Kecman V., ***System Identification Using Modular Neural Network With Improved Learning,*** The International Workshop on Neural Networks for Identification, Control, Robotics, and Signal/Image Processing, NICROSP' 96, Venice, pp. 40-48, IEEE Computer Society Press 1996

System Identification Using Modular Neural Network With Improved Learning

Vojislav Kecman
The University of Auckland
Private Bag 92019, Auckland, New Zealand
Email: v.kecman@auckland.ac.nz

Abstract

This paper addresses the problem of the identification of nonlinear dynamic systems using modularly structured neural network with the new learning algorithm for the learning of both gating and expert networks weights. Here we start with the standard learning procedure for such networks in the sense that the problem of learning is formulated and treated as a mixture estimation problem in which the log-likelihood function should be maximised. But, as opposite to the established methods for modular networks we combine the gradient type of learning for gating weights with a least-squares algorithm for the learning of expert networks weights, and the experts are simple one-layer nets with a single linear output unit. The very result of such an approach is a simple structured modular network with improved learning and it seems with good capabilities for the identification of general nonlinear dynamic systems. The identifications of a discrete-time nonlinear system corrupted with noise and of a real world system are presented. Later process represents the identification of the positioning of a car engine throttle valve from real data set (3000 samples of measured noisy data).

Introduction

In past few years artificial neural networks (NN) have been applied to a number of problems and with an exponential growth. This is particularly true for the fields of control and identification of nonlinear systems [6]. Such a development is primarily due to the fact that NN are general approximators in the sense that they are able to model any nonlinear high dimensional mapping providing enough representation power (meaning enough neurons). For the application of NN in the field of identification of nonlinear dynamic systems many questions are still open (selection of the input signals, sampling time, type of NN, learning method, to name just a few) but many results presented in literature are promising.

Here, the results of identifying two different nonlinear dynamic systems: discrete-time nonlinear system corrupted with noise and a real world system identification will be presented. Later process represents the identification of the positioning of a car engine throttle valve from the real data set. The relationship between the voltage of servo-motor (input variable) and the throttle valve position angle (output variable) should be identified using 3000 samples of recorded noisy data. Classical identification for such system has failed due to the fact that the process could only has been operated in a closed loop and because it has a heavily nonlinear friction characteristics [9], [7].
For such a task we use modularly structured NN consisting of gating and expert networks [1-5]. Modular neural networks seem to be a new and promising type of networks which combine good properties of both 'local' type of neural networks resulting in approximations having low bias and high variance and 'global' type of NN which approximate the data with higher bias and lower variance, in general. Similar to the start of the whole NN field a quite big deal of effort is devoted to the developing of proper (meaning robust and fast) learning algorithm for the (hierarchically) structured modular network, too [1-5]. Here the modular NN will be trained with the new learning algorithm for training of both gating and expert networks weights. We kept to the standard approach for learning in such networks in the sense that the problem of learning was formulated and treated as a mixture estimation problem in which the log-likelihood function should be maximised. But, this is just a framework of the learning here, because as opposite to the other methods we combine the gradient type of learning for gating weights with a least-squares algorithm for the learning of expert networks weights (where the experts are simple one-layer nets with a single linear output unit). The very result of such an approach is modular network with improved learning (basically the learning is almost always an one epoch training) and it seems with good capabilities for the identification of general nonlinear dynamic systems.

Modular network and learning by mixed optimisation procedure

The basic idea in application of the modularly structured NN is to have a network which is able to divide a complex mapping into simpler mappings, which are to be modelled by expert networks, and then to combine the outputs of expert networks in accordance to their 'responsibilities' for certain domains of input variables **x**. The division of input space is to be solved by gating network(s) by giving the domain of 'responsibility' to the experts. The simplest modular NN structure having one level of networks is represented in Fig. 1 and such a NN will be used for the identification tasks here. (Interestingly, such a 'simple' structure will be good enough to model quite complex nonlinear dynamic mapping). Note that in general the network could be hierarchically organised having deliberate number of levels or depth. (For more about network hierarchy see [5]). Modular NN in Fig. 1 comprises of expert networks and one gating module. There are no particular requirements on the type of artificial NN used in expert modules and in general they do not have to be of the same type or complexity in each expert. The approach presented here and the learning law do not depend on particular type of expert network providing that the activation functions (AF) in output layers are linear ones what is usually more than reasonable for regression type of tasks. Note that

the splitting of input domain into sub-regions will enable the expert networks to be of relatively simpler structure than in the case when they would be required to model the input to output mapping over the whole, originally compact, input space.
The basic model of such a network is given below (see [1-5] for detailed and in depth description, motivation and basic explanations of modular NN).

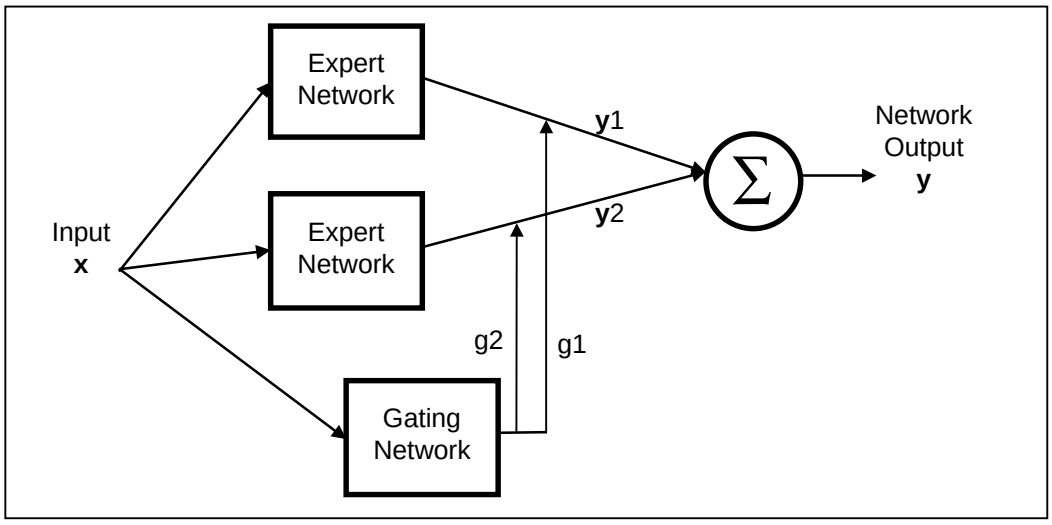

Fig. 1: Basic structure of one level modular neural network

Input vector **x** (real data vector augmented with the 'bias' term +1) is same for experts and gating network. Here, the AF of experts modules are linear units and of the gating network one is so-called softmax function given by (2)

$$\mathbf{y}_k = \mathbf{w}_k^T \mathbf{x} \tag{1}$$

$$g_k = \frac{\exp(net_k)}{\sum_{j=1}^{K} \exp(net_j)} \tag{2}$$

Notice that the number of outputs from the gating network equals to the number of experts *K (k=1,...,K)* and input *net* to gating network is given as,

$$net_k = \mathbf{v}_k^T \mathbf{x} \tag{3}$$

(The vectors **x, y, w** and **v** are of proper dimensions and **g** is *K*-dimensional column vector. Notice that in Fig 1, if the output vector **y** is one-dimensional, meaning scalar, the number of the weights that should be learned is (dim(**x**), 4) and *K=2*. In general case, when **y** is of higher dimension the expert network output layer weight matrix **w** in (1) is a matrix (dim(**x**), dim(**y**)) and not a column vector. Important property of **g** is that a sum of its elements equals to 1. This is necessary because the outputs g_k are interpreted as *conditional* (on **x**) *a priori* probabilities. The outputs $\mathbf{y}_k$ from expert

modules are interpreted as *conditional* (on both **x** and *k*) *mean vectors* and the desired (target) vector **d** is treated as a linear combination of *K* different multivariate Gaussian distributions.

$$p(\mathbf{d}|\mathbf{x}) = \sum_{k=1}^{K} g_k p_k \tag{4}$$

$$p_k = p(\mathbf{d}|\mathbf{x},k) = \frac{\exp(-\frac{1}{2}(\mathbf{d}-\mathbf{y}_k)^T \Sigma_k^{-1}(\mathbf{d}-\mathbf{y}_k))}{(2\pi)^{\dim(y)/2}\left|\Sigma_k\right|^{-1/2}} \tag{5}$$

In this *mixture model* p_k represents the multivariate Gaussian distribution of the desired (target) vector **d**, given the input vector **x** and that the *k-th* expert is chosen or responsible. For the given probability model in (4) the expected value of the output is given as follows,

$$\mathbf{y} = E[\mathbf{d}|\mathbf{x}] = \sum_{k=1}^{K} g_k \mathbf{y}_k \tag{6}$$

It might be helpful to graphically represent the underlying ideas of modular neural networks. This representation will necessarily be done on the problem with one-to-one mapping. The extensions to the multivariate mapping should be obvious. Assume that we want our modular NN to model a simple one dimensional function $y = \text{abs}(x)$.

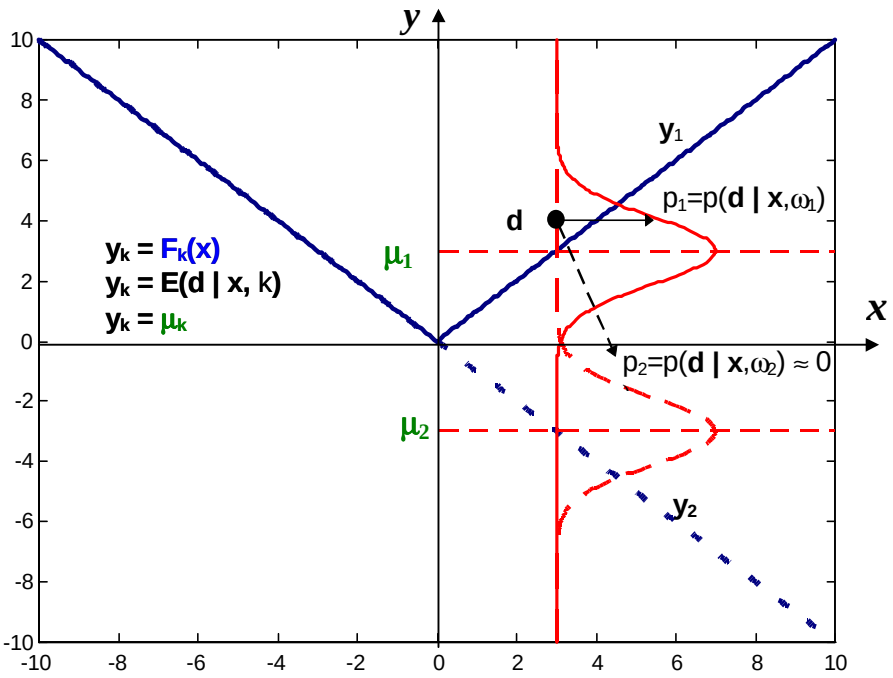

Fig. 2: Geometrical representation of probabilistic data modeling

Conditional probabilities p_1 and p_2 from (5), for some measured data point (x, d), are represented as one dimensional Gaussian functions in Fig. 2. Note that we may think of the network output y_k in three different ways and we stressed this fact by writing y_k as: being the underlying function F_k we are interesting at, as the expected value of our data conditioned on both $\boldsymbol{x}$ and chosen expert k (or as being the conditional mean μ of data.

Eq. (6) represents a finite associative Gaussian mixture model *associated* with a set of measured data $\boldsymbol{X} = \{\mathbf{x}(j), \mathbf{d}(j), j = 1,...,N\}$ consisting of the input vector **x** and output, or system response, vector **d.** The goal of the learning algorithm will be to model the distribution of the given set of recorded data. The choice of learning approach should be made between the two most widely used techniques in the estimation of model parameters (here, vectors **w** and **v**): minimisation of the sum of error-squares or maximisation of the likelihood function. It turned out that for a Gaussian mixture model the method of the maximisation of log-likelihood is more effective and generally preferred one. So, having the set of N measured data pairs performance index to be maximised looks as,

$$L(\mathbf{X}, \mathbf{w}, \mathbf{v}) = \log \prod_{j=1}^{N} p(\mathbf{d}_j \left| \mathbf{x}_j \right.) = \sum_{j=1}^{N} \log \sum_{k}^{K} g_{kj} p_{kj} \tag{7}$$

The problematic part with the learning of mixture densities is that the *log* can not be shifted behind of the second summation sign. At the same time, the problems with the first summation sign could be avoided simply by dropping it out. In the terms of NN learning that means that the on-line learning is taking place. What remains is to maximise,

$$L(\mathbf{X}, \mathbf{w}, \mathbf{v}) = \log \sum_{k=1}^{K} g_k p_k \tag{8}$$

with respect to **w** and **v.** It should be mentioned that two basic approaches to the maximisation of log-likelihood are: *standard gradient ascent procedure* and the *expectation maximisation (EM) technique*, with their many variants. It seems that the later method is much more viable approach for the finite mixture models. **(**More on comparisons of different optimisation procedures for modular NN could be found in [1] to [5]).

Here, we propose and test slightly different approach to learning in modular networks than as in [1] to [5]. First, it should be realised that in the system identification tasks basically, the kind of regression problem is to be solved. For such a problem the AF of the output-layer neurons of the experts should be linear. In such a situation the sound procedure for the learning of weights is to split the optimisation in two different learning schemes. One for **w** and one for **v**. Note that this splitting introduces a switch to 'multicriterial' optimisation in which the learning of the expert weights leads to the minimisation of the sum of the least-squares of error and adapting of the gating weights

results in maximisation of the log-likelihood given by (8). So, we suggest to use the so-called*; Mixed Optimisation Procedure (MOP)* firstly proposed and tested in [8]. There, for three different binary and analog type of mapping problems the superiority (in terms of the iteration steps) of the MOP learning in comparison to the other very well known and established learning methods was demonstrated. Taking the same approach for the identification tasks by using modular NN seems to promise similar improvements toward the EM and gradient type maximisation schemes.
Standard gradient type learning aimed to maximise L will take place if the gating and expert weights would be adapted according to the learning scheme given below ([3]),

$$\mathbf{w}_k(n+1) = \mathbf{w}_k(n) + \eta h_k(n)(\mathbf{d}(n) - \mathbf{y}_k(n))\mathbf{x} \tag{9}$$

$$\mathbf{v}_k(n+1) = \mathbf{v}_k(n) + \eta[h_k(n) - g_k(n)]\mathbf{x} \tag{10}$$

Where,

$$h_k = \frac{g_k \exp(-\frac{1}{2}(\mathbf{d} - \mathbf{y}_k)^T \Sigma_k^{-1}(\mathbf{d} - \mathbf{y}_k))}{\sum_{k=1}^{K} g_k \exp(-\frac{1}{2}(\mathbf{d} - \mathbf{y}_k)^T \Sigma_k^{-1}(\mathbf{d} - \mathbf{y}_k))} \tag{11}$$

is defined as *a posteriori* probability associated with the output of the *k-th* expert, *n* represents the iteration step and η is the learning rate. (Here, the covariance matrix Σ was not a subject of learning and equalled to identity matrix).

Basically, the change in the optimisation procedure of the expert output layer weights is only proposed here. The equations given above follow directly from the steepest ascent approach and they result in the incremental changes of the gating as well as of the expert output layer weights. But for the experts, having linear AF in output neurons, there is no need to calculate output layer weights using gradient methods. Rather, the approach from the linear in parameter estimation schemes as in standard least-squares learning is proposed for the calculation of the expert weights. Practically, this means that for the given outputs from the gating module the weights $\mathbf{w}$ will be obtained through a one-step procedure (using a pseudoinversion of the corresponding matrix). So, in each iteration step (and iteration will take place because of the gating weights learning) and for the given *g*, **w** will represent the best least-squares error solution. The calculation of the gating module weights will not be altered with this change and equation (10) will remain the same. By its very nature this *mixed optimisation procedure (MOP)* (calculation of gating weights takes place in gradient fashion and of the expert networks ones by using the pseudoinversion) is the *batch algorithm.* (Common practice in identification is to use the recursive variant instead and we worked with this on-line technique, too). The proposed MOP approach is both intuitively and in geometrical terms an easily understandable one:
Suppose, that we, by chance, start with the ideal or close to the 'best' gating module weight vectors ie., there is no need to learn them and the remaining task will simply be

to find the expert weights. This problem could be solved in gradient fashion presented above with (9), which results in incremental changes of **w.** But, this procedure is generically featured with many iterative steps and long learning times. Another possible approach could be in using the least-squares error method which requires just one pseudoinversion ie., the solution will be reached in one-step procedure. Here the second approach is used. In general, the starting random gating module weights vectors will be far from the ideal values and they will be the very subject of the optimisation, too. But, there is no reason (besides that of the appreciation of the correct mathematical model originated from the chain rule for the functions' derivative) to apply the same methodology for obviously different mapping schemes between **w, v,** L and approximation error *e:* the nonlinear mapping of **v** to L and the linear one of the expert weights **w** to the approximation error *e*. The chain rule structure of equation (10) remains intact but in the calculation of g the best in the least-squares sense expert modules output layer weights **w** will take part and this will speed up the convergence of the learning.

The MOP learning starts with randomly chosen gating networks weights **v**. After calculating g-s, expert networks output layer weights (providing that the output layer neurons are linear) can be calculated as follows. Starting from (6) we can write

$$\mathbf{y} = g_1 y_1 + g_2 y_2 + ... + g_K y_K$$

$$\mathbf{y} = g_1(w_{11}x_1 + w_{12}x_2 + ... + w_{1n}) + ... + g_K(w_{K1}x_1 + w_{K2}x_2 + ... + w_{Kn})$$

$$\mathbf{y} = \mathbf{M}\,\mathbf{w} \qquad (12)$$

The structure of the matrix **M** is obvious and the best in the least-squares sense expert networks output layer weight vector **w** using measured training data **d** can be found using standard formula,

$$\mathbf{w} = \mathbf{M}^{+}\mathbf{d} \qquad (13)$$

where $\mathbf{M}^{+}$ stands for the pseudoinversion of the matrix **M**. Obviously, the solution of (13) could be obtained recursively, too. Note that using constant bias input the maximal dimension of matrix **M** would be $(N, K(n+1))$, where N is the number of data and n represents the dimension of input signal **x.** If the calculation of the expert weights **w** would be done in recursive fashion the overall learning in modular network would be the 'pure' on-line procedure. We make use of mixture learning: the calculation of **v** was done in on-line fashion and **w** was obtained using the different size batches of the data. The smallest batch is obviously the one that results in $(K(n+1), K(n+1))$ matrix **M**. In order to better filter out the noise, it is better to take and use the batches with more than $K(n+1)$ data. If the size of data set is not to big (computing facilities dependent) one can always use all data. (Numerically critical part connected with the operation of pseudoinversion is not the number of rows of **M** but the number of its columns that is related to the dimension of input ***x*** and to the number of experts K).

Simulation results

As an application of the proposed MOP learning the identification of two different nonlinear dynamic systems was performed.

Example 1: Nonlinear Noisy System Given by Nonlinear Difference Equation

The behaviour of the first order nonlinear system is governed by the next nonlinear difference equation,

$$y_{k+1} = (0.9 - 0.003y_k)y_k + 0.2u_k \qquad (14)$$

with sampling time $T_s = 1\ s$, and a state-dependent time constant of about $T = 10\ s$. Such type of the dynamics is difficult to identify by classical methods when the mathematical structure of the nonlinearity is unknown, because the nonlinearity cannot be separated from the linear part like eg. in Hammerstein model. The characteristic feature of this model is that for small input steps it shows a fairly linear behaviour whereas for the bigger deviations from steady state, the positive step responses are different from the negative ones. Measurement data are spoiled with a noise with variance $0.05y_{\max}$. In order to obtain a good model of a nonlinear process it was important that the learning data cover the whole relevant space and contain a rich spectrum of frequencies as well as of magnitudes. Therefore the process is excited with a pseudo-random binary noise (PRBS) signal with the amplitude modulation. The training set of 621 data samples is plotted in Fig. 3.

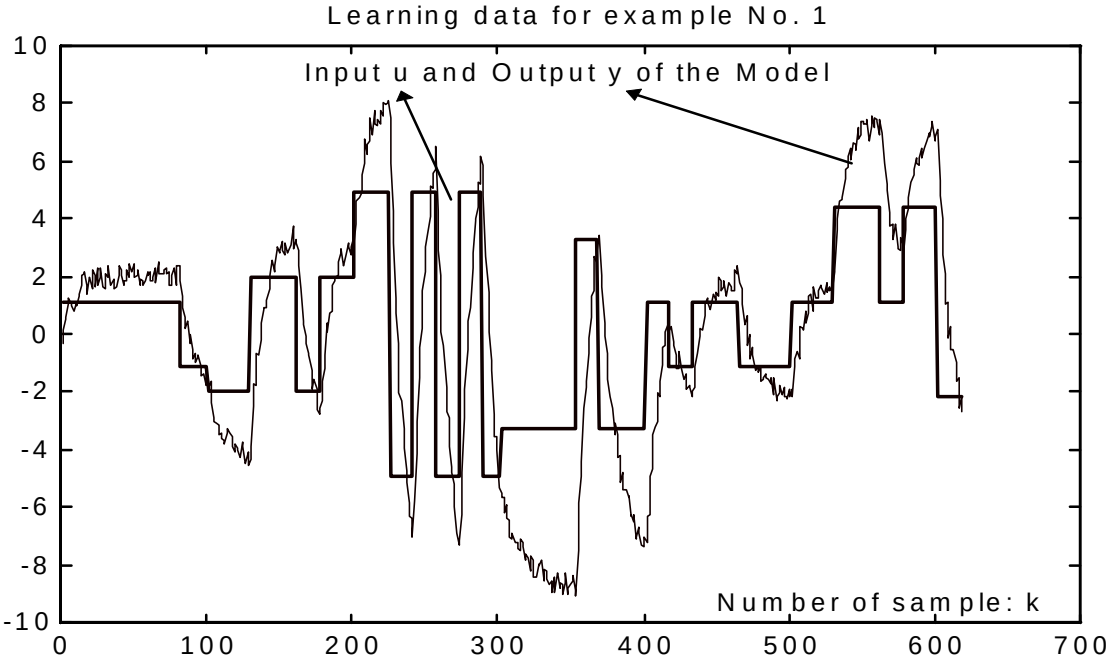

Fig. 3: Learning data set for the nonlinear difference equation (14)
(Stepwise smooth function is input *u*. Hairy curve is system response y)

To verify the model in the sense of generalisation, the input to the trained modular NN was changed and the response of the NN to the saw-type input was compared with the noisy response of the original system represented by (14) to the very same input. Fig. 4 shows a good generalisation capabilities of the modular NN as well as a capacity for a good filtering of given noise.

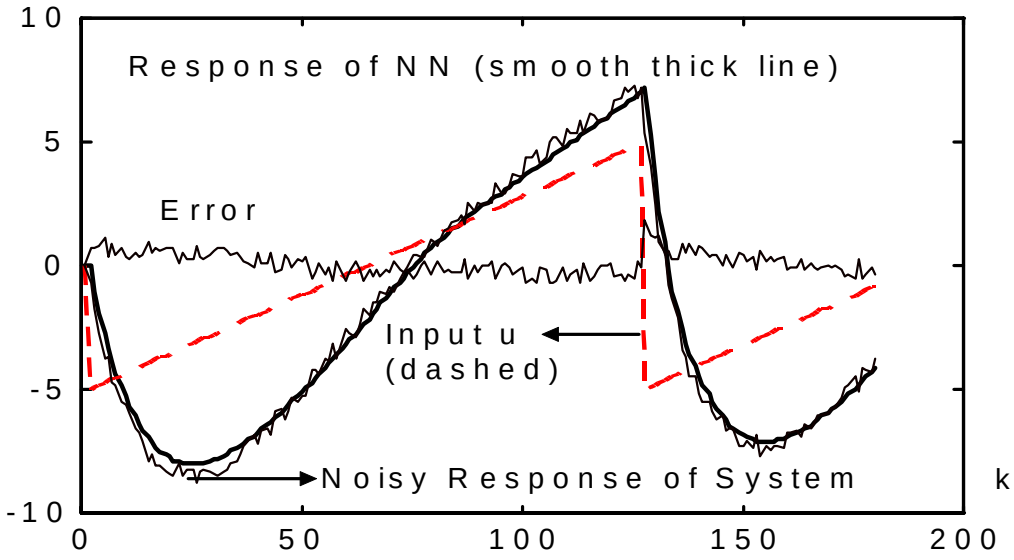

Fig. 4: Verification run driven by previously unseen saw-type input

Example 2: Identification of the Car Engine Throttle Valve

Next example is represented by a real world system. The modular NN should have found the relations between the voltage of servo motor (input variable) and the throttle valve position angle (output variable) of the car engine throttle valve. A set of 3000 training data was recorded on the real test stand [9]. During the learning the first 1250 data pairs have been used and the verification with the 'frozen' or 'fixed' previously learned weights has been done on the whole range of measured data set, meaning by using the recorded voltage signal from $k = 0$ to $k = 3000$ as an input to the NN. As it could be seen in Fig. 5 quite a good generalisation has been taking place along the whole transient period ($k = 0$ - 3000) in the sense that there is no big difference in the dynamic behaviour in the second (unlearned) part of the transients in comparison to the first 1250 response samples.

It should be mentioned that the modular NN gave more than comparable results to the identification results presented in [7] where Radial Basis Function NN and Fuzzy Models have been implemented and compared. Here, 12 weights and 16 in [7] have been used. The quality of identification was similar. More investigation on different

problems of various size and complexity is still needed in order to make any kind of such comparisons, though.

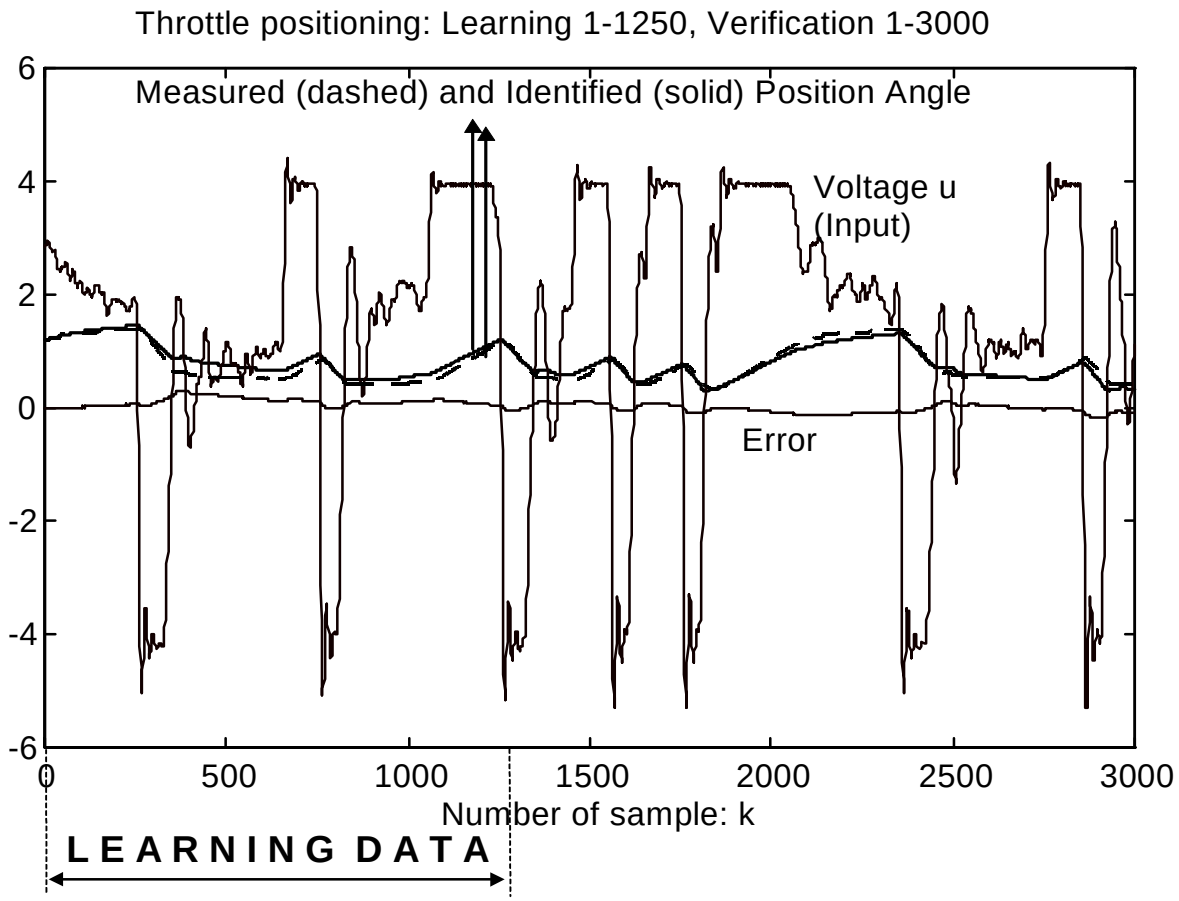

Fig. 5: Car engine throttle positioning identification

Conclusions

The identification of two nonlinear dynamic systems using modularly structured neural network with the new learning algorithm for the learning of both gating and expert networks weights was presented. The problem of learning was formulated and treated as a mixture estimation problem in which the log-likelihood function should has been maximised. But, as opposite to the other methods we combined the gradient type of learning for gating weights with a least-squares algorithm for the learning of expert networks weights. Here, the experts were simple one-layer nets with a single linear output unit. The very result of such an approach is a simple structured modular network with improved learning and it seems with good capabilities for the identification of general nonlinear dynamic systems. The results of the identification of discrete-time nonlinear system corrupted with noise as well as of a car engine throttle valve from real data set of 3000 recorded samples of measured noisy data are very promising. In both

cases the learning was an one epoch procedure. It seems that the learning is a robust one in the sense that the results are not heavily dependant on the weights initialisation and on the learning rate. This robustness was supported by proper scaling of input variables. The MOP approach is applicable whether the expert networks have hidden layer(s) or not, what was the case here. In the former case the training of the expert output layer weights should be done as described above and hidden layer(s) weights could be learned by any standard (gradient or second order descent algorithms) or novel (eg, genetic algorithm) procedure.

References

1. Jacobs, R. A., Jordan, M. I., **A Modular Connectionist Architecture for Learning Piecewise Control Strategies**, Proc. of ACC 1991, TP1, pp. 1597-1602, 1991
2. Jacobs, R. A., Jordan, M. I., Nowlan, S. J., Hinton, G. E**., Adaptive Mixtures of Local Experts,** Neural Computation 3, 79-87, 1991
3. Jordan, M.I., **Connectionists Models of Cognitive Processes**, Lectures-Course 9.641, MIT, Cambridge, MA, 1993
4. Jordan, M. I., Xu L., **Convergence Results for the EM Approach to Mixtures of Experts Architectures**, AIL & CfBCL A.I. Memo No. 1458, MIT, Cambridge, MA, 1993
5. Jordan, M. I., Jacobs, R. A**., Hierarchical mixtures of experts and the EM algorithm,** Neural Computation 6, 1994
6. Kecman, V., **Application of Artificial Neural Networks for Identification of System Dynamics**, MIT, Dept of ME, Tech. Rep., TR 93-YUSA-01, Cambridge, MA, 1993
7. Kecman V., Pfeiffer B-M., **Exploiting the Structural Equivalence of Learning Fuzzy Systems and Radial Basis Function Neural Networks**, Proc. of 2nd Europ. Congr. on Intell. Techn. and Soft Comput., EUFIT ‘94, Aachen, Vol. 1, pp. 58-66, 1994.
8. Kecman V., **The Ultimate EBP Scheme: Mixed Optimisation Procedure**, Submitted to the IEEE Transactions on Neural Networks, Oct. 1994.
9. Pfeufer, T., **Improvement of Flexibility and Reliability of Automobile Actuators by Model-Based Algorithms**, IFAC Symp. on Intell. Comp. and Instr. for Cont. Applic., SICICA ‘94, Budapest, 1994