FAULT TOLERANCE ANALYSIS OF DIGITAL FEED-FORWARD DEEP NEURAL NETWORKS

Minjae Lee, Kyuyeon Hwang, and Wonyong Sung

Department of Electrical and Computer Engineering, Seoul National University
Gwanak-gu, Seoul 151-744 Korea
Email: mjlee@dsp.snu.ac.kr, khwang@dsp.snu.ac.kr, wysung@snu.ac.kr

ABSTRACT

As the homeostatis characteristics of nerve systems show, artificial neural networks are considered to be robust to variation of circuit components and interconnection faults. However, the tolerance of neural networks depends on many factors, such as the fault model, the network size, and the training method. In this study, we analyze the fault tolerance of fixed-point feed-forward deep neural networks for the implementation in CMOS digital VLSI. The circuit errors caused by the interconnection as well as the processing units are considered. In addition to the conventional and dropout training methods, we develop a new technique that randomly disconnects weights during the training to increase the error resiliency. Feed-forward deep neural networks for phoneme recognition are employed for the experiments.

Index Terms— dropout training, fault model, fault tolerant characteristic, neural network hardware

1. INTRODUCTION

Feed-forward deep neural networks (DNNs) that employ multiple hidden layers show high performance in various applications [1–6]. Although DNNs demand high complexity circuit, their implementation using nano-scale semiconductor technology is quite promising. However, nano-scale systems frequently suffer from not only hard circuit faults but also soft operational errors due to insufficient timing margin or operating voltage. There are several researches to overcome the reliability problem of nano-scale systems, such as redundant arithmetic and stochastic computing [7, 8].

There have been many studies on fault tolerant characteristics of artificial neural networks. The fault tolerance of neural networks was studied when the number of hidden layers and that of units in each layer are varied in the retraining process in [9]. The study of spiking neural circuits with improved redundancy was conducted in [10]. The effect of training method on fault tolerance of an artificial neural network was studied in [11, 12]. The previous studies were mostly based on shallow neural networks with a small number of neurons in the hidden layers. Since neural networks with deeper architecture shows better performance, it is necessary to examine fault tolerant characteristics of DNNs.

In this study, we examine the fault tolerance of digital VLSI-based feed-forward deep neural networks. Digital VLSI-based DNN circuits have very regular structures mainly composed of memory and processing units. For this study, we design fixed-point digital deep neural networks for speech phoneme recognition. Two kinds of fault are assumed, one in the memory and the other in the processing units. Both stuck to zero and random errors are considered. The fault tolerance of the network when the number of units in each layer varies is studied. Also, the fault tolerant effect according to the training technique is also measured. We develop a weight dropout training technique to strengthen the fault-tolerance of the network against weight errors.

This paper is organized as follows. In Section 2, the fixed-point deep neural networks for phoneme recognition are developed. Section 3 describes the fault model employed in this study. The experimental results are shown in Section 4, followed by concluding remarks in Section 5.

2. FIXED-POINT DNN DESIGN

In this study, we use the DNN-based phoneme recognition algorithm that consists of an input layer, four hidden layers, and one output layer. The input layer contains 429 linear units to accept real valued inputs that correspond to 11 frames of MFCC (Mel-frequency cepstral coefficient) parameters. Four hidden layers have the same number of logistic units, which is 256, 512, or 1024 in this study. The output layer consists of 61 logistic units that correspond to 61 target phoneme labels. The similar structure can be found in [4].

The networks are pre-trained with unsupervised greedy restricted Boltzmann machine (RBM) learning. Training parameters that critically affect the performance are care-

This work was supported in part by the Brain Korea 21 Plus Project grant funded by the Ministry of Education, Science and Technology (MEST), Republic of Korea, and in part by the National Research Foundation of Korea (NRF) grants funded by the Korean government (MEST) (No. 2012R1A2A2A06047297)

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

Table 1. Frame level phoneme recognition error rate (%) of 1024-unit-layer network, which is trained conventionally using M-point weight quantization.

Approach	Signal word-length	M = 3	M = 7	M = 15
Floating point	–		26.24	
Fixed point (direct)	1 bit	66.58	46.53	38.34
	2 bits	56.15	34.56	30.44
	3 bits	54.10	33.36	28.85
	8 bits	50.20	32.85	28.55
Fixed point (retrain)	1 bit	29.97	29.76	29.67
	2 bits	28.35	28.46	28.02
	3 bits	27.63	27.90	27.73
	8 bits	27.37	27.87	27.84

fully selected by experiments. The binary-Gaussian RBM is trained by 40 epochs with the learning rate of 0.005. For the other RBM, we use 20 epochs of 1-step contrastive-divergence based stochastic gradient descent with the mini-batch size of 128, the learning rate of 0.05, and the momentum of 0.9. For the fine-tuning, we use 10 epochs of the back-propagation with the stochastic gradient descent, the mini-batch size of 128, the fixed learning rate of 0.05, and the momentum of 0.9.

The conventional training does not employ any regularization technique such as dropout during the back-propagation [18]. For this training, we use the TIMIT corpus that is comprised of a training set from 462 speakers and a test set from 168 speakers [13]. All SA recordings, utterances of the same sentences from every speaker, in the corpus are removed during training since it can give bias to the results. The input receives 39 dimension MFCCs, which are 12th-order MFCCs with energy and their first and second temporal derivatives. MFCCs are extracted using the 25-ms Hamming window with the 10-ms frame rate. We use 11 consecutive frames that are normalized to have zero mean and unit variance [4]. The evaluation uses 39 phone classes which are mapped from the original 61 phones as described in [14].

For VLSI-based implementation of a DNN, fixed-point arithmetic [15] is much desired. However, direct quantization of floating-point weights for obtaining fixed-point weights does not yield good results when the precision of weights is very low. To address this issue, we employ a fixed-point optimization scheme that retrains the directly quantized neural network. [16, 17]. The input signal is quantized with fixed 8-bit word-length, where the output is not quantized. In digital VLSI based neural networks, reducing the word-length of the weights is very important for hardware cost reduction because the number of them usually exceeds millions. We only use 3 levels (+1, -1, and 0) for representing the weights and 3 bits for internal signals. Initial fixed-point weights are obtained by directly quantizing the optimum floating-point weights. The quantization step size is determined using L2 minimum optimization followed by exhaustive search.

Table 2. Frame level phoneme recognition error rate (%) according to the network size with floating-point and fixed-point arithmetic.

Hidden layer size and training method	Error rate (%)	
	Floating-point	Fixed-point
256-unit-layer with conv. training	28.19	32.53
512-unit-layer with conv. training	26.84	28.91
1024-unit-layer with conv. training	26.24	27.63
1024-unit-layer with unit dropout	23.71	24.93
1024-unit-layer with weight dropout	25.87	28.03

In order to further refine the fixed-point weights, the back-propagation-based retraining algorithm is reapplied. In the retraining procedure, we maintain both the high- and low-precision weights and the signals to accumulate the effects of small adaptation error. Table 1 shows that the retraining based fixed-point optimization scheme results in good performance even when the network employs only ternary weights and 3-bit signals for the hidden layers. This indicates that two bits are sufficient to represent a single weight.

Recently, a new regularization algorithm that intentionally drops out some of the processing units in the network to prevent early over-fitting has been developed [18]. In this algorithm, called dropout, some randomly chosen processing units are forced to have zero output. This algorithm shows excellent performance with floating-point arithmetic. We found that this training algorithm also yields better recognition performance when applied to fixed-point DNN optimization. The performance of dropout when applied to phoneme recognition is shown in Table 2.

In this study, we develop a modified form of dropout training method that randomly drops out (forcing to zero) about 30% of the weights. The performance of this training technique is shown in Table 2. The developed weight dropout training does not show better results when compared to the unit dropout method. However, the weight-error tolerance of the DNN with the weight dropout training is worth studying.

3. FAULT MODEL OF DNN

A fixed-point deep neural network mainly consists of two components; one is the interconnection with weights of +1, -1, and 0, and the other is the processing units that include adders and other logic components. Therefore, we study the performance of both cases: one with the fault in the interconnections and the other in the processing units.

3.1. Interconnection fault model

A DNN contains many connections, and the weight of each connection can be simplified to +1, -1, or 0 by the fixed-point optimization. The interconnection networks can be implemented in two ways; one is using dedicated wires and con-

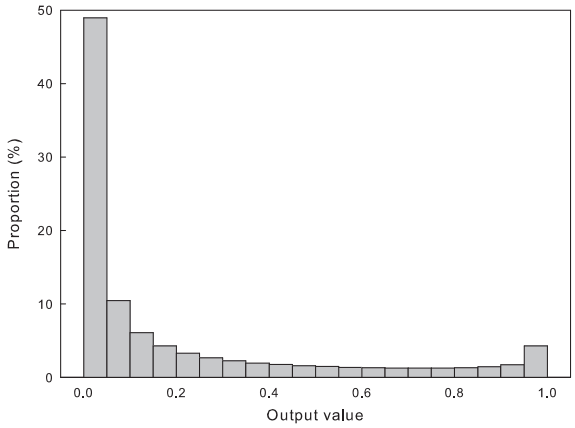


Fig. 1. Output value distribution of processing units.

tacts, while the other is employing CMOS switches whose control is determined by weights stored in memory.

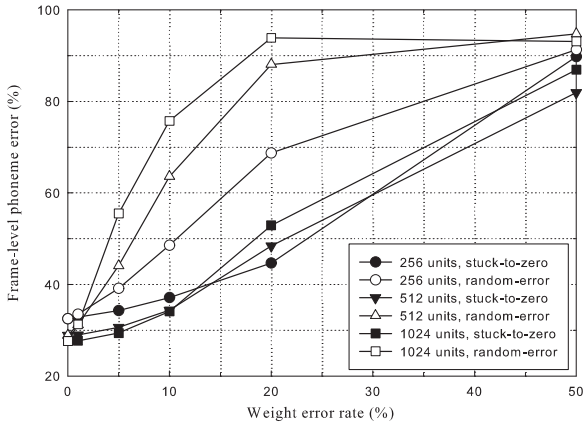
We consider two types of interconnection faults. One is the uni-directional faults and the other is the random faults. In the former case, interconnections whose weights are either +1 or -1 can be disconnected, but disconnected ones are not connected by faults. When the weights are stored in the memory, the memory contents of +1 or -1 can be changed to 0, but those of 0 are not subject to change. When weights are stored in the flash memory, where the electric charge is injected for programming, the leakage of stored charge incurs only one directional data error. On the other hand, the random fault model assumes that the interconnection weights of +1, -1, or 0 can be changed to any other values by faults. This is the case when the weights are stored in CMOS SRAM that does not have any polarity in the direction of faults.

3.2. Processing unit fault model

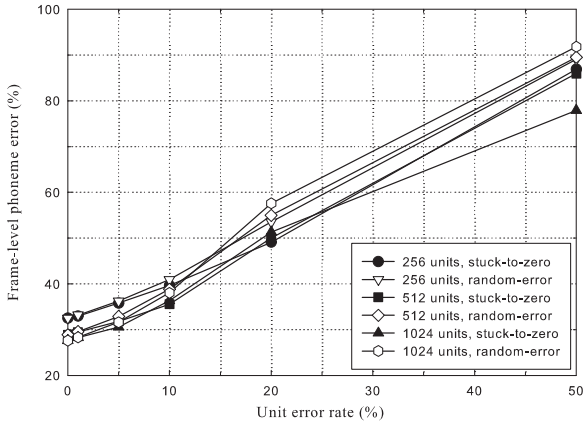
Each processing unit in digital CMOS-based VLSI of a DNN usually consists of adders, registers, and a logistic sigmoid function unit. The number of adders employed in each processing unit depends on the time-multiplexing factor. We consider two fault models depending on the output value of a faulty processing unit; one is always 0 and the other one is randomly determined between 0 and 1. We profiled the output value of all the processing units in the DNN for phoneme recognition and found that the majority output value is close to 0 as shown in Figure 1. Thus, it can be advantageous to make the output value of a faulty processing unit zero.

4. EXPERIMENTAL RESULTS

The fault tolerance of DNNs depends on several factors. We show the tolerance of each fault model with various sizes of the network. Fault tolerant characteristics of DNNs trained with ordinary training method and DNNs with dropout training are compared.



(a) Weight-error tolerance of DNNs having different network size.



(b) Unit-error tolerance of DNNs having different network size.

Fig. 2. Frame-level phoneme error rate of conventional DNN.

4.1. Weight-error tolerance

Figure 2(a) shows the frame-level phoneme error rate of two weight-error models. The results clearly show that the stuck-to-zero fault model for the weights yields much better performance than the random error model. For the random-error model, the network with an increased number of units per layer is more severely affected by the error. This shows that error tolerance of DNNs can be improved by forcing random weights errors to zero. Also, we can notice that the network degrades very modestly with the stuck-to-zero fault model when the weight-error rate is less than 5%.

4.2. Unit-error tolerance

The unit-error in the neural networks is similar to a faulty neuron itself. The phoneme error rate respected to the unit-

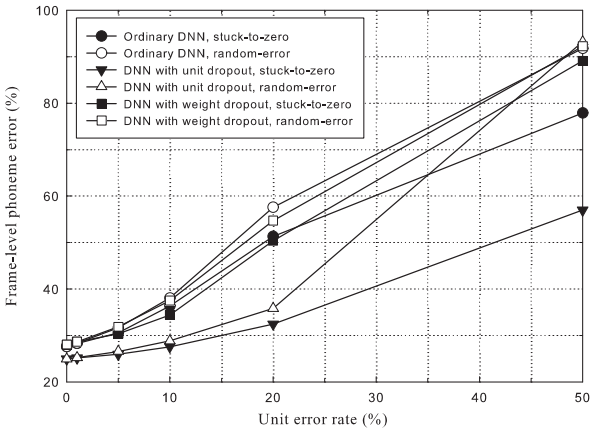


Fig. 3. Unit-error tolerance of DNNs trained by different methods.

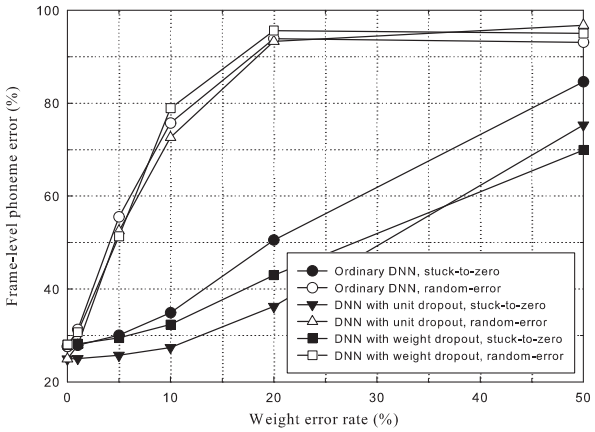


Fig. 4. Weight-error tolerance of DNNs trained by different methods.

error rate is shown in Figure 2(b). Here, we can find that the effect of the fault model is not significant. Although the stuck-to-zero model shows better recognition performance when the error rate is very high, there is not much difference when the error rate is less than 5%. Varying the network size also does not show consistent results. When the unit size in each layer is 1024, the network degrades very gracefully until the unit-error rate of 5%.

4.3. Processing unit error tolerance of DNNs with different training and fault models

Dropout was developed for the purpose of overcoming the over-fitting in the training process [19]. In the training with dropout, some outputs of the processing units are randomly forced to zero. In order words, processing unit faults are randomly injected during the back-propagation-based supervised training. This technique was developed to overcome the over-fitting problem and obtains better results with a limited size of training data. Figure 3 compares the phoneme recognition rates with 1024 units per hidden layer. We can find that DNNs trained with dropout are more tolerant than the ordinary DNNs. The DNNs trained with weight dropout does not show better results than the DNN with the (processing unit) dropout training.

4.4. Weight error tolerance of DNNs with different training and fault models

The weight-error tolerance of DNNs with different training methods and fault models are shown in Figure 4. The number of units per hidden layer is 1024. It is clear that the fault model is the most Significant factor. The DNNs with the stuck-to-zero model are always better than those with the random error model. We can find that the error resiliency of the DNN trained with weight dropout is the best when the error rate is very high, around 50%. Note that the (processing unit) dropout training yields the best results when the error rate is small. However, the DNN with the weight dropout training shows better results when the weight-error rate exceeds 40%. This shows that tolerance of DNN is also dependent on the characteristics of faults injected during the training process, where the fault injection is originally intended to prevent over-fitting.

5. CONCLUDING REMARKS

We examine the fault tolerance of digital feed-forward deep neural networks (DNNs) when altering the fault model, network size, and training method. Both interconnection and processing unit errors are considered. We also compare the effects of training methods that include conventional, unit dropout, and weight dropout techniques. The stuck-to-zero fault model yields higher tolerance when compared to the random error model. The most influential factor is the training method. The unit dropout training prominently increases the fault tolerance of the network against unit errors, while the proposed weight dropout training also modestly strengthens the fault tolerance against severe weight errors. We can find that DNNs with appropriate training methods experience only modest performance loss when the error rate is under 10%. This study contributes to the implementation of robust deep neural networks employing nano-scale digital CMOS technology.

6. REFERENCES

[1] 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.

[2] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[3] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[4] A. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 20, no. 1, pp. 14–22, 2012.

[5] A. Mohamed, T. N. Sainath, G. E. Dahl, B. Ramabhadran, G. E. Hinton, and M. A. Picheny, "Deep belief networks using discriminative features for phone recognition," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2011, pp. 5060–5063.

[6] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[7] N. R. Shanbhag, R. A. Abdallah, R. Kumar, and D. L. Jones, "Stochastic computation," in *ACM Proceedings of the 47th Design Automation Conference*, 2010, pp. 859–864.

[8] Y. N. Chang and K. K. Parhi, "Architectures for digital filters using stochastic computing," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 2697–2701.

[9] M. Vural, A. Ozgur, A. Schmid, and Y. Leblebici, "Fault tolerance of feed-forward artificial neural network architectures targeting nano-scale implementations," in *50th IEEE Midwest Symposium on Circuits and Systems*, 2007, pp. 779–782.

[10] N. Joye, A. Schmid, Y. Leblebici, T. Asai, and Y. Amemiya, "Fault-tolerant logic gates using neuromorphic cmos circuits," in *Proceedings of the 2007 Ph. D. Research in Microelectronics and Electronics Conference*. IEEE, 2007, pp. 249–252.

[11] W. S. Hsieh and B. Y. Sher, "Fault tolerant capability of multi-layer perceptron neural network," in *Proceedings of the 20th EUROMICRO Conference. System Architecture and Integration*, 1994, pp. 644–650.

[12] C. H. Sequin and R. D. Clay, "Fault tolerance in artificial neural networks," in *IJCNN International Joint Conference on Neural Networks*, 1990, vol. 1, pp. 703–708.

[13] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[14] K. F. Lee and H. W. Hon, "Speaker-independent phone recognition using hidden Markov models," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 11, pp. 1641–1648, 1989.

[15] W. Sung and K. -I. Kum, "Simulation-based word-length optimization method for fixed-point digital signal processing systems," *IEEE Transactions on Signal Processing*, vol. 43, no. 12, pp. 3087–3090, 1995.

[16] A. Choudry E. Fiesler and H. J. Caulfield, "Weight discretization paradigm for optical neural networks," *The Hague'90 April. International Society for Optics and Photonics*, pp. 164–173, 1990.

[17] C. Z. Tang and H. K. Kwan, "Multilayer feedforward neural networks with single powers-of-two weights," *IEEE Transactions on Signal Processing*, vol. 41, no. 8, pp. 2724–2727, 1993.

[18] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for lvcsr using rectified linear units and dropout," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 8609–8613.

[19] A. Krizhevsky I. Sutskever G. E. Hinton, N. Srivastava and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," `http://arxiv.org/abs/1207.0580`, 2012.