A Model for MMI-Attenuation Based on Artificial Neural Networks and Genetic Algorithms

G-Akis Tselentis

Seismological Laboratory, University of Patras

RIO 261 10, Greece

tselenti@upatras.gr

Liviu Vladutu

Dublin City University, Republic of Ireland

lvladutu@computing.dcu.ie

Laurentiu Danciu

Seismological Laboratory, University of Patras

RIO 261 10, Greece

Abstract

Complex application domains involve difficult pattern classification problems. The present paper introduces a model for the MMI-attenuation relation and its dependence upon engineering ground motion parameters, which is based on artificial neural networks (ANN) and Genetic Algorithms (GA). The ultimate goal of this investigation is to evaluate the applicability of ground-motion attenuation relations, developed for a host region, in a target region, by training an ANN, using the seismic patterns of the host region. The ANN learning is based on supervised learning using existing data from past earthquakes. The combination of these two learning procedures (GA and ANN) introduces a new method in pattern recognition with seismological applications. The performance of this new GA-ANN regression method has been evaluated on a Greek seismological database with satisfactory results.

Introduction

A major problem encountered in engineering seismology is to assess the damage potential of an earthquake, expressed by the distribution of seismic intensities from well known recorded ground motion parameters (e.g. Tselentis and Danciu, 2008; Danciu and Tselentis, 2007). However, a physically based ground-motion measure is needed for engineering purposes. With the advent of instrumental seismology, the relatioship between the intensity and the ground-motion parameters has become a topic of increasing interest.

The objective in the present investigation, is to uncover hidden complex and often fuzzy relations between the engineering ground motion parameters and macroseimic intensity, in the form of input/output dependencies. The emergence of neural network technology (Haykin, 1999; Bishop, 1996), offers valuable insight to confront with these complicated problems. In this context, neural networks can be viewed as advanced mathematical models for discovering complex correlations between variables of physical processes from a set of perturbed observations.

Engineering seismological parameters

The quantification of ground motion requires a good understanding of the ground motion parameters that characterize the severity and the damage potential of the earthquake and the seismological, geological, and topographic factors that affect them. Parameters related solely to the amplitude of the ground motion such as the peak ground acceleration (PGA) and the peak ground velocity (PGV) are often poor indicators of structural damage.

Because structure or equipment damage is measured by its inelastic deformation, the earthquake-damage potential depends on the time duration of motion, the energy absorption capacity of the structure or equipment, the number of strain cycles, and the energy content of the earthquake. Therefore, for engineering purposes, parameters that incorporate in their definition the previously mentioned characteristics are more reliable predictors of the earthquake's damage potential. The most frequently used ones are Arias intensity (I_a), acceleration response spectrum (S_a), and cumulative absolute velocity (CAV). The Cumulative Absolute Velocity (CAV) is defined as

$$CAV = \int_0^t |a(t)| dt \qquad (1)$$

where t is the total duration of the record, and $a(t)$ is the acceleration time history.

CAV defines a simple energy-related bound of the response velocity of a SDOF (Single-Degree-of-Freedom) system subjected to seismic excitation, (Ahmadi, 1986). According to its definition CAV accounts for the contribution of both the amplitude and the duration of the motion.

The Arias Intensity (I_a) introduced by Arias, (Arias, 1970), is a measure of ground motion intensity corresponding to the total energy stored at the end of a family of linear undamped or moderately dumped oscillators with varying frequency and can be expressed as,

$$I = \int_0^\infty E d\omega \qquad (2)$$

where E is the energy dissipated per unit weight of a structure and ω is the frequency.

Data set

The strong-motion records used for the present investigation have been provided by the European Strong Motion Database (Ambraseys *et al.*, 2004) and are presented in Figure 1. More details about these data can be found in (Danciu and Tselentis, 2007). The macroseismic information is available partly from the digital database of the web site for European strong motion data and partly estimated separately by us from the macroseismic data provided by the Geodynamic Institute of the National Observatory of Athens (Kalogeras et al., 2004). The general criterion was to allocate at each station the nearest MMI values within an uncertainty of one unit to every station. If more than one MMI value was observed near the station location at equal distance, the average of the values was used (Tselentis and Danciu, 2008).

Artificial Neural Network

There are several well-known categories of ANN like the feed-forward neural networks, which are including Radial-Basis Function (RBF) networks, or multi-layer perceptrons. For example, RBF networks exploit the Tikhonov's regularization (Poggio and Girosi, 1990a; Poggio and Girosi, 1990b; Girosi, 1998), while Multi-Layer Perceptrons (MLP) are well-known as universal approximators (Hornik et al., 1989), and, they have a simple structure easier to interpret in comparison with other neural networks. Figure 2 shows a simplified view of a feed-forward ANN.

It consists of a network of simple processing elements (artificial neurons) which are organized in several layers: an input one (which has the number of neurons linked to the dimensionality of the input), one or several hidden layers and an output layer. The hidden layer provides a representation for the inputs. After the inputs are weighted they are summed up and passed through a function *f*. A representation of the NN used in the present investigation is shown in Fig. 3, with the 9 inputs being the

engineering seismological ground motion parameters. The architecture containing multiple hidden layers is more powerful than single-layer networks. The units (neurons) have their activation function characterized by a nonlinear function (like the sigmoid function in Fig. 4). This function maps the output of the function to its input and this is expressed as

$$Y = f(b + \sum_{i=1}^{R} W_i * p_i), \quad (3)$$

where b is the bias, and W_i (i = 1,…,R) is a weight corresponding to input p_i.

We have considered this problem of automatic MMI assessment based on ground motion parameters as part of a larger category of problems encountered in Pattern Recognition, (Poggio and Girosi, 1990a),(Duda and Hart, 2001)]. In the present investigation we consider the following four phases

- feature extraction
- classification
- pre-processing and optimization
- regression

Feature extraction

During this phase, we combined the enhanced selection offered by GA with the performance of an ANN as a classifier. At first, we used all nine possible input parameters that characterize an earthquake ground motion at a site corresponding to an MMI value. These measures are M, log(R), PGV, log(PGV), S_a, PGA, log(PGA), I_a and CAV (Fig.3).

During this procedure we use 9-bits strings quantification with binary field-values as

M logR S_a PGV logPGV PGA logPGA I_a CAV

For example, the 3 parameters string [logR, I_a, CAV] is represented as the string: **010000011**. Like in the natural selection (the genes of animals or plants) the mutations and crossovers are allowed (this time, between strings). A genetic algorithm was used to generate populations of strings out of the 512 possible combinations (from **000000000** to **111111111**). The total number of available strings (the equivalent of 'chromosomes') at a certain time, (i.e. after the Max_No_Generations), is called 'the genome', and was evaluated by an ANN implemented as a k-NN (k-Nearest Neighbor). This was achieved by comparing the corresponding MMI (the outputs) with the selected inputs (out of the nine possible ones) represented by the strings generated by the genetic algorithm. In our implementation we have allowed a population size of 20, and in this case, the population size is the maximum number of chromosomes (strings) allowed in a generation.

Classification

The second phase, dealing with the classification, is implemented by a k-NN type neural network. The k-nearest neighbors algorithm (k-NN), is a method for classifying objects based on closest training examples in the feature space. k-NN is a type of instance-based learning, or *lazy learning* where the function is only approximated locally and all computation is deferred until the classification.

In the machine-learning community, the *instance-based learning* -IBL (Aha et al., 1991) also known as *memory-base learning* is a family of learning algorithms that instead of performing explicit generalization, compare new instances with instances which have been seen during the training. It is called instance-based because it constructs hypotheses directly from the training instances themselves. A direct consequence is that the complexity of the problem grows with the amount of the data available for training and testing.

From the data set of 310 data values some were left for testing, while most of them were considered for training. We have used this approach (GA-ANN with IBL) in order to avoid data changes produced by normalization techniques. The Euclidean metric is used to assess distances between training/ testing epochs.

In our approach, we used one of the simplest examples of IBL, the k-nearest neighbors classifier (k-NN), and it's Java implementation based on Weka-toolbox (Witten and Frank, 2005). k-NN is amongst the simplest machine-learning algorithms. An object is an instance out of the 310 data values and is given in the *features space*, which is found to be the 5 parameters string: [M, log(R),PGA, I_a , CAV]. It is classified by a majority vote of its neighbors with the object being assigned to the class most common amongst its *k* nearest neighbors.

Thus, the k-NN classifier takes the optimum combination of inputs (from the nine mentioned) and, therefore only 5 inputs [M, log(R), PGA, I_a and CAV] were selected. The k-NN will perform the classification only for those columns (the 1^{st}, 2^{nd},6^{th}, 8^{th} and 9^{th}) based on quadratic error criteria.

Pre-processing and optimization

In the previous two phases we considered all inputs as they were, but for the processing purposes, we convert all inputs into integers by multiplying them with powers of 10 (Härdle et al., 1995; Mierswa et al., 2006).

Candidate solutions to the optimization problem play the role of individuals in a population, and the fitness function determines the environment within which the solutions "live" (e.g. cost function). Genetic algorithms are a particular class of evolutionary algorithms -EA, (Blickle et al, 1995),(Fonseca et al,1995) that use techniques

inspired by evolutionary biology such as inheritance, mutation, selection, and cross-over (also called recombination).

In other words, a GA is quantifying the information (the parameters of the k-nearest neighbor classifier) in the form of strings (the chromosomes), and through the EA only the fittest chromosomes survive over the generations of the evolution. Therefore, an important parameter for the proposed, GA-ANN method in this investigation was the Max_No_Gener in which the GA algorithm was allowed to evolve so to achieve the optimum solutions.

In our case there are several parameters that have to be modified (fine-tuned) in order to achieve an optimum behavior of the classifier- the neural network (like the number of hidden layers and the number of neurons in each hidden layer).

After finding the optimum Max_No_Gener parameter the other parameters of the method were determined accordingly. These are: *k,* the optimum number of neighbors and the selection scheme or the size of the tournament. In doing this *k* was found by a trial-and-error procedure to have a value of 2. An advantage of the selection mechanism of a GA, (Tobias and Lothar, 1995) is its independence of the representation of the individual, as only the fitness values of the individuals are taken into account.

A fitness function is a particular type of objective function (i.e. the function considered for the optimization procedure) that prescribes the optimality of a solution, (the solutions are represented as chromosomes), in a genetic algorithm, so that the particular chromosome may be ranked amongst all the other chromosomes from the genome. Chromosomes which are found to be '*more optimal'* are allowed to breed (i.e. further binary combinations will be created by the GA on the 'skeleton' of these 'more optimal' ones) and mix their datasets by any of several available techniques, producing a new generation of chromosomes that will hopefully be better. The 'mix' in our

case can be represented by taking first 4 digits from one 'optimal' string and the last 5 digits from an other 'more optimal' one and in this way create a new chromosome (9-digits string) that can hopefully perform better in the k-NN classification. This simplifies the analysis of the selection methods and allows a comparison that can be used for all kinds of genetic algorithms.

One of the frequently used selection schemes is *"the tournament selection"*. In this scheme we run a "tournament" among a few individuals chosen at random from the population (i.e. from the genome) and select the one with the best fitness as the winner for crossover that is adjusted by varying the tournament size.

In the theory of genetic algorithms, the crossover is the genetic operator used to modify the programming of a chromosome or a group of chromosomes from one generation to the next. It is the analogous of the reproduction and biological crossover (in the nature) upon which the simplified theory of GA in computational intelligence was built.

First, a single crossover point on both parents organism's strings is selected (avoiding obviously the extreme points). All data beyond that point in either organism string are swapped between the two organism strings. The resulting organisms are the children (offspring) as is shown in Fig.5.

If the tournament size is larger, weak individuals (chromosomes for which the objective function, i.e. the error has a higher value) have a smaller chance to be selected for breeding, crossover and perpetuation in the next generations. The performance was quantified by RMS criteria and square error. A flow chart describing all the above operations is presented in figure 6.

In order to validate the performance of the above mentioned GA-ANN selection schemes, we selected a validation scheme based on the regression performance. Re-

sults for the selection of the 1st optimum parameter, from those described above, (Max_No_Gener) are presented in Table 1. Judging from the values depicted in Table 1 we have the following cases:

First, an "underdetermined solution", which results in values of Max_No_Gener in the range 25-70. This is rejected because GA need a minimum number of generations in order to obtain the optimum fitness among all available individuals. This optimum fitness is given (as input to the genetic algorithm) by the k-NN neural-network classifier. For example, let's suppose that we have to compare the following two situations:

In the case that 75 are the maximum-number of generations, we obtained as 'fittest' the string S_1, where S_1 is given by the GA+k-NN algorithm and corresponds to the combination [M, log(R), log(PGV), I_a, CAV]. This is represented as **11000011**. The fittest' chromosome is considered one for which the objective function has an extreme value (in our case, the objective function is the square-error of the k-NN type of ANN, and so, the objective function must have a minimum value).

If we consider the last generation, (the 80th in our case), with string S_2, as the fittest one, corresponding to the combination (M, log(R),PGA, I_a, CAV) and represented as 110010011. In this situation, S_2 is retained as having the best fitness, since the output of the k-NN classifier for S_1 is given by the squared-error of: 0.329 +/- 0.170 and for S_2, by the 0.327 +/- 0.164, i.e. both the error and it's standard deviation are higher in the case of S_1.

Second, a higher than optimum Max_No_Gener (90) which is rejected due to worse regression performance (higher RMS error). Finally, the solution with Max_No_Gener =94 was rejected since it practically took almost all input parameters (was a trivial solution).

Third, an optimum value of Max_No_Gener equal to 80 (minimum RMS, square and absolute errors) and was not under-trained like case (1).

In this case, the optimum selected input parameters are: M, logR, PGA, I_a and CAV. These are the parameters which will be used to express MMI throughout a regression process. Absolute error performed like the best harbinger (descriptor) out of the 3 types of errors was used for validation.

Regression

In the case that the existing data are not sufficient for the analysis (since we have to use part of the data for validation and test sets), it is common to use the cross-validation or rotation estimation method, (Kohavi, 1995).

This is a technique for assessing how the results of a statistical analysis will generalize to an independent data set. It is mainly used in cases where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice. Cross-validation involves partitioning a portion of the data into complementary subsets, performing the analysis on one subset (called the training set), and validating the analysis on the other subset (called the validation set or testing set). For relatively larger datasets a usually higher cross-validation is used (25 in our case).

Next, the selected optimum input parameters [M, logR, PGA, I_a and CAV] are considered as X multivariate inputs, and the response variable Y as the MMI-attenuation. In order to calculate the coefficients that are linking Y to the 5-dimensional X variable we used linear regression described by the equation (Härdle et al., 1995).

$$Y = b_0 + \sum_{i=1}^{5} b_i * X_i \qquad (4)$$

where b_0 is the intercept and b_1 ,…, b_5 are the coefficients for the ground parameters in the MMI equation.

The obtained b values are presented in Table 2, and the results of the ANOVA statistical test are shown in Table 3. Thus, the relation which describes MMI as a function of [M, logR, PGA, Ia CAV] is

$$MMI=8.824+0.417M-7.960\log R+0.380PGA+1.105I_a-0.551CAV$$

Fig.7 shows the time series corresponding to the original data and the results of the regression analysis. For the GA-ANN selection scheme we used a Java based implementation, built around the Weka (the IBk lazy learner) data-mining system, (Witten and Frank, 2005).

5. Conclusions

In this research, we presented a model of MMI attenuation relation and its dependence upon engineering ground motion parameters, based on ANN and GA. The performance of this new regression approach has been tested on a Greek strong motion data base with satisfactory results.

We note that not all the features selected in the GA-ANN approach have the same influence on the MMI-attenuation. An approach based on Evolutionary Algorithms can be useful in weighting the importance of those features. Also a new type of neural networks (Evolutionary NN) can be used to replace the classical k-NN that we've used in the current paper. If we implement also and an expert system, making the analysis of the results of feature selection presented in Table 1, we end up with a real-time signal processing system, to be used in seismology and not only.

References

Ambraseys, N., P. Smit, J. Douglas, B. Margaris, R. Sigbjornsson, S. Olafsson, P. Suhadolc, and G. Costa (2004). Internet-site for European strong-motion data, Boll. Geofis. Teor. Appl. 45, no. 3, 113–129.

Aha, D. W. (1991).Instance-based Learning, *Machine Learning*, Kluwer Academic Publishers, 6, 36-77.

Arias, A. (1970). *A measure of earthquake intensity*, Cambridge, MA.

C. M. Bishop C.M. (1996). *Neural Networks for Pattern Recognition,* Oxford University Press.

Danciu, L. and Tselentis, G-A. (2007). Engineering ground motion parameters attenuation relationships for Greece, *Bull. Seismol. Soc. Am.* 97,No1B, 162-183.

Fujinawa, Y., Matsumoto, T. , Takahashi, K. (2005). Method for estimating origin time, hypocentral distance and scale based on electric field observation, and apparatus for prediction", *Patent No. US 6,885,945 B2*.

Girosi, F. (1998). An Equivalence Between Sparse Approximation and Support Vector Machines, *Neural Computation,* 10:6, 1455-1480.

Härdle, W., Klinke, S. and Turlach, B. (1995). XploRe - an Interactive Statistical Computing, Environment, Springer, Heidelberg.

Härdle, W. (1994). *Applied Nonparametric Regression*, Humboldt-Universität zu Berlin Wirtschaftswissenschaftliche Fakultät Institut für Statistik und Ökonometrie.

Haykin S. (1999). *Neural Networks,* MacMillan College Publishing Company, Second Edition.

Hornik, K., Stinchcombe, M., and White, H (1989). Multilayer feedforward networks are universal approximators, *Neural Networks*, 2:359–366.

Kalogeras, I. S., G. Marketos, and Y. Theodoridis (2004). A tool for collecting, querying, and mining macroseismic data, Bull. Geol. Soc. Greece XXXVI.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence* 2 (12): 1137–1143, (Morgan Kaufmann, San Mateo), http://citeseer.ist.psu.edu/kohavi95study.html.

Mierswa, I., Wurst, M., Klinkenberg, R., Scholz M. and Euler T. (2006). YALE: Rapid prototyping for Complex Data Mining Tasks, in *Proceedings of the 12th ACM SIGKDD International Conference on Data Mining and Knowledge Discovery, (KDD-06)*.

Poggio T. and Girosi. F. (1990a). Regularization algorithms for learning that are equivalent to multilayer perceptrons, *Science 247*, 978-982.

Poggio, T., and Girosi, F. (1990b). Networks for approximation and learning, *Proceedings of the IEEE,* vol. 78, 1481-1497.

Tselentis G.-A. and Danciu, L. (2008). Empirical relationships between MMI and engineering ground motion parameters in Greece. *Bull. Seim. Soc. Am.* Vol.98, N4.

Witten Ian H. and Frank, E. (2005). *Data Mining- Practical Machine Learning Tools and Techniques*, Elsevier.

Captions of tables

Table 1: The parameters retained by the GA-ANN selection algorithm and the corresponding regression performance (columns 2 and 3).

Table 2: The parameters for the linear regression obtained by the program XploRe for the input data selected by the GA-ANN proposed methodology.

Table 3: The statistical parameters obtained for the linear regression using the ANOVA test.

Captions of figures

Figure 1: Epicentral distribution of the earthquakes used in the present analysis.

Figure 2: General topology of a feed-forward ANN with one hidden layer.

Figure 3: Topology of the feed-forward ANN (of k-NN type) used in the present investigation.

Figure 4: The graph shows two classical nonlinear activation functions.

Figure.5: One point crossover.

Figure 6: Flowchart showing the sequences of the feature-selection and classification.

Figure 7: The output of the regression algorithm (in red) and the original MMI data (in blue) for the 310 data points considered.

Max_No_Gener.	Sq error	RMS error	**root_relative_ squared_error**	The retained params
75	0.329 +/- 0.170	0.553 +/- 0.152	0.608+/- 0.178	M,logR,logPGV,I_a,CAV
80	**0.327 +/- 0.164**	0.546 +/- 0.169	**0.607+/-0.195**	M,logR,PGA,I_a,CAV
50	0.321 +/- 0.165	0.545 +/- 0.155	0.613+/- 0.224	M,logR,S_a,I_a,CAV
90	0.339 +/- 0.231	0.548 +/- 0.196	0.618+/-0.189	logR,I_a,CAV
94	0.324 +/- 0.194	0.541 +/- 0.176	0.608+/-0.194	M,logR,S_a,logPGV,PGA,I_a,CAV

Table 1

PARAMETERS	Beta	SE	StandB	t-test	P-value
b[0,]	8.8236	4.0481	0.0000	1.439	0.1513
b[1,]	0.4173	0.9553	0.2733	0.437	0.6625
b[2,]	-7.9601	12.2908	-0.4084	-0.648	0.5177
b[3,]	0.3801	0.4533	0.1499	0.839	0.4024
b[4,]	1.1046	0.5022	0.7718	2.200	0.0286
b[5,]	-0.5508	0.5965	-0.2046	-0.923	0.3566

Table 2

A N O V A	SS	df	MSS	F-test	P-value
Regression	155.893	5	31.179	67.556	0.0000
Residuals	140.304	304	0.462		
Total Variation	296.197	309	0.959		

Multiple R = 0.72548

R^2 = 0.52632

Adjusted R^2 = 0.51853

Standard Error = 0.67936

Table 3

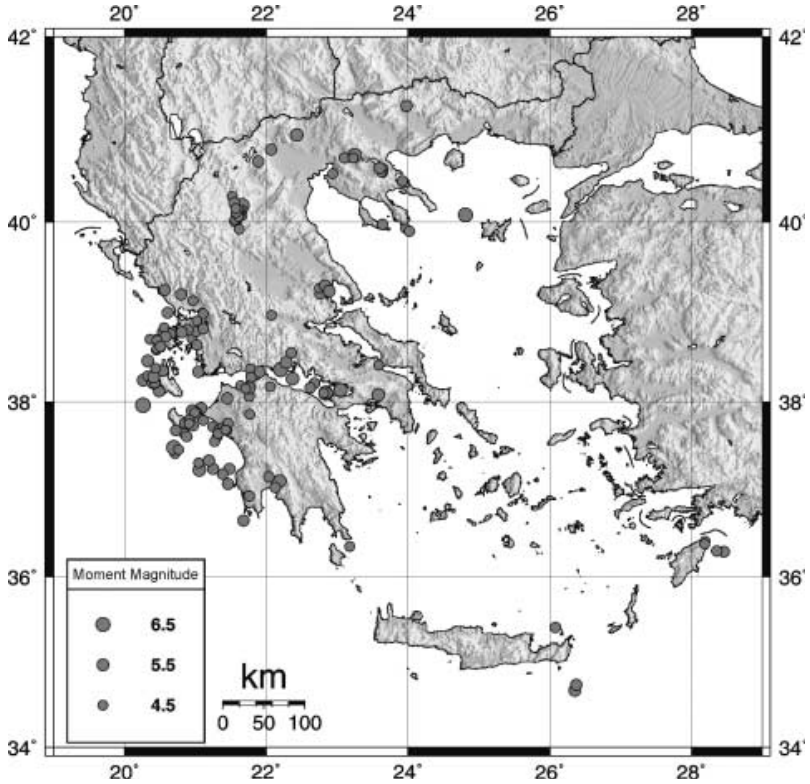
20°
22°
24°
26°
28°
42°
40°
38°
36°
34°
Moment Magnitude
6.5
5.5
4.5
km
0 50 100

Fig.1

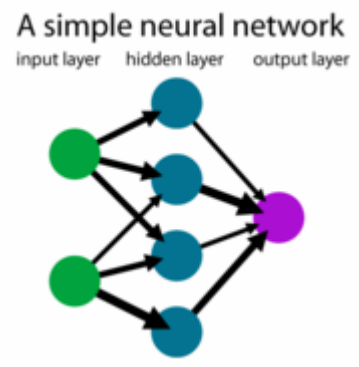
A simple neural network
input layer
hidden layer
output layer

Fig.2

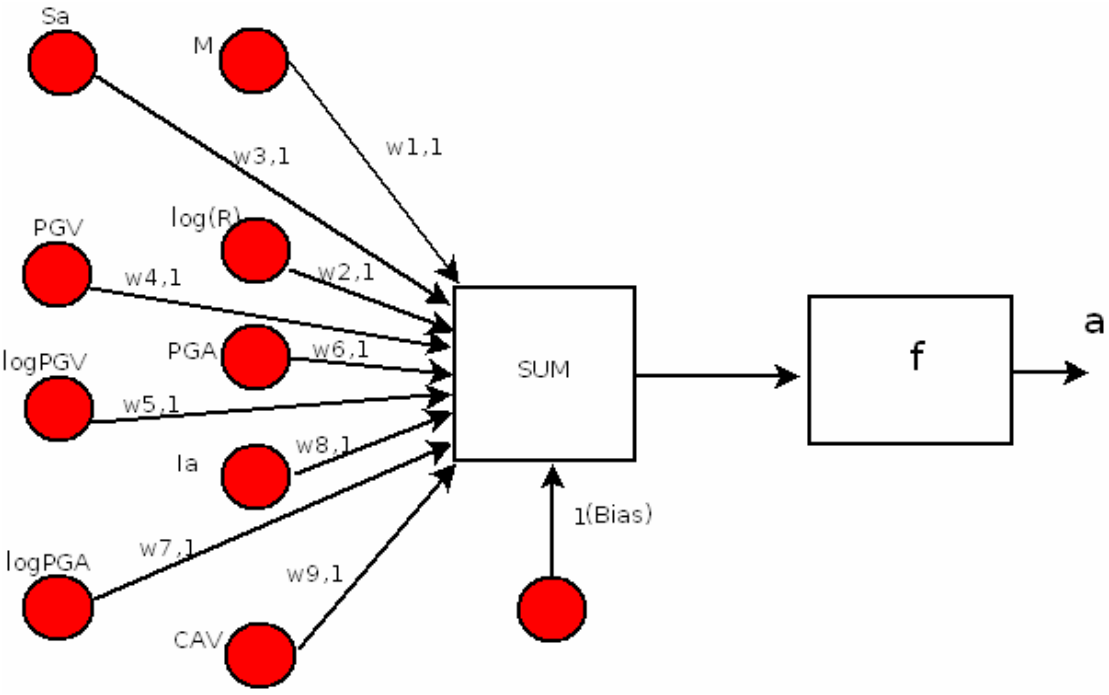

Fig.3

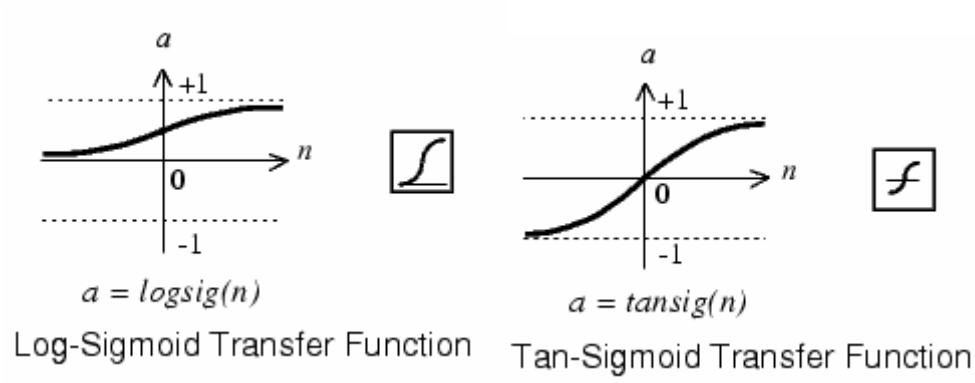

Fig.4

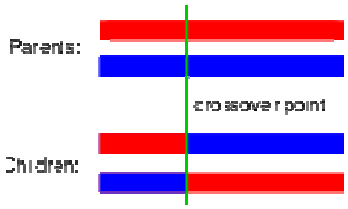

Fig.5

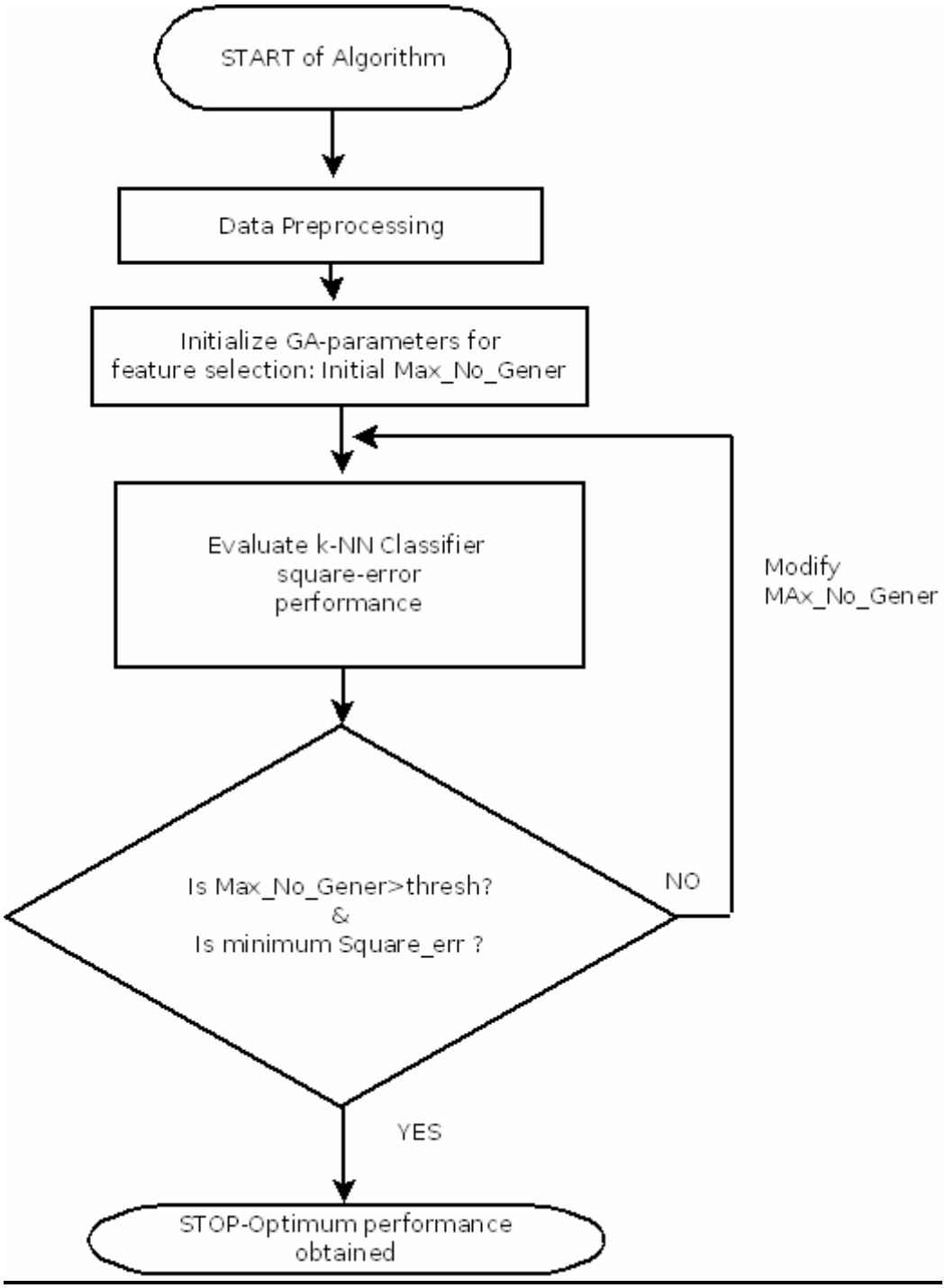
START of Algorithm
Data Preprocessing
Initialize GA-parameters for feature selection: Initial Max_No_Gener
Evaluate k-NN Classifier square-error performance
Modify MAx_No_Gener
Is Max_No_Gener>thresh? & Is minimum Square_err ?
NO
YES
STOP-Optimum performance obtained

Fig.6

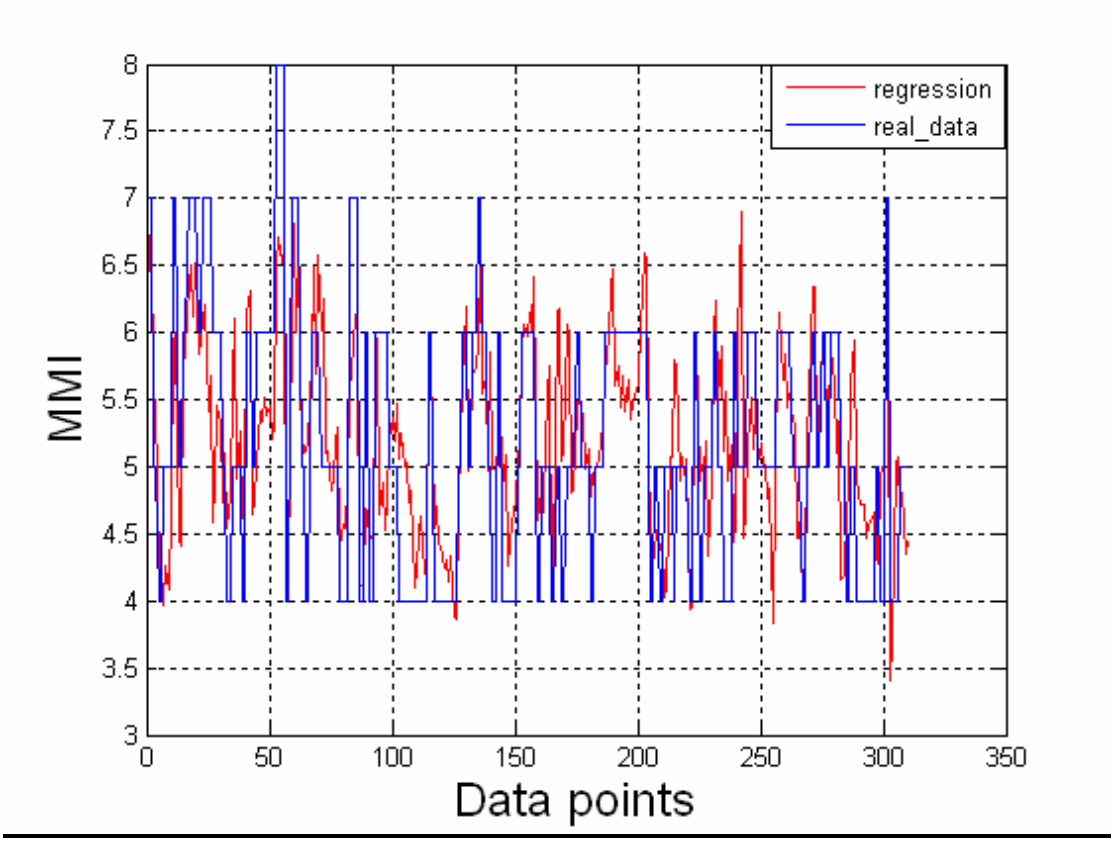
regression
real_data
MMI
Data points
8
7.5
7
6.5
6
5.5
5
4.5
4
3.5
3
0
50
100
150
200
250
300
350

Fig. 7