Preliminary Experiments on Fault-Tolerance of a Small Convolutional Neural Network

Haruhiko Kaneko
School of Computing, Tokyo Institute of Technology, Japan

***Abstract*—Fault-tolerance is an important property of safety critical systems, and there exist many established fault-tolerant technologies for conventional computing devices. As preliminary for fault-tolerant design of complex neural networks, this paper presents some simulation results of fault-tolerance of a small convolutional neural network for digit classification, and we show relations between error probability of synaptic weights and classification accuracy.**

I. Introduction

Dependability is a fundamental requirement for many electronic systems. High dependability of recent electronic systems is achieved based on many fault-tolerance technologies of integrated circuits, such as modelings of logical and timing faults, test pattern generation for fault detection, built-in self test, fault simulation, space and time redundancies, and error detection/correction coding [1]. Electronic systems using complex neural networks, such as deep neural network (DNN) [2], will be deployed in many safety-critical systems, and thus dependability of such electronic systems will be crucial. It is expected that neural networks have high tolerance to faults because of its redundant structure [3], and some experiments on phoneme recognition algorithm showed that DNN has high tolerance to faults [4]. Dependability analysis of complex neural networks, however, is not necessarily sufficient compared to conventional electronic systems.

Fault, error, and failure are fundamental terminologies for fault-tolerant computing, and their relations are described as follows: fault is physical defect of system component, such as short circuit, which causes incorrect output of the component; error indicates the incorrect signal value caused by the fault; and failure is incorrect system behavior observable from users caused by errors. Errors may not cause failure immediately, and such unobservable errors are called *latent error*. Figure 1 illustrates the above relations. This paper considers some relations between error and failure of convolutional neural network (CNN) for digit recognition algorithm. We do not address the relation between fault and error because it highly depends on implementation, and its analysis is left for future work.

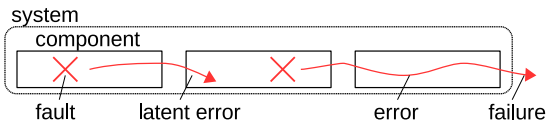


Fig. 1. Fault / error / failure.

II. Evaluated classification algorithm

We consider a simple CNN for handwritten digits classification [5] having three convolution layers, C_1, C_2, and C_3, and a fully connected (FC) layer, as well as multiple normalization, nonlinear activation (ReLU), and max-pooling layers, as depicted in Fig. 2. Input of the convolution is $3\times3\times N_c$ array $[u_{i,j,k}]_{3\times3\times N_c}$, and its output is $v_l = \sum_{i,j,k} w_{i,j,k,l} u_{i,j,k} + b_l$ for $l \in \{0, \ldots, N'_c - 1\}$, where N_c and N'_c are the numbers of input and output channels, $w_{i,j,k,l}$ is weight, and b_l is bias. The values of (N_c, N'_c) are $(1, 16), (16, 32)$, and $(32, 64)$ for C_1, C_2, and C_3, respectively. The nonlinear activation function $f(x)$ is defined as $f(x) = x$ for $x \geq 0$ and $f(x) = 0$ for $x < 0$. The max-pooling operation downsamples the input $[u_{i,j}]_{2\times2}$ by a factor of 2×2 as $v = \max_{i,j} u_{i,j}$. The fully connected layer has 3136 inputs and 10 outputs each corresponds to a digit.

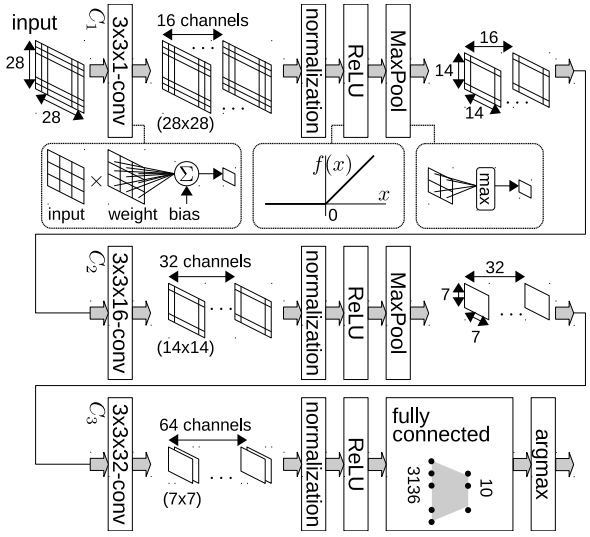
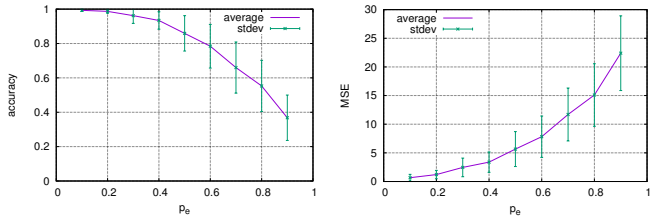


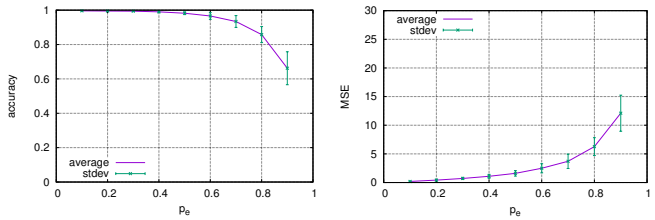
Fig. 2. Digit classification algorithm [5].

III. Error model

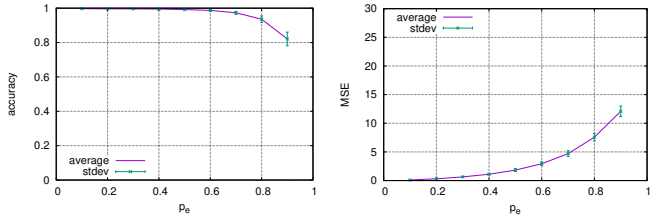
This paper considers fundamental error models found from [3], that is, we assume the following two types of errors in the weights, $w_{i,j,k,l}$, of convolution: (1) stuck-at-0 error in which $w_{i,j,k,l}$ is fixed to zero, and (2) additive error in which the weight is altered to $w_{i,j,k,l} \pm e$, where e is a constant. The following three types of error distributions are assumed: (1) stuck-at-0 error occurs with probability p_e in weight $w_{i,j,k,l}$; (2) additive error occurs with probability p_e in weight $w_{i,j,k,l}$; and (3) stuck-at-0 error occurs with probability p_e in the neuron output.



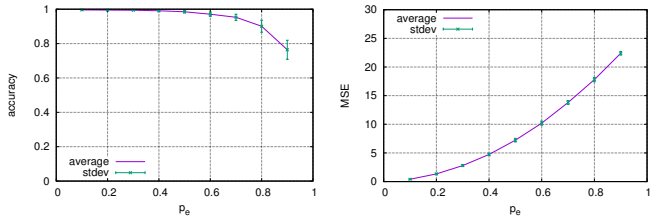
(a) accuracy (errors in C_1). (b) MSE (errors in C_1).



(c) accuracy (errors in C_2). (d) MSE (errors in C_2).



(e) accuracy (errors in C_3). (f) MSE (errors in C_3).



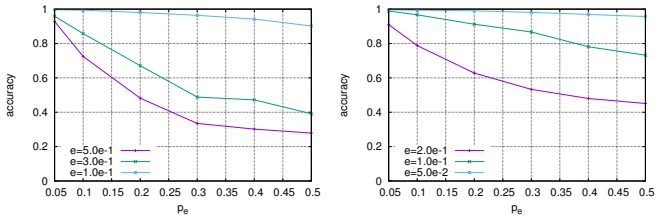
(g) accuracy (errors in FC). (h) MSE (errors in FC).

Fig. 3. Accuracy and MSE of CNN with stuck-at-0 faults in $w_{i,j,k,l}$.

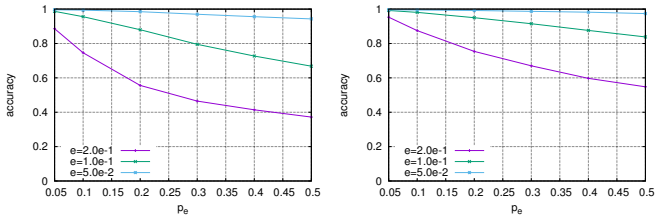
IV. Simulation Results

The CNN is trained using 750 samples without imposing errors in the weights, and then 250 samples are classified by the trained CNN corrupted by errors. The following values are evaluated for the CNN with errors: (1) accuracy defined as the ratio of correct classification to the total number of inputs; and (2) mean squared error (MSE) in the output of FC layer. The simulation is performed 100 times for each error parameter.

Figure 3 shows the average accuracy and MSE when stuck-at-0 errors occur in the synaptic weights of layers C_1, C_2, C_3, and FC, where the error bars indicate the standard deviation. It is observed that the CNN has high tolerance against weight errors, and also it gracefully degrades for around $p_e \geq 0.7$. Figure 4 shows the average accuracy when additive errors of value e occur in the weights of C_1, C_2, C_3, and FC, where the standard deviations of weights in layers C_1, C_2, C_3, and FC are 9.61×10^{-2}, 2.40×10^{-2}, 1.65×10^{-2}, and 2.01×10^{-2}, respectively. Figure 5 shows the accuracy for stuck-at-0 errors in the neuron output. Degradation of the accuracy is more significant compared to Fig. 3 where errors exist in neuron inputs.

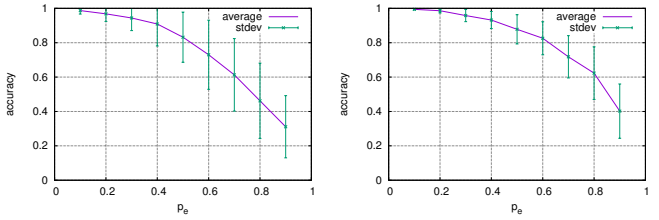


(a) errors in C_1. (b) errors in C_2.

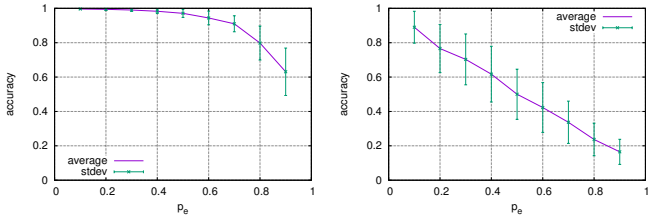


(c) errors in C_3. (d) errors in FC.

Fig. 4. Accuracy of CNN with additive errors in $w_{i,j,k,l}$.



(a) errors in C_1. (b) errors in C_2.



(c) errors in C_3. (d) errors in FC.

Fig. 5. Accuracy of CNN with stuck-at-0 error in neuron output.

V. Conclusion

This paper presented simulation results of fault tolerance of a small CNN. It was shown that the CNN has high fault tolerance especially for stuck-at-0 faults, while large additive errors considerably affects the accuracy. Further failure analysis will be required for, e.g., large network, errors in other than synaptic weights, and errors in learning phase. Also, fault-tolerant structures of complex neural network need to be investigated.

References

[1] E. Fujiwara, *Code Design for Dependable Systems,* Wiley, 2006.

[2] G. E. Hinton, S. Osindero and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation,* Vol. 18, No. 7, pp.1527–1554, 2006.

[3] C. Torres-Huitzil and B. Girau, "Fault and Error Tolerance in Neural Networks: A Review," *IEEE Access,* Vol. 5, pp. 17322–17341, 2017.

[4] M. Lee, K. Hwang, and W. Sung, "Fault Tolerance Analysis of Digital Feed-Forward Deep Neural Networks," *IEEE Int. Conf. Acoustic, Speech and Signal Processing,* pp. 5031–5035, 2014.

[5] https://jp.mathworks.com/help/nnet/examples/create-simple-deep-learning-network-for-classification.html?lang=en