about 1000 $\sim$ 3000 generations for test 2. The ESLM took few minutes in training for test 1 and about ten minutes for test 2 on Pentium III PC with Linux OS. Figure 5 (a) and (b) show the recognition results for test 1 and test 2, respectively. As shown in the results, the recognition ability of NFRS is somewhat remarkable. However, the NFRS sometimes finds incorrect first arrival events. This is caused by two factors. First, the shapes of training waves are somewhat different each other. Second, in some traces neighbor waves of the first arrival waves are more similar to the training waves. Although the NFRS almost finds correct first arrival events in the traces that contain training waves, it fails to find correct events in a few cases. We think that this may be caused by incomplete learning

†The max and min indicate the maximum and minimum values of each input and output, respectively.



(a)



(b)

Fig. 5 Recognition results for (a) test 1 and (b) test 2 as well as the above two factors. In test 3, we used a user friend GUI program developed by our team. With this program, users can set or show the training windows, searching windows and can examine and edit the picking results. The parameters of NFRS for test 3 are the same as those for test 1 and 2 (see Table 1) except for the number of training data and rules. In test 3, the number of OOTPs are not restricted to 3 and training traces are located at any traces. Users can set any number of OOTPs in any traces. The number of OOTPs in test 3 is 5 and the ZOTPs per one OOTP is 2 (the shifting array value is one). Therefore, the number of training patterns is 15 $(5 + 2 * 5)$. In test 3, the number of rules are decided by Algorithm 2. The number of rules obtained by the algorithm is 15 with the value of $m^d = 0.5$. That is, the number of rules is the same as that of training patterns. We observed that the m^d increases, the number of rules also increases. It is natural by intuition. In case that the number of training patterns is large, the algorithm will be useful to reduce the hidden nodes of NFDM.

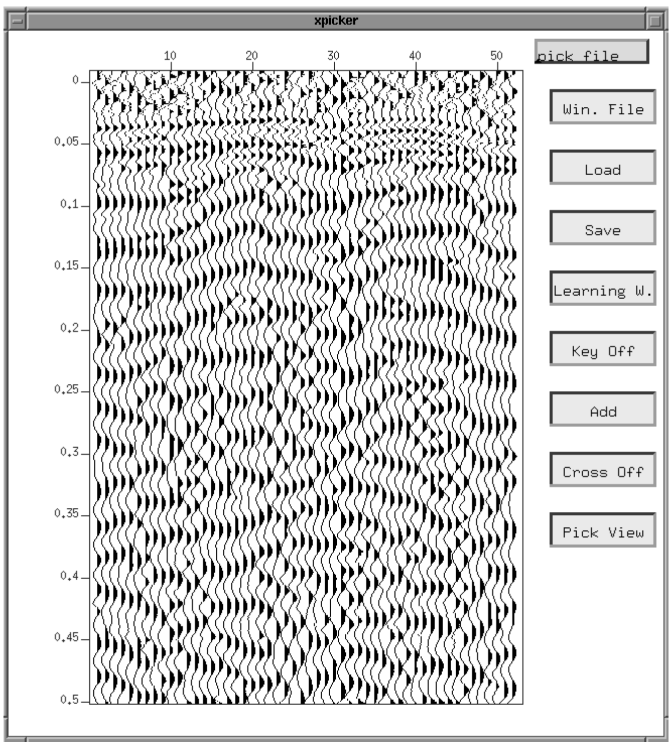


Fig. 6 Seismic record for test 3

Figure 6 shows the seismic record for test 3. Figure 7 shows the training windows and searching windows we selected. In Figure 7, the shaded area indicates the searching windows and four rectangular areas represent training windows. Unlike the case in test 1 and 2, the length of searching windows was given large compared to that of training windows. The picking results is shown in Figure 8. Although the NFRS failed to

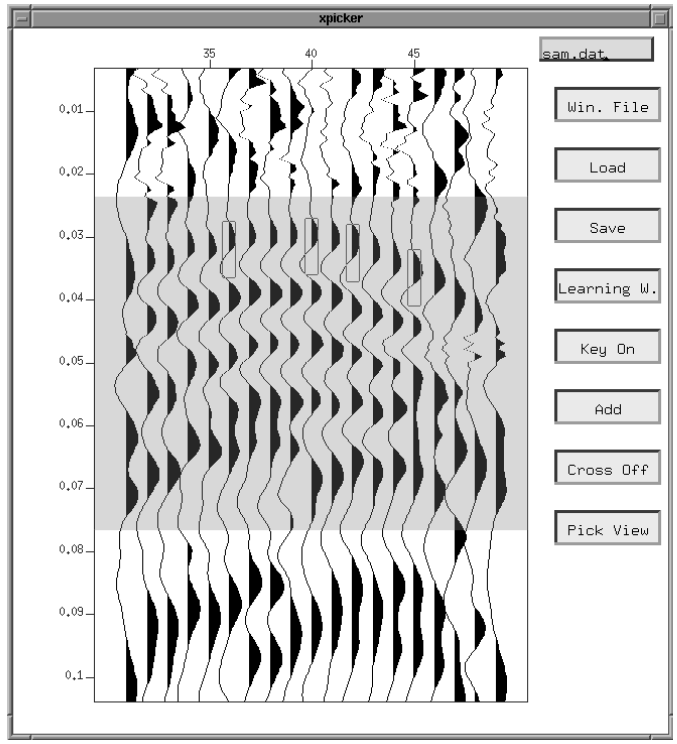


Fig. 7 Training windows and searching windows for test 3

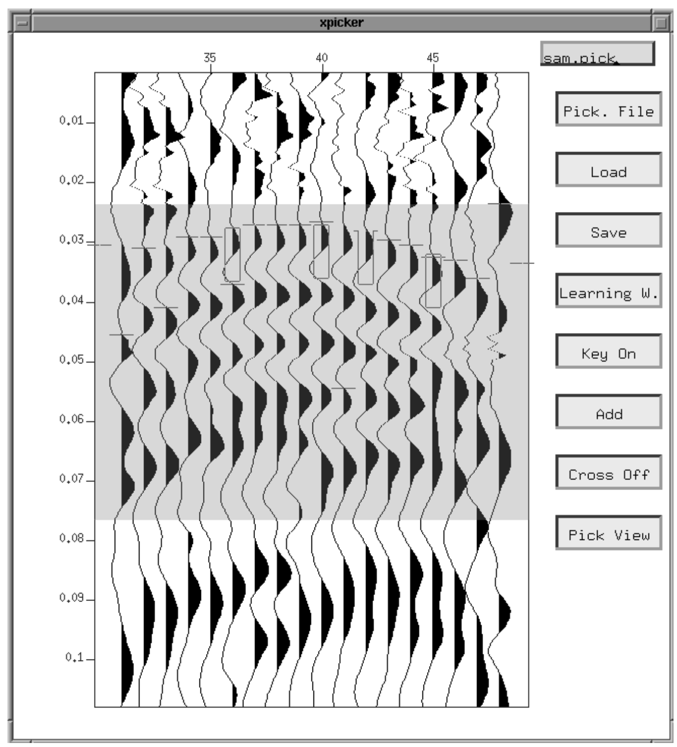


Fig. 8 Recognition result of test 3

pick in traces 31, 33, 36, and 41, the picking results are relatively good. Note that the picked wave patterns in the failed traces are very similar to the training wave patterns. Moreover, the other patterns on the failed traces are somewhat different from the training wave patterns. This problem may be overcame by a post processing algorithm which utilizes the training windows and the picked values of neighbor traces.

5. Conclusion

In this paper, we proposed a neuro-fuzzy system for recognizing specified wave patterns. We used wavelet coefficients of data as features and evolution strategy as learning algorithm of the system. This learning method can avoid for the system to get stuck at a local optimum that is inherent in conventional back-propagation algorithms. Moreover, the learning speed of this method is considerably fast. We applied our system to picking the first arrival events in seismic data. The picking process has been well known as a very difficult and time consuming process in geophysics. Experimental results with three seismic records showed that the performance of our system was better than those of previous ones. As further works, we will devise a post processing algorithm for improving the performance.

Acknowledgement

This work was conducted as a part of the National Research Lab project titled "Seismic Imaging Technology R&D and Applications". The authors thank the Korea Institute of Science and Technology Evaluation and Planning for the financial support of this project.

References

[1] C.-T. Hsieh, M.-C. SU, and C.-H. HSU, "Continuous Speech Segmentation Based on A Self-Learning Neuro-fuzzy system," *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, vol. E79-A, pp. 1180–1187, Aug. 1996.

[2] N. K. Kasabov, R. Kozma, and M. J. Watts, "Phoneme-Based Speech Recognition via Fuzzy Neural Networks Modelling and Learning," *Information Sciences*, vol. 110, pp. 61–79, Sept. 1998.

[3] N. Kasabov and R. Kozma, "Hybrid Intelligent Adaptive Systems: A Framework and a Case Study on Speech Recognition," *International Journal of Intelligent Systems*, vol. 13, pp. 455–466, June 1998.

[4] Y. Yan, "Understanding Speech Recognition Using Correlation-Generated Neural Network Targets," *IEEE Transactions on Speech and Audio Processing*, vol. 7, pp. 350–352, May 1999.

[5] C. I. Christodoulou and C. S. Pattichis, "Unsupervided Pattern Recognition for the Classification of EMG Signals," *IEEE Transactions on Biomedical Engineering*, vol. 46, pp. 169–178, Feb. 1999.

[6] M. Kundu, M. Nasipuri, and D. K. Basu, "A Knowledge-Based Approach to ECG Interpretation Using Fuzzy Logic," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 28, pp. 237–243, Apr. 1998.

[7] M. McCormack, D. Zaucha, and D. Dushek, "First break refraction event picking and seismic data trace editing using neural networks," *Geophysics*, vol. 58, no. 1, pp. 67–78, 1993.

[8] J. Veezhinathan, D. Wagner, and J. Ehlers, "First break picking using a neural network," *Expert Systems in Exploration, Soc. Exploration Geophys.*, pp. 179–202, 1991.

[9] C.-K. P. Chu and J. M. Mendel, "First break refraction event picking using fuzzy logic systems," *IEEE Transactions on Fuzzy Systems*, vol. 2, pp. 255–266, Nov. 1994.

[10] M. Figueiredo and F. Gomide, "Design of Fuzzy Systems Using Neurofuzzy Networks," *IEEE Trans. on Neural Networks*, vol. 10, pp. 815–827, July 1999.

[11] Y. Y. Tang, L. H. Yang, J. Liu, and H. Ma, *Wavelet Theory and Its Application to Pattern Recognition*. World Scientific, 2000.

[12] L. Prasad and S. S. Iyengar, *Wavelet Analysis with Application to Image Processing*. CRC Press, 1997.

[13] M. Vetterli and J. Kovacevic, *Wavelets and Subband Coding*. Prentice Hall, 1995.

[14] S. P. Chael, A. M. Sayeed, and R. G. Baraniuk, "Improved Wavelet Denosing via Empirical Wiener Filtering," *Proc. SPIE*, vol. 3169, pp. 389–399, Oct. 1997.

[15] A. Graps, "An Introduction to Wavelets," *IEEE Computational Science and Engineeing*, vol. 2, no. 2, pp. 50–61, 1995. reference site http://www.amara.com/IEEEwave/IEEEwavelet.html.

[16] S. H. Jung, K. H. Cho, T. G. Kim, and K. H. Park, "Defuzzification Method for Multishaped Output Fuzzy Sets," *Electronics Letters*, vol. 30, pp. 740–742, Apr. 1994.

Sung Hoon Jung He is an assistant professor in the School of Information and Computer Engineering, Hansung University. He received his B.S degree from Hanyang University, Korea, in 1988 and M.S. and Ph.D. degrees from KAIST, in 1991 and 1995, respectively. His research interests are in the field of pattern recognition, intelligent control, and in particular of applications of neural networks, fuzzy control, and evolutionary computation. Dr. Jung is a member of KFIS, KIPS, and IEEK.

Doo Sung Lee He is an associate professor in the School of Information and Computer Engineering, Hansung University. He received his Ph.D. degree from CSM, USA, in 1979. His research interests are image processing, seismic imaging, and remote sensing. Dr. Lee is a member of SEG, KSEG, SEGJ, KSRS, and KSGE.