A global model for fault tolerance of feedforward neural networks

Fernando Morgado Dias [1,2] and Ana Antunes[3]

[1] Departamento de Matemática e Engenharias, Universidade da Madeira
Campus da Penteada, 9000-390 Funchal
Tel: +351 291-705150/1, Fax: +351 291-7051993
[2]Centro de Ciências Matemáticas - CCM
Universidade da Madeira, Campus Universitário da Penteada
9000-390 Funchal, Madeira, Portugal
Tel: + 351 291 705181, Fax: + 351 291 705189
[3]Escola Superior de Tecnologia de Setúbal do Instituto Politécnico de Setúbal,
Departamento de Engenharia Electrotécnica, Campus do IPS, Estefanilha, 2914-508 Setúbal, Portugal Tel: +351 265 790000,
morgado@uma.pt, aantunes@est.ips.pt

Abstract. It is commonly assumed that neural networks have a built in fault tolerance property mainly due to their parallel structures. The international community of Neural Networks discussed these properties only until 1994 and afterwards the subject has been mostly ignored. Recently the subject was again brought to discussion due to the possibility of using neural networks in nano-electronic systems where fault tolerance and graceful degradation properties would be very important. In spite of these two periods of work there is still need for a large discussion around the fault model for artificial neural networks that should be used. One of the most used models is based on the stuck at model but applied to the weights. This model does not cover all possible faults and a more general model should be found. The present paper proposes a model for the faults in hardware implementations of feedforward neural networks that is independent of the implementation chosen and covers more faults than all the models proposed before in the literature.

1 Introduction

Fault tolerance in Neural Networks (NNs) has been a topic almost forgotten in the last decade. Only a few papers concerning this topic have been published after 1994: (Eickhoff et al., 2005) (Tchernev et al., 2005) (Arad et al., 1997) (Cavalieri et al., 1999) (Phatak, 1995). In spite of that, the intrinsic fault tolerance of Neural Networks is a very important characteristic. As stated in (Protzel et al., 1993): “The characteristic of a graceful performance degradation without additional redundancy is specially interesting for applications such as long-term, unmanned space missions, where component failures have to be expected but no repair or maintenance can be provided”. Recently the utility of NNs’ built in fault tolerance was also pointed out within nano-electronic systems (Eickhoff et al., 2005). Beyond these areas, situations where hardware acts inside the human body can make use of both fault tolerance and graceful

degradation. It is suitable that in the presence of a malfunction such hardware still performs, at least, part of its functions instead of showing complete failure.
To study fault tolerance and graceful degradation in a general way a global model for the faults is needed. Several authors have addressed this question and proposed solutions but, as will be shown here, the solutions proposed have all left uncovered some important situations.
This paper proposes the fault model that the authors believe to be more accurate and general (regardless of the implementation type used in the hardware).
The main objective of the paper is to enable the discussion of the model proposed which will be the base for future work.

2 Possible Faults in a Neural Network

From a functional point of view and independently of the hardware implementation options, a NN can have the following faults:

1. Fault in a connection/weight or multiplier.
2. Fault in an input.
3. Fault in a multiplier, adder or accumulator.
4. Fault in the activation function.

The fault definition presented here is quite similar to the one proposed in (Piuri et al., 1991). Some of these faults can mask each other in the sense that it can be undistinguishable which fault occurred. This is the case for faults of type 3 and 4, which produce the same type of impact in the output and can be considered as the same situation. While other faults occur in parts of the network which have connections in parallel and the impact in the output must be analyzed, in this part of the circuit the connection is serial and the effect is similar.
Faults of type 1 and 3 both include the multiplier. This duplication was considered because the existence of a multiplier associated with each connection falls into fault type 1 and the existence of a single multiplier before the activation function falls in to fault type 3.

3 Previous Work and Limitations

Most of the previous work was done regarding NNs with binary output. The motivation for this is easy to understand since fault tolerance is much more likely to occur in this type of networks because the fault's impact in the output might be covered by the fact that the output is binary. In spite of that the present work looks for a general model that can be applied to every NN without feedback.
In (Chiu et al., 1993) the fault model used is based on modifications of the weight value of the form $W(1+\alpha)$ where α is allowed to take values in the interval $]-1,1[$. The reason to use this model is clear: it is less restrictive than most of the other solutions but it does not cover most of the possible situations. The weight itself is therefore allowed to change from 0 to 2W. Just to show an example of the limitations of this model, if a digital implementation is considered, a single fault affecting the sign bit would not be covered by this model.

In (Piuri et al., 1991), although the analysis of the fault possibilities is very similar to the one proposed in section II, after a mathematical evaluation the conclusion reached is that from the behavioral point of view all the faults can be summarized as an error in the output of the neuron. This analysis is not correct since it does not hold for an error in the input because this error would affect not one but several neurons.
The paper (Tchernev et al., 2005) considers two types of faults: in the links or connections and in the hidden units. For the first case the faults are modeled as stuck at +MAX, 0 or –MAX, where MAX is the largest weight value in the network. For the second situation the faults are stuck at +Output_Max or -Output_Max. This model is more complete than the previous ones but the choice made for the hidden units is related to the activation functions used and is not general. The model also does not cover a fault in the inputs.
In (Phatak, 1995) the fault model used consists of stuck at +W,-W and 0, where W is the maximum of {maximum magnitude among all the parameter values developed during learning; value needed to drive a unit into saturation}. For the bias, only +W and –W faults are considered. This model covers most of the situations but again it does not cover a fault in the input.
Several other authors used a simpler model with only a stuck at 0 possibility as pointed out in (Phatak, 1995). This model only covers the possibility of a missing link and ignores all other types of faults. Because of this, these models are not considered as a valid option. Their reference is not included here for simplicity's sake.

4 Fault Model Proposed

For easier perception of the following discussion figure 1 shows a representation of a NN of Multi Input Multi Output (MIMO) type.
Starting with a simple model similar to the one used in (Tchernev et al., 2005) of stuck at +WMAX, -WMAX or 0 for all connections it is possible to verify what else is necessary to model all fault possibilities. This analysis will be done considering a Multi Input Multi Output (MIMO) NN with one single fault at a time and for the sake of simplicity.
Looking at the list of possible faults in section II it is easy to verify that this model covers faults of type 1.
Faults of type 2, in one of the inputs would affect more than a single connection. If input 2 is taken as an example, it will affect the links associated with the weights w21 and w22. This would be better modeled considering the range of the affected input and analyzing the impact of stuck at faults at +IMAXN, - IMAXN and 0, where IMAXN is the maximum value that the input N can assume with practical effect (for the hardware implementations there is always a limit in the maximum input due to the DACs, ADCs and power supply limits). Here each input can have a different range and it is important for the evaluation of the fault impact that the different ranges are considered.

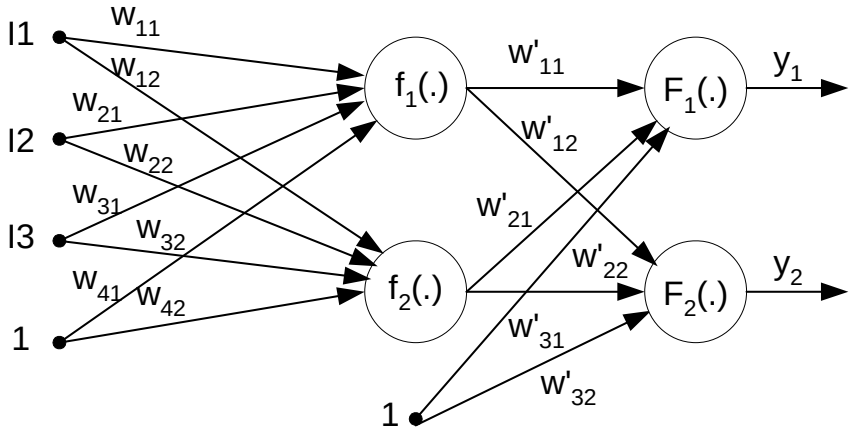

Fig. 1. Representation of a simple feedforward NN of MIMO type.

Faults of type 3 and 4 will be analyzed together because of the similar effects they produce. The proposal here is to model these faults as faults in the weights connecting the outputs of the corresponding neurons to the next layer. This does not hold when dealing with the output neuron and contains also an approximation since the output of the neuron does not have the same range of the weights. In the first situation none of the models presented in the previous section covers an error at the output neuron and such an error is so critical that the output will have no relation with the input in most of the situations. For the second situation if the hidden layer's activation function has a limited output (which is true for a large part of the applications) the fault considered does cover appropriately the situation. Otherwise the fault will be under evaluated.

The global model to be used consists of faults of type stuck at +WMAX, -WMAX or 0 for all the connections and faults of type stuck at +IMAXN, -IMAXN or 0 for all the inputs, where WMAX is the maximum value that the weight can assume with practical effect and IMAXN is the maximum value that the input N can assume with practical effect. This model must be used in the following way:

i) Test the effect of each individual fault by setting the weight associated with each connection to +WMAX, -WMAX and 0.

ii) For each input test the effect of a fault by setting the input to +IMAXN, -IMAXN and 0.

iii) For each neuron in the hidden layer(s) test the effect of a fault by setting all the weights deriving from the output of this neuron to +WMAX, -WMAX and 0. This test is not needed in the case of MISO networks of only one hidden layer since then only one weight connects the output of each neuron.

This model covers more fault situations than all the models present in the literature. However the model does not cover all possible situations. One type of fault that is not covered by the model introduced here is the possibility that a connection is displaced from neuron i to neuron j. Also a fault at the output of a neuron in the output layer is not covered and faults at the output of neurons in the hidden layer with unbounded outputs may be under evaluated. Nevertheless, since the faults are evaluated at ex-

treme situations (worst case) the model can be considered as covering with a reasonable accuracy all possible single faults.
The possibility of setting a weight to 0 to model a fault is included here more because of its physical significance (corresponds to a missing link) than because of its effect since setting the weights to +WMAX or -WMAX has a stronger impact on the output.

5 Conclusion

The strict definition of fault tolerance does not apply to most NNs since they do not give error free results at any time, merely an approximation (Bolt, 1991) and appropriate models are required for accessing the effective fault tolerance of a network.
In the definition of the fault model it might be acceptable to test only the most common parts or the ones which will be responsible for the largest area (Bolt, 1991) in silicium. Applying this principle to NNs, due to the large number of connections, would point to testing only the weights as an acceptable solution for a large number of applications. In the present work the choice was to try to cover all possible situations and the analysis of each possible fault led to a different contribution to the final model.
The model presented here covers more situations than the models previously discussed in the literature. These models all failed to evaluate the effect of a fault in the inputs and under evaluate the faults in the neurons in the hidden layer. Although the present model has some limitations (considers only single faults, faults in neurons in the hidden layer with unbounded outputs may be under evaluated, faults at the output neuron are not considered) it is more accurate than the previous ones.
The existence of global models for faults is of extreme importance to permit evaluation and comparison of different networks, training algorithms and special modification algorithms regarding fault tolerance of NNs.
One limitation of the present work, which also occurs in the previous reported papers, is that only setting the weights to zero, maximum value and minimum value is considered. Since the NN main output does not need to be a monotonic or a linear function of the weights and inputs, other combinations may occur which produce lower fault tolerance than the maximum one reported with this model.

References

1. B. S. Arad and A. El-Amawy, “On Fault Tolerance Training of Feedforward Neural Networks”, Neural Networks, Vol. 10/3, pp.539-553, 1997.
2. G. Bolt, “Investigating the Fault Tolerance in Artificial Neural Networks”, Technical Report YCS 154 from the University of York, available at the internet, 1991.
3. S. Cavalieri and O. Mirabella, “A novel learning algorithm which improves the partial fault tolerance of multiplayer neural networks”, Neural Networks, Vol.12, pp. 91-106, 1999.
4. C. Chiu, K. Mehrotra, C. Mohan and S. Ranka, “Robustness of Feedforward Neural Networks”, 2nd IEEE International Conf. on Neural Networks, Vol. 2, pp. 783-788, 1993.
5. R. Eickhoff and U. Rückert, “Tolerance of Radial Basis Functions Against Stuck-at-Faults”, International Conference of Artificial Neural Networks, pp. 1003-1008, Poland, 2005.

6. H. Elsimary, S. Mashali and S. Shaheen, “Generalization Ability of Fault Tolerant Feed Forward Neural Networks”, International Conference on Systems, Man and Cybernetics, Vol. 1, pp. 30-34, 1995.
7. D. S. Phatak, “Complete and Partial Fault Tolerance of Feedforward Neural Nets”, IEEE Transactions on Neural Networks, Vol. 6/2, pp. 446-456, 1995.
8. V. Piuri, M. Sami, R. Stefanelli, “Fault Tolerance in Neural Networks: Theoretical Analysis and Simulation Results”, 'Advanced Computer Technology, Reliable Systems and Applications', 5th Annual European Computer Conference, Bologna, Italy, 1991.
9. P. W. Protzel, D. L. Palumbo and M. K. Arras, “Performance and Fault-Tolerance of Neural Networks for Optimization”, IEEE Transactions on Neural Networks, Vol. 4, July 1993.
10. E. B. Tchernev and D. S. Phatak, “Investigating the Fault Tolerance of Neural Networks, NCA, 2005