13th Int. Workshop on Generalized Nets
London, 29 October 2012, 49–55

Modelling the backpropagation algorithm of the Elman neural network by a generalized net

Sotir Sotirov

"Prof. Asen Zlatarov" University
1 "Prof. Yakimov" Blvd, Burgas–8010, Bulgaria
e-mail: ssotirov@btu.bg

Abstract: The proposed GN model presents the functioning of recurrent neural networks. Here we discuss the Elman network and the 'backpropagation' algorithm for learning. In comparison with other types of neural networks, here we describe the process in its temporal development. In a series of papers, we have described many different neural networks using the apparatus of generalized nets. The present research deals with another kind – neural network with feedback into the hidden layer.

Keywords: Neural networks, Recurrent neural networks, Elman neural network, Generalized net, Backpropagation algorithm.
AMS Classification: 68Q85, 62M45.

1 Introduction

The Elman neural network (ENN) is described commonly as a two-layer network with feedback from the first-layer output to the first-layer input, [4]. This recurrent connection allows the Elman network to both detect and generate time-varying patterns. A two-layer Elman neural network is shown on the Figure 1, where:

- a^m is the exit of the m- layer of the neural network for m = 1, 2, 3;
- w is a matrix of the weight coefficients of the everyone of the entries;
- b is neuron's entry bias;
- f^m is the transfer function of the m-layer.

The Elman network has neurons in its hidden (recurrent) layer with hyperbolic tangents transfer function, and neurons in its output layer with linear transfer function. This combination is special in that two-layer networks with these transfer functions can approximate any function (with a finite number of discontinuities) with arbitrary accuracy. The only requirement is that the hidden layer must have enough neurons. More hidden neurons are needed as the function being fitted increases in complexity.

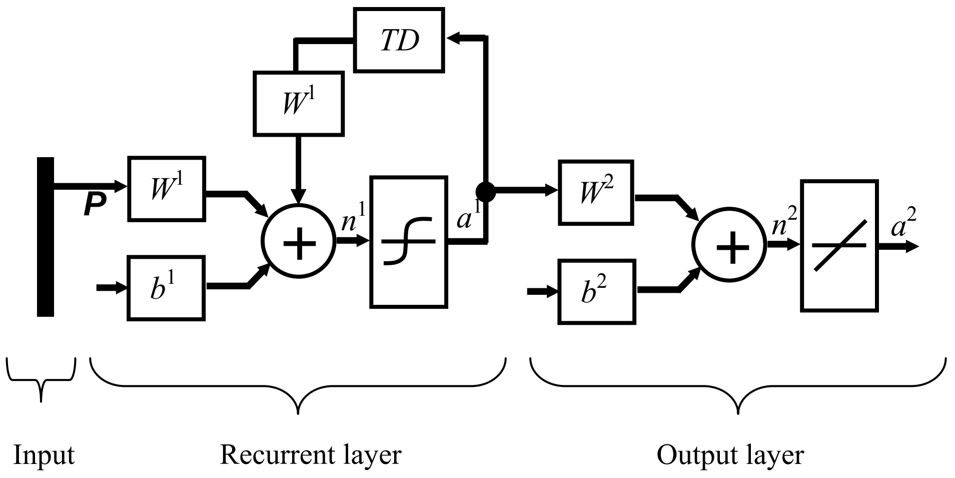

Figure 1: Two-layers Elman neural network

The Elman network differs from conventional two-layer networks in that the first layer has a recurrent connection. The delay in this connection stores values from the previous time step, which can be used in the current time step.

Thus, even if two Elman networks, with the same weights and biases, are given identical inputs at a given time step, their outputs can be different because of different feedback states.

Because the network can store information for future reference, it is able to learn temporal patterns as well as spatial patterns.

The neuron in the first layer receives outside entries *p*.

The neurons's exits from the last layer determine the neural network's exits *a*.

Since it belongs to the supervised learning methods, to the algorithm are submitted couple numbers (an entry value and an achieving aim – on the network's exit)

$$\{p_1, t_1\}, \{p_2, t_2\}, ..., \{p_Q, t_Q\}, \tag{1}$$

$Q \in (1, ..., n)$, n being the number of learning couples, where p_Q is the entry value (on the network entry), and t_Q is the exit's value replying to the aim. Every network's entry is preliminary established and constant and the exit have to reply to the aim. The difference between the entry values and the aim is the error $e = t - a$.

The "back propagation" algorithm [6] use least-quarter error:

$$\hat{F} = (t - a)^2 = e^2. \tag{2}$$

While learning the neural network, the algorithm recalculates the network's parameters (W and b) so to achieve least-square error.

The 'backpropagation' algorithm for the i-th neuron, for the $(k + 1)$-st iteration, uses equations:

$$w_i^m(k+1) = w_i^m(k) - \alpha \frac{\partial \hat{F}}{\partial w_i^m}, \tag{3}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha \frac{\partial \hat{F}}{\partial b_i^m}, \tag{4}$$

where:

- α is the learning rate for neural network;
- $\frac{\partial \hat{F}}{\partial w_i^m}$ is the relation between changes of square error and changes of the weights;
- $\frac{\partial \hat{F}}{\partial b_i^m}$ is the relation between changes of square error and changes of the biases;

The network is considered learned when

$$e^2 < E\text{max}, \tag{5}$$

where Emax is maximum square error.

The fundamental part, giving the algorithm's name "Back Propagation" [2, 3, 6, 9], is the way of calculating the error in the many-layered neural network. For error's calculation, it is used the notion "sensibility". It reflects the alteration of the error's function compared to the alteration of every one of the weight coefficients and biases. The sensibility is the basic implement for calculating the new weight coefficients and biases.

For the calculation of the sensibility of the M-th layer, it is necessary to know this sensibility for the $(M + 1)$-st layer, so in this way the sensibilities are propagating back through the network, from the last layer to the first one:

$$s^M \to s^{M-1} \to \ldots \to s^2 \to s^1.$$

This backward propagation gives the algorithm's name.

In our previous research [8], we described the forward propagation of the process – calculating of the outputs of the Elman neural networks [2, 3]. Here, we will describe a backpropagation of the calculating of the weight coefficients and biases.

Shortly, the backpropagation algorithm (according to [9]) can be described for each epoch in the following manner:

1. The entire input sequence is presented to the network, and its outputs are calculated and compared with the target sequence to generate an error sequence (forward propagation).
2. For each time step, the error is backpropagated to find gradients of errors for each weight and bias. This gradient is actually an approximation since the contributions of weights and biases to errors via the delayed recurrent connection are ignored.
3. This gradient is then used to update the weights with the backpropagation training function chosen by the user (here we describes as s f^m).

2 Constructing the generalized net

All definitions related to the concept of generalized nets (GNs) are taken from [1]. The network, describing the work of the neural network learned by the backpropagation algorithm [4], is shown on Figure 2.

The constructed GN-model, as illustrated on Figure 1, is a reduced one. It does not feature temporal components, and the priorities of the transitions, places and tokens are equal, with places' and arcs' capacities being equal to infinity.

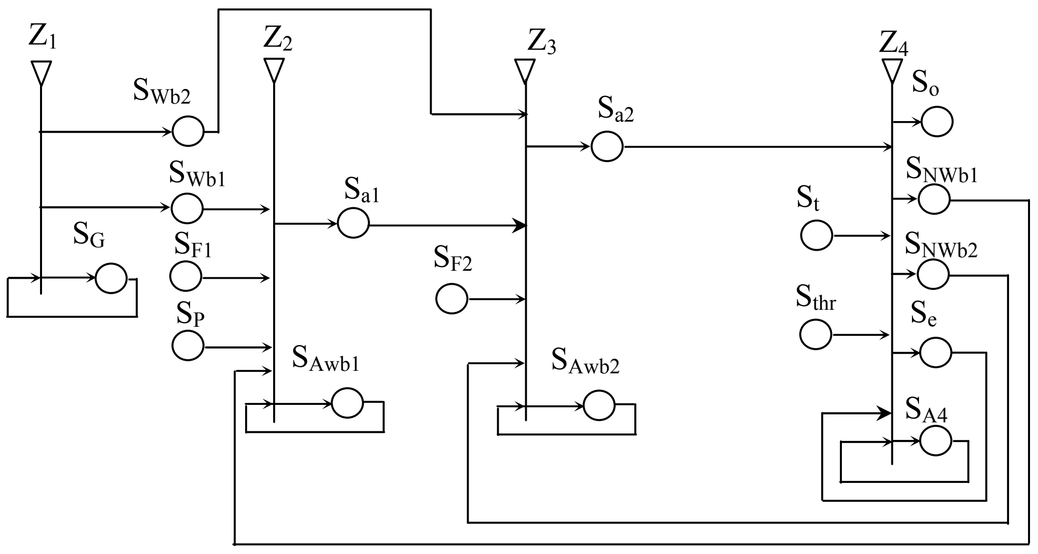

Figure 2: Generalized net model of the backpropagation algotirhm of the Elman neural network

Initially there is one α-token that is located in place S_G with characteristic

$$x_0^{\alpha} = \text{“Generator of random values between 0 and 1”}.$$

In the next time moment, this token will generate a new α-tokens (α', α'' and so on). The original α-token will continue to stay in place S_G, while the other α-tokens will move to transitions Z_2 and Z_3 via transition Z_1.

Initially the following tokens enter in the generalized net:

- in place S_{F1} — one δ'-token with characteristic $x_0^{\delta'}$ = “transfer funcion f^1”.
- in place S_P — γ''-token with characteristic $x_0^{\gamma'}$ = “$\{p_1\}$, $\{p_2\}$, ..., $\{p_Q\}$”;
- in place S_{F2} — one δ''- token with characteristic $x_0^{\delta''}$ = “transfer funcion f^2”;
- in place S_t — γ''-token with characteristic $x_0^{\gamma''}$ = “$\{t_1\}$, $\{t_2\}$, ..., $\{t_Q\}$”;
- in place S_{thr} — β-token with characteristic x_0^{β} = “Threshhold value for the least square error”.

The generalized network is present by the set of transitions [1]:

$$A = \{Z_1, Z_2, Z_3, Z_4\},$$

where the transitions describe the following processes:

- Z_1 — generating random vector for values of the weight matrix W and b;
- Z_2 — calculating the forward values of the firts layer;
- Z_3 — calculating the forward values of the second layer;
- Z_4 — checking if the neural network is learned or calculating the new weight coefficients and neural network’s bias.

The four transitions are described in details below.

$$Z_1 = \langle \{S_G\}, \{S_{Wb1}, S_{Wb2}, S_G\}, R_1, \wedge(S_G) \rangle$$

where:

$$R_1 = \begin{array}{c|ccc} & S_{WbN} & S_{Wb} & S_G \\ \hline S_G & W_{G,WbN} & W_{G,Wb} & True \end{array}$$

and the predicates in the index matrix R_1 have the form: $W_{G,Wb1} = W_{G,Wb2}$= "Random vector for calculating the network exit is generalized".

The token that enters place S_{Wb1} obtains characteristic $[W^1, b^1]$. The token that enters place S_{Wb2} obtains characteristic $[W^2, b^2]$.

$$Z_2 = \langle \{S_{Wb1}, S_{F1}, S_P, S_{NWb1}, S_{Awb1}\}, \{S_{a1}, S_{Awb1}\}, R_2, \vee(S_{NWb1}, S_P, S_{Wb1}, \wedge(S_{F1}, S_{Awb1}))\rangle$$

where

$$R_2 = \begin{array}{c|cc} & S_{a1} & S_{AWb1} \\ \hline S_{Wb1} & False & True \\ S_{F1} & False & True \\ S_P & False & True \\ S_{NWb1} & False & True \\ S_{AWb1} & W_{AWb1,a1} & True \end{array}$$

and the predicate in the index matrix R_2 has the following meaning: $W_{Awb1,a1}$ = "The outputs of the first neural layer are calculated".

The token that enters place S_{a1} obtains characteristic $[a^1, W^1, b^1]$. The token that enters place S_{Awb1} obtains characteristic $[W^1, b^1]$.

$$Z_3 = \langle \{S_{Wb2}, S_{a1}, S_{F2}, S_{NWb2}, S_{Awb2}\}, \{S_{a2}, S_{Awb2}\}, R_3, \vee(S_{NWb2}, S_{a1}, S_{Wb2}, \wedge(S_{F2}, S_{Awb2}))\rangle$$

where:

$$R_3 = \begin{array}{c|cc} & S_{a2} & S_{AWb2} \\ \hline S_{Wb2} & False & True \\ S_{F2} & False & True \\ S_{a1} & False & True \\ S_{NWb2} & False & True \\ S_{AWb2} & W_{AWb2,a2} & True \end{array}$$

and the predicate in the index matrix R_3 is $W_{Awb2,a2}$ = "The outputs of the second neural layer are calculated".

The token that enters place S_{a2} obtains characteristic $[a^2, W^2, b^2]$. The token that enters place S_{Awb2} obtains characteristic $[W^2, b^2]$.

$$Z_4 = \langle \{S_{a2}, S_t, S_{thr}, S_e, S_{A4}\}, \{S_o, S_{NWb1}, S_{NWb2}, S_e, S_{A4}\}, R_4, \vee(\wedge(S_{a2}, S_t), \wedge(S_{thr}, S_e), \wedge(S_{A4}, S_e))\rangle$$

where:

$$R_4 = \begin{array}{c|ccccc} & S_o & S_{NWb1} & S_{NWb2} & S_e & S_{A4} \\ \hline S_{a2} & W_{a2,O} & False & False & True & W_{a2,A4} \\ S_t & False & False & False & True & False \\ S_{thr} & False & False & False & True & False \\ S_e & False & False & False & False & W_{e,A4} \\ S_{A4} & W_{A4,o} & W_{A4,NWb1} & W_{A4,NWb2} & False & False \end{array}$$

and the predicates in the index matrix R_4 have the meaning:

- $W_{a2,O}$ = "$e < t_{hr}$";
- $W_{a2,A4} = W_{e,A4} = W_{A4,O}$ = "$e > t_{hr}$";
- $W_{A4,NWb1} = W_{A4,NWb2} = \neg W_{a2,A4}$.

where e is the mean square error and t_{hr} is the threshold value for the mean square error.

In the first activation of the transition values in the token from the place S_{a2} subtracts from the value of the token from place S_t. This is the mean square error.

The token from the places S_e, S_{A4}, S_{a2} united in one token in place S_{A4} with characteristic "New weight coefficients, new biases".

The token that enters in place S_o obtains characteristic "output values of the weight coefficients, outputs value of the biases of the neural network".

The token that enters in places S_{NWb1} and S_{NWb2} obtain characteristics "New weight coefficients for the first layer, new biases for the first layer" and "New weight coefficients for the second layer, new biases for the second layer.

Conclusion

This model introduces the working method of the Elman neural network with forward propagation and its learning using the backpropagation algorithm.

For the construction of a model of the information processes in the so described structure, generalized nets are used, because they offer powerful apparatus for modeling of parallel processes and allow tracing their behavior in future, as well as their management and optimization.

The Elman networks are one of the dynamic kinds of neural networks, which are appropriate for prediction of processes in their temporal development.

References

[1] Atanassov, K. *Generalized nets*, World Scientific, Singapore, 1991.

[2] Gadea, R., F. Ballester, A. Mocholi, J. Cerda, Artificial Neural Network Implementation on a Single FPGA of a Pipelined On-Line Backpropagation, *Proc. of the 13th Int. Symposium on System Synthesis*, 2000, 225–229.

[3] Hagan, M., H. Demuth, M. Beale, *Neural Network Design*, PWS Publishing, Boston, MA, 1996.

[4] Haykin, S. *Neural Networks: A Comprehensive Foundation*, Macmillan, New York, 1994.

[5] Krawczak, M. *Generalized Net Models of Systems*, Bulletin of Polish Academy of Science, 2003.

[6] Rumelhart, D., G. Hinton, R. Williams. Training representation by back-propagation errors, *Nature*, Vol. 323, 1986, 533–536.

[7] Sotirov, S. Modeling the algorithm Backpropagation for training of neural networks with generalized nets. Part 1, *Proc. of the Fourth International Workshop on Generalized Nets*, Sofia, 23 September 2003, 61–67.

[8] Sotirov, S., E. El-Darzi, Generalized net model of the Elman neural network, *Proc. of the Eleventh Int. Workshop on GNs and Second Int. Workshop on GNs, IFSs, KE*, London, 9–10 July 2010, 21–26.

[9] Haykin, S., *Neural Networks and Learning Machines*, McMaster University, Canada, 2008.