Experience with FIR BP neural networks for prediction problems

STANISLAV KALETA, DANIEL NOVOTNY, PETER SINCÁK
Center for Intelligent Technologies, Computational Intelligence Group
Department of Cybernetics and AI
Faculty of Electrical Engineering & Informatics
TU Kosice, Letna 9, 040 00 Kosice
SLOVAK REPUBLIC
(draft paper)

Abstract: - Paper deals with experience of application of FIR filter (Finite Impulse Response) in BP neural networks used for prediction system. Prediction can be defined as extrapolation of unknown function determine by representing data sets. One of the most difficult problems in prediction systems is determination of the size of the time series variable input. It is always very difficult to determine the right size of the “history” of input variable necessary for prediction – extrapolation of the modeled function. So in general prediction problem can have input e.g. \((x(t), x(t-1), \ldots, x(t-\tau))\), where \(\tau\) - determine the width of the input window size and output \((y(t+1), y(t+2), \ldots, y(t+\lambda))\), where \(\lambda\) - is size of the length of the time series of the predicted variable and \(x\) and \(y\) are vectors. Usually both \(\tau\) and \(\lambda\) depend on the problem and in case of output the size of the \(\lambda\) is small then there is bigger chance that prediction will be more reliable. The avoid a problem of \(\tau\) determination a FIR neural networks offer solution to gather “history” of the unknown function inside of neural network topology, namely on neural networks synapses in the form of FIR filters. The length of the filters indicates the time-delay of the input. Some modification of FIR concerning Adaptive FIR NN was under research and self-adaptation of length of particular FIR filters associated with each synaptical weight. Experiments were done on selected SANTA-FE data for prediction problems and also on real-world data related to power engineering domain with aim to make electricity load forecast possible. Results indicate that FIR NN is an interesting tool for prediction problem with possibility to avoid determination of the input window size. Comparison study between FIR NN and Adaptive FIR NN shows some advantages of adaptive FIR over the non-adaptive approach.

Key-Words: Prediction, function extrapolation, Finite Impulse response filters (FIR), adaptive FIR, input window size.

1 Introduction

Intelligent technologies represent a very interesting direction in current and future technology. These technologies are based on AI techniques including neural networks and other intelligent approaches. Prediction problems are actual in any part of everyday life including technological processes in industrial as well as non-industrial domain. Prediction is in fact an extrapolation of the unknown function, which is being approximated by neural network or any statistical tools. To develop a suitable prediction system is a problem dependent problem and very interesting and challenging task. There are many problems in setting up such a prediction tool including window size determination of the input to the prediction system. The length of the window represents an amount of historical data according to that can be used for the next state prediction. Prediction is in fact extrapolation of the unknown function where usually we have in the input

\[(x(t), x(t-1), \ldots, x(t-\tau))\]

where \(\tau\) - determine the width of the input window size and output \((y(t+1), y(t+2), \ldots, y(t+\lambda))\), where \(\lambda\) - is size of the length of the time series of the predicted variable and \(x\) and \(y\) are vectors. Always the

\[\tau \gg \lambda\]

and the easiest situation is when \(\lambda=1\) or only one value is predicted. Certainly problems if \(\lambda>>1\) are more difficult and prediction task is more complex and difficult. Generally we can divide prediction procedure into 2 basic types:

1 Research is supported via project # 9433 awarded by Slovak National Agency for Science “Computational Intelligence in Decision Procedures” for 1999-2001
Fig. 1. Input to the neural network represents a window of the time series \((t-1), (t-2), (t-\tau)\) related to the input variable and the output is a single value related to time \((t+1)\). Iterations are connected to shifting the input window on the training curve as it is shown on above figure (iterations 1-4).

- prediction of a single value of the variable – one step prediction
- prediction of more than one values of the variables – multi-step prediction.

In case of multi-step prediction 2 basic approaches are used:

- on the output is expected a time series of the output variable
- on the output is expected the single variable predicted \(\lambda\) - times – so in fact it is a single step prediction applied \(\lambda\) - times and predicted values are incorporated into multi-step prediction so output \(y(t+1)\) is an input in situation when we predict \(y(t+2)\). This approach is called recursive prediction approach.

The aim is still to make a multi-step prediction with the highest possible accuracy. The quality of extrapolation is being measured by expressing a difference between predicted and real data considering averaging the prediction errors through data used for prediction procedure. Many very useful sources about prediction and neural networks can be found in [1,2,3]

2 Motivation of the Project

The main objective of the project was to obtain practical experience with prediction systems, which do not require having a window (time series) of input variable to provide “historical” information for prediction (extrapolation) purposes. Whereas simple feed-forward network can be trained to accomplish pattern classification tasks with complex nonlinear boundaries, they are limited to processing static patterns – patterns that are fixed rather than temporal in nature. We test (and do some minor modifications) here a network that overcomes this limitation and that is capable of processing temporally modulated signals. Networks with this capability can play an important role in applications domains that have naturally time-varying properties to their signals and dynamic situations.

The main reason is the problem of the size of the mentioned window. In biological systems the neuron consists of the soma and axon, which ends with connection based on synopsis. The length of the axon, speed of signal distribution, quality or membrane and many other factors influence the time delay of the signal distribution. So the time related model is in the connection between neurons and in this connection the information about the time-series is being encoded.

Fig. 2. The illustration of biological neuron and axon connection to other neurons is presented on this figure. Time series data is being encoded into this structure and so the history recording depends on the topology of the neural networks.

The simulation of the about approach in very simplified version is in putting a FIR filters in the synapses of neural network. That means that instead
of having a single parameter (synaptic weight) we will have more parameters incorporated into the FIR filters associated with each weight.

Figure 3. Practical implementation of FIR filters into synapses of artificial neural network is on this figure.

The challenge of this project is to modify the BP approach, verify the approaches about adaptive FIR filters on each synaptic connection. FIR filter neural network description can be found in [4].

3 Error backpropagation for FIR neural networks

Application of Error backpropagation includes temporal error BP approach to the FIR neural training. If we assume that the normal BP neural networks was based on regular Error function as follows:

$$\sum_{i} = O_{ip} - p_{ip}^i P_{xevJ}^2$$

where $p_{ip}^i$ is expected value on the neural network for $i$-th output neuron concerning input pattern “p”. Also $x_{ip}^i$ represents the real output of on $i$-th output neuron concerning input pattern “p” while $N_0$ is number of neuron in output layer. Then a usual BP-like adaptation rule is used as follows:

$$\Delta w_{ij}^l(n) = -\gamma \delta_{j}^{l+1}(n) x_{i}^{l}(n)$$

where

- $i$ - pre-synaptic neuron
- $j$ - post-synaptic neuron
- $n$ - iteration step
- $w$ - synaptic weight
- $\gamma$ - is learning parameter of learning
- $\delta_{j}^{l+1}$ - is an error signal associated to neuron “j” which is in layer “l+1”
- $x_{i}^{l}$ - is value of activation of pre-synaptic neuron “i” in layer “l”

There is basic difference between regular BP approach and this applied for FIR neural network as follows:

- It is associated not a single synaptic weight to each synapses but a vector of weights as it can be seen on figure 3.
- The propagation of Error is being effective after number of inputs, which will propagate through the overall neural network up to the end. Only after then an Error will occur and can be propagated back to the neural networks including FIR filters.

Concerning the FIR filters on neural network which have length of the FIR filter “$M$” and there are L layer of the neural networks then we can express the adaptation as follows:

$$\Delta w_{ij}^{L-1}(n) = -\gamma \delta_{j}^{L-n}(n-(L-1)M) x_{i}^{l-(L-1)n}(n-(L-1)M)$$

and adaptation starts for $n > (L-1)M$ to be able to fill all filters with signal in the neural networks.

4 AFIR - adaptive time-delays neural network

Both time delays $\tau_{ijk}$ and connection weights $w_{ijk}$ are adapted on-line according to a gradient descent approach. AFIR as adaptive FIR network in meaning of adaptation of time-delays in weight connections is equivalent to ATNN (Adaptive time-delays neural network), presented in [2]. This network adapts its time delay values as well its
weights during training, to better adapt to changing temporal patterns, and provide more flexibility for optimal tasks. Furthermore, time-windows are not used as in previous (FIR) example. After adaptation of ATNN (AFIR) is structure of network transformable to FIR one, only some of connections according some of time-delays are cut or added. Because of way of implementation and similarity of the weight structure, name AFIR was chosen (Adaptive FIR neural network). In AFIR are retro operators from FIR \( q^{-1} \) replaced by \( \tau_{jik} \), where \( q^{-1} \) corresponds to memory cells for one iteration and \( \tau_{jik} \) as variable, holds number of iteration for memory cell for which to hold signal value. Simply, one \( \tau_{jik} \) means as many \( q^{-1} \) time-delays to past.

\[
\Delta \tau_{jik}^{h-1}(t_n) = -\eta \frac{\partial J(t_n)}{\partial \tau_{jik}^{h-1}}
\]

and the final formula for time-delay adaptation is as follows:

\[
\Delta \tau_{jik}^{h-1} = \eta \rho_j^h (t_n) x_j^{h-1} (t_n - \tau_{jik}^{h-1})
\]

where \( \rho_j^h \) is error signal related to adaptation of “\( \tau \)” and \( \eta \) is the learning parameter for the same adaptation. The adaptation of the synaptic weight is as follows:

\[
\Delta w_{jik}^{h-1} = \gamma \delta_j^h (t_n) x_j^{h-1} (t_n - \tau_{jik}^{h-1})
\]

This is the way, how every filter, in the form of weight vector is expanding or reducing. After adaptation, \( \tau_{jik} \) are calculated, and only relevant connections of time delay remain. Certainly they may be different on every synapse, because it is not necessary to have the same number of delays for different units in the same layer or the same delay value from different units in general, since the computation is local for each interconnection. This possibility give a very big variation of results and adaptation of “time-delays” can be various on each synapses. This represents the idea of “history” encoding of the input in sufficient way to be able to generate the prediction output. We are estimating the proper historical time-delay window size according to Error function related to the neural network training procedure.

More details can be found in [5]. During this project many implementation problems were solved related to the training procedure. One of this is situation when adaptation time-delay “\( \tau \)” is similar or very close to the previous one and after the numerical rounding it gives the same results. The solution was made and the following averaging of time-delay “\( \tau \)” and addition weights is done.

\[
\tau_v = \frac{\tau_i + \tau_j}{2} \quad w_v = w_i + w_j
\]

More implementation details can be found on the [6].
5 Experiments and results

Both FIR and AFIR network with one hidden layer were trained on the same data, with more different topologies. These networks were trained on the same data sets. The “A” set from Santa - Fe data for prediction problems and also real-world data related to power engineering domain - electricity load forecast were chosen. The first set consists of 1000 points and indicate chaotic intensity pulsations of an \( \text{NH}_3 \) laser in chaotic state. The second set of 2000 points represents samples of electricity load measured in one junction of energetic system.

As the first step should be data preprocessing, which includes transformation into interval \( <0;1> \) in this case. Data sets were divided into three intervals: training, validation and testing set. The initial synapse weights were random numbers with uniform distribution between –1 and 1. Topologies of networks had one input and one output neuron with different numbers of neurons in hidden layer and there were used more different initial parameters for trained networks:

- number of hidden neurons: 5; 25
- (max) length of the filters: 10; 30
- of sigmoid activation function: 0.1; 1
- interval of function values of sigmoid: \( <0;1>; <-1;1> \)
- adaptation coefficient for \( w \): 0.001; 0.01; 0.1
- adaptation coefficient for \( \tau \): 0.001; 0.01; 0.1
- momentum: 0.02; 0.2

There are parameters used for experiments and all combinations of them were investigated. The topology of neural network was designed with a single input and single output and with one hidden layer with various numbers of neurons. All networks were trained on the data for some small number of iterations. When error lowered, network was retrained for next 1000 iterations. For comparison purposes was used normalized mean error:

\[
\text{ME} = \frac{1}{N} \sum_{i=1}^{N} |x_i - d_i|
\]

where \( x_i \) is pattern value, \( d_i \) is predicted value and \( N \) is number of patterns. Best results for recursive prediction on test data are in following tables and graphs.

<table>
<thead>
<tr>
<th>Laser</th>
<th>ME on testing set average for 3 steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIR MLP</td>
<td>0.043</td>
</tr>
<tr>
<td>AFIR MLP</td>
<td>0.0076</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Energy</th>
<th>ME on testing set average for 3 steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIR MLP</td>
<td>0.062</td>
</tr>
<tr>
<td>AFIR MLP</td>
<td>0.0376</td>
</tr>
</tbody>
</table>

Table 1, 2 : Comparing of errors on laser and electricity load data

6 Conclusion

The obtained experience confirms that idea of avoiding a problem of the size of input window is possible to solve and FIR and AFIR approach were tested on real world data. Some implementation modifications were done and we obtained the prediction results from both approaches. The results show that AFIR is useful for more “complex” problems when prediction is more difficult and results on simple FIR are not promising. Adaptation of FIR filters ends up with different lengths of FIR filters in each synapsis and therefore we assume that this approach is useful and effective to obtain the best prediction results. Generally it is believed that this type of research should be progressing and some more experiments on various data will be underway in the close future.

References:

Graph 1. Prediction results on real-world data from electricity load forecast in Slovak Republic. The lower graph is the original data and the upper graph is a predicted curve. From 1-1600 was used for training, 1601-1800 was a validation set and 1801-2000 was a testing set.

Graph 2. Prediction results on SanteFe data type A. The lower graph shows the original data and the upper curve represents the prediction results.