Article Information

Title	Fault-Tolerant Ensemble CNNs Increasing Diversity Based on Knowledge Distillation
Authors	Shunsuke Koeda, Yoichi Tomioka, Hiroshi Saito
Citation	2023 IEEE 16th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC), Singapore, 2023, pp. 399-405, doi: 10.1109/MCSoC60832.2023.00066.
Copyright	
Note	This is the author's accepted version of the paper published in *Proceedings of IEEE International Symposium on Embedded Multicore/Many-core Systems-on-Chip (2023)*. The final version is available at IEEE Xplore DOI: https://doi.org/10.1109/MCSoC60832.2023.00066

Fault-Tolerant Ensemble CNNs Increasing Diversity Based on Knowledge Distillation

Shunsuke Koeda, Yoichi Tomioka, Hiroshi Saito
The University of Aizu
Aizu-Wakamatsu City, Fukushima, Japan
{m5261125, ytomioka, hiroshis}@u-aizu.ac.jp

***Abstract*—False inference of convolutional neural networks (CNNs) can lead to serious accidents in mission-critical artificial intelligence (AI) applications such as self-driving and medical systems. It is important not only to achieve sufficient accuracy but also to detect hardware faults and continue reliable inferences. Triple Modular Redundancy (TMR) is a conventional fault-tolerant method taking the majority voting of the outputs of three identical modules. However, it increases the computational cost three times, leading to increasing the power consumption and circuit area. In this paper, we propose a computationally low-cost fault-tolerant ensemble model consisting of multiple CNN models and its training approach to improve the accuracy. Each CNN model involves a shared part that has common parameters among all CNN models. It can detect the fault of each hardware module running on each CNN model by comparing the outputs of shared parts and dynamically reconfigure the ensemble model by taking the average of the outputs only for non-faulted CNN models. In the experiments, we demonstrate that the proposed method significantly reduces the computational cost to 43% while achieving comparable accuracy and robustness.**

***Index Terms*—fault-tolerant, ensemble learning, convolutional neural network, knowledge distillation**

I. Introduction

Artificial intelligence (AI) is expected to play an important role in realizing a safer, more convenient, and more intelligent society. For example, it is helpful to improve productivity through smart factories, reduce traffic accidents, enhance public transportation services in underpopulated areas through self-driving technologies, and improve the quality of medical and infrastructure services.

In mission-critical systems that are controlled based on AI recognition, failure of AI circuits due to aging deterioration or other factors may cause serious malfunctions. The defects in the oxide film caused by applied bias and temperature is called Bias Temperature Instability (BTI), which is an example of circuit wear-out. It has been reported that BTI increases the delay of some circuits by approximately 10% in one year and 17% in 10 years [1]. Aged AIs may cause unexpected behavior due to BTI ten years later. Therefore, it is essential not only to improve the accuracy of AI but also to realize fault-tolerant AI that can detect failures and maintain proper output even when AI suddenly fails.

This work was supported by JST, PRESTO Grant Number JPMJPR22C7, Japan.

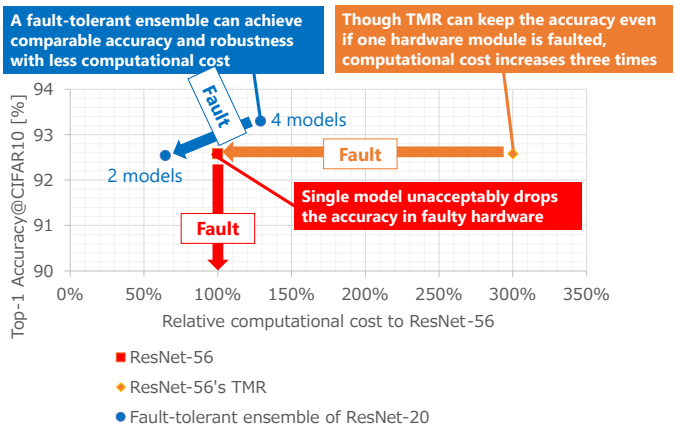


Fig. 1. Relationship between accuracy and computational cost of a single model, TMR, and the proposed fault-tolerant ensemble model.

Convolutional neural network (CNN)-based image recognition is a promising technology for realizing reliable image classification, object detection, and image segmentation. For example, CNNs play important roles in a self-driving system in recognizing pedestrians, cars, traffic signals, road lances, and so on. In this paper, we focus on the realization of fault-tolerant CNN models.

An n-modular redundancy (n-MR), which compares the results of n identical modules, is one of the conventional fault-tolerant techniques. When n = 2, n-MR is called Dual Modular Redundancy (DMR) and can detect a failure if at most one module outputs incorrect values. Also, it is called triple modular redundancy (TMR) when $n = 3$. The TMR can detect a fault and produce correct output by majority voting if only one of the three modules does not work correctly. For example, there is an AI system such as Tesla's Full Self-Driving System [2] supporting DMR. For image recognition AI of mission-critical systems, it is more desirable to maintain correct inference by TMR even in a sudden failure, which triples the computational cost as shown in ResNet-56's TMR of Figure 1. Moreover, it also increases the circuit area, power consumption,

energy consumption, and heat dissipation, which is an obstacle to introducing fault-tolerant AI in edge devices with severe cost, power, and heat dissipation conditions. In order to improve the reliability of edge AI, fault-tolerant technology with low computational load and circuit area overhead is required.

In this paper, we propose a computationally low-cost fault-tolerant ensemble CNN model, which can reduce the computational cost compared with TMR and achieve comparable accuracy as in Figure 1. This model consists of multiple CNN models. Each CNN model involves a shared part that has common parameters among the CNN models. It can detect the fault of each hardware module running on each CNN model by comparing the outputs of shared parts and dynamically reconfigure the ensemble model by taking the average of the outputs only for non-faulted CNN models. Moreover, we propose a training method for the fault-tolerant ensemble CNN model to increase the diversity of CNN models based on knowledge distillation [3] and improve the accuracy of the fault-tolerant ensemble CNN model. We employ Knowledge distillation techniques to promote increasing the diversity of CNN models while maintaining the shared part. In the experiments, we demonstrate that the proposed method significantly reduces the computational cost to 43% while achieving comparable accuracy and robustness and has a high potential to realize low-cost and fault-tolerant CNN models.

The remainder of this paper is organized as follows. In Section II, we explain related work. In Section III, we explain the knowledge distillation proposed in [3]. In Section IV, we explain the proposed fault-tolerant ensemble CNN model and its training approach. In Section V, we show the experimental results. Finally, we conclude this paper in Section VI.

II. Related Work

Reference [4] has proposed an approach that generates a low-bit quantized neural network model and runs it an accelerator based on a multiply-accumulate (MAC) unit with guard bits. This approach realizes more reliable inference which is robust to BTI. However, this method has a large area overhead for MAC units and also cannot realize fault detection and reproduction of correct outputs. Reference [5] has proposed a robust neural network model against a single bias attack. This method applies a DMR approach to only neurons sensitive to the single bias attack and reduces the additional computational cost for attack detection. This approach is also helpful to detect a fault related to bias. However, it cannot detect faults caused by other factors such as BTI. Approximated DMR unit consisting of binary and ternary convolutional layers has been proposed in [6]. The DMR unit consists of the original layer and the approximated layer by random forest model. Although this approach can significantly reduce the circuit area and the computational cost, it cannot directly be applied to more general CNN models with wider bit width. On the other hand, in this paper, we aim to realize a low-cost fault-tolerant method that is applicable to more general CNN models with wider bit width and can detect permanent hardware faults, and can continue sufficiently accurate inference.

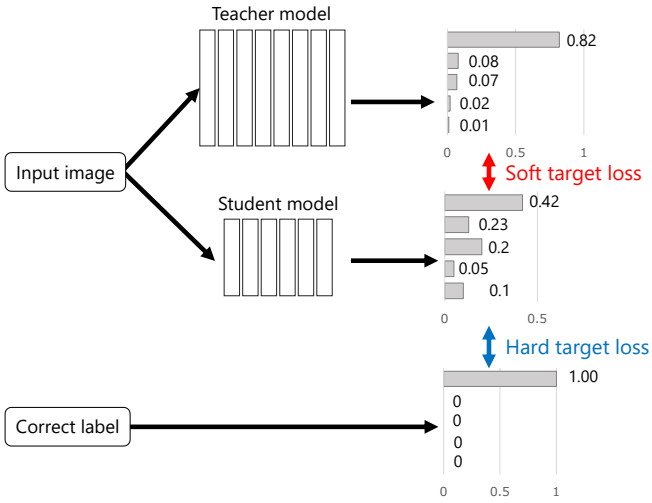


Fig. 2. Outline of knowledge distillation.

III. Knowledge Distillation

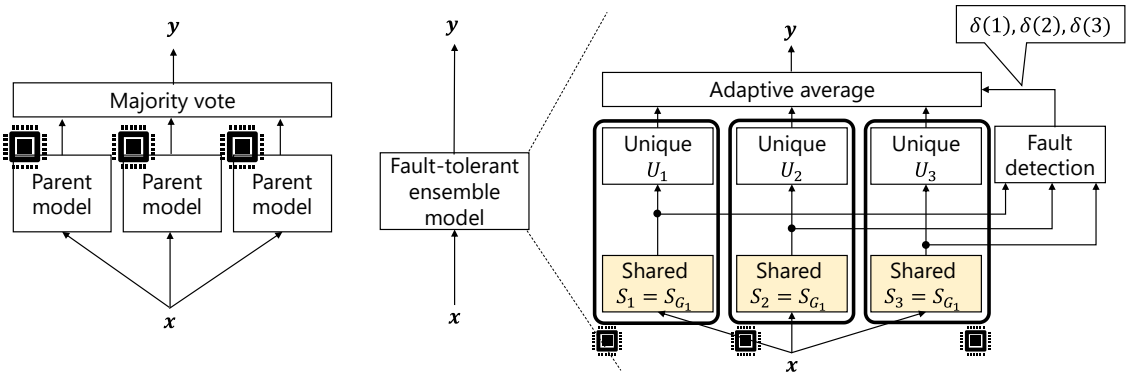
In this section, we explain a knowledge distillation technique that will be used to increase the accuracy of a fault-tolerant ensemble model in the proposed method. Knowledge distillation has been proposed in [3], which is one of the model compression techniques. A CNN model is trained so that the loss function defined with the output of the CNN model and the teacher signals is minimized. Typically, it is more difficult for a smaller model to achieve higher accuracy than a larger model. Knowledge distillation is a technique to improve a small CNN model by transferring knowledge of a larger CNN model.

We show the outline of knowledge distillation in Figure 2. In knowledge distillation, we assume that there are two models: a teacher model and a student model. The teacher model is a large CNN model which is pre-trained with a dataset. We train the student model, which is a smaller CNN model than the teacher model so that the output probability distribution becomes closer to that of the teacher model.

The loss function consists of hard target loss and soft target loss. The hard target loss is defined by a cross-entropy loss $H(\boldsymbol{p}^{(\boldsymbol{s})}, \boldsymbol{y})$ between the probability distribution $\boldsymbol{p}^{(\boldsymbol{s})}$ of the student model and the one-hot teacher signal $\boldsymbol{y}$ as follow.

$$H(\boldsymbol{p}^{(\boldsymbol{s})}, \boldsymbol{y}) = \sum_i y_i \log p_i^{(s)} \tag{1}$$

where $p_i^{(s)}$ is the i-th element of probability distribution $\boldsymbol{p}^{(\boldsymbol{s})}$ and y_i is the i-th element of teacher signal $\boldsymbol{y}$. The



(a) TMR of CNN. (b) Fault-tolerant Ensemble CNN.

Fig. 3. Concept of the proposed fault-tolerant CNN model.

probability distribution of the student model is calculated by inputting the logits $\boldsymbol{z}^{(s)}$ of the student model to the softmax function

$$\sigma(z_i^{(s)}) = \frac{\exp(z_i^{(s)})}{\sum_i \exp(z_i^{(s)})} \tag{2}$$

where $z_i^{(s)}$ is the i-th element of logit vector $\boldsymbol{z}^{(s)}$.

The Kullback–Leibler divergence of the smoothed probability distributions of student and teacher models defines the soft loss function. Let $\boldsymbol{z}^{(x)}$ be the logits of the teacher (t) or student (s) model. The smoothed probability distribution $\boldsymbol{q}^{(x)}$ is calculated from logits $\boldsymbol{z}^{(x)}$ using softmax function with a temperature $\sigma_t(z^{(x)}, T)$, which is proposed in [3].

$$\sigma_t(z_i^{(x)}, T) = \frac{\exp(\frac{z_i^{(x)}}{T})}{\sum_i \exp(\frac{z_i^{(x)}}{T})}. \tag{3}$$

The softmax function with a temperature promotes smoothly transferring a teacher model's knowledge to a student model.

IV. Fault-tolerant Ensemble CNN

A. Fault detection and dynamic reconfiguration of ensemble CNN

In neural network circuits, various types of failures may occur. For example, there is a risk that the electric charge generated by cosmic rays will flow current between normally insulated areas, resulting in permanent failures. In systems that are operated for long periods of time, there is a possible risk of permanent timing violations due to increased transistor delays caused by wear-out of a circuit. In any of these failures, from the standpoint of maintaining the inference function, there is no problem in ignoring the failure if the intermediate and final outputs of the neural network do not change. It is important to know how to detect faults that cause outputs to change and how to maintain the correct outputs. Although error-correcting code (ECC) memory can detect single-bit errors of feature maps and parameters and correct them on the memory, it is not enough to address such permanent faults regarding electrical conductivity. Our approach mainly focuses on fault-tolerant inference against a permanent fault of a circuit that causes the output to change, which can improve the reliability of neural networks when used with ECC memory.

Figure 3(a) shows a conventional TMR approach that replicates the original CNN model called the parent model into three. Each parent model is executed on a different hardware module. We can continue correct inference by taking a majority vote on the outputs of three parent models even when one hardware module is faulted. TMR increases the number of operations by three times and requires three hardware modules to run the CNN model, which increases the system's circuit area and power consumption. The hardware module can be classified into two types. In the first type, the layers of a neural network model are processed in parallel with different circuit modules. This type is suitable for highly-parallel computing though it may spend more hardware resources. In the second type, the same circuit modules are used, and they sequentially process layer by layer. Because of the limitation of hardware resources, many hardware modules that can process neural network inference belong to this type. We also employ hardware modules of the second type. Please note that a permanent fault of such hardware module can cause the output change of each layer. For example, the hardware module could be a graphics processing unit (GPU), a neural network accelerator (NNA), or a central processing unit (CPU).

Modern CNN models such as ResNet [7] and ConvNext [8] consist of multiple stages, and each stage consists of multiple blocks. Although the number of blocks is different in each stage, the combination of layers is the

same. Therefore, if we can confirm that no failure occurs throughout the execution of one stage, we can efficiently detect a permanent hardware failure that would affect the inference of the CNN model. Therefore, we propose a new fault-tolerant ensemble CNN model shown in Figure 3(b).

The fault-tolerant ensemble CNN model is an ensemble model consisting of N small CNN models, called child models. The inference of each child model is performed on a different hardware module. The fault-tolerant ensemble CNN model achieves the same level of accuracy as the parent model using ensemble techniques. On the other hand, because each child model requires less computation than the parent model, each hardware module can also be smaller. Each child model $C_k = (S_k, U_k)$ $(k = 1, 2, \ldots, N)$ consists of a shared part S_k and an unique part U_k, which are connected in series so that the output of S_k is inputted to U_k. That is, $C_k(\boldsymbol{x}) = U_k(S_k(\boldsymbol{x}))$ where $\boldsymbol{x}$ is an input. The output of an ensemble model consisting of $\{C_1, C_2, \ldots, C_N\}$ is defined by

$$\frac{1}{\sum_{k=1}^{N} \delta(k)} \sum_{k=1}^{N} \delta(k) C_k(\boldsymbol{x}).$$

If the hardware module in which the child model C_k is executed is faulty, $\delta(k)$ is set to 0. Otherwise, $\delta(k)$ is set to 1. That is, we calculate the average of the outputs of all non-faulty child models. Initially, we set $\delta(k) = 1$ for all k.

The child models are divided into groups $\boldsymbol{G_i}$ $(i = 1, 2, \ldots, M)$. For each group $\boldsymbol{G_i}$, a shared part $S_{\boldsymbol{G_i}}$ is generated. The shared part S_k $(k \in \boldsymbol{G_i})$ is set to $S_{\boldsymbol{G_i}}$. Therefore, we can detect a fault by comparing the outputs of S_k $(k \in \boldsymbol{G_i})$. In the case that $|\boldsymbol{G_i}| = 2$, we cannot know which child model in $\boldsymbol{G_i}$ is faulted. Therefore, we set $\delta(k) = 0$ for all $k \in \boldsymbol{G_i}$. In the case that $|\boldsymbol{G_i}| > 2$, we can identify a faulted child model C_f. Therefore, we set $\delta(f) = 0$.

Figure 3(b) shows an example of fault-tolerant ensemble model consisting of three child models of one group $\boldsymbol{G_1} = \{1, 2, 3\}$. Moreover, we show an example of a fault-tolerant ensemble model consisting of four child models of two groups $\boldsymbol{G_1} = \{1, 2\}$ and $\boldsymbol{G_2} = \{3, 4\}$ in Figure 4.

B. Training for increasing diversity

We need shared parts to detect a fault in the proposed fault-tolerant ensemble model. However, such shared parts can decrease the diversity of child models and accuracy improvement by ensemble learning. Therefore, we introduce knowledge distillation techniques to increase the diversity of child models.

First, we train N teacher models $T_k = (S_k^T, U_k^T)$ $(k = 1, 2, \ldots, N)$. Though each teacher model has the same structure as child models, S_k^T is not shared with each other. That is, $S_p^T \neq S_q^T$ if $p \neq q$. Each model is trained with the same dataset. However, different teacher models are obtained because different data augmentation is applied due to different random seeds.

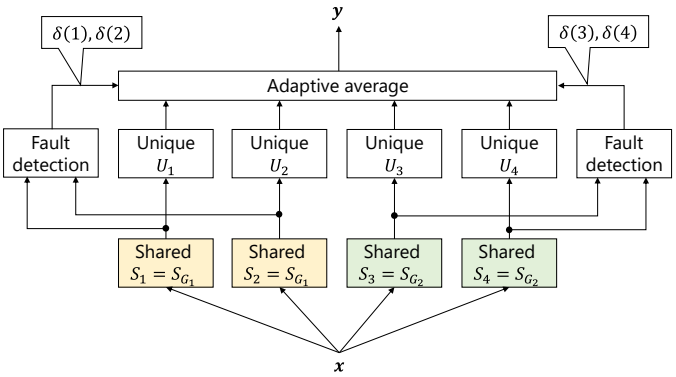


Fig. 4. An example of a fault-tolerant ensemble model consisting of four child models of two groups.

Second, we train N child models using knowledge distillation from the teacher models. Child model C_k is trained with knowledge distillation from teacher model T_k. That promotes generating child CNN models of different characteristics.

V. Experimental results

A. Environment and settings

In the experiments, we used a desktop PC equipped with 3.8GHz Core i7-10700K CPU, 32GB DDR4 memory, and Nvidia RTX3080 GPU. We used a local runtime of Google Colaboratory and PyTorch 1.10.0 as a deep learning framework.

We used CIFAR-10 [9] dataset for evaluating the proposed method. We used the ResNet family proposed in [7] in the experiments. A ResNet-56 model trained with CIFAR-10 was used as a baseline model without fault tolerance. Moreover, TMR of the ResNet-56 model was used as a baseline model with fault tolerance. ResNet-20 models, which are smaller than ResNet-56, were used as child CNN models. The optimizer was set to Adam. The number of epochs was set to 100. The learning rate was set to 0.001. In knowledge distillation, the temperature T of the softmax function was set to 10 for the soft target loss. The loss function was defined by (hard target loss)+(soft target loss) as described in Section III.

B. Evaluation indices

We evaluated the proposed method with two evaluation indices. The first one was accuracy. The accuracy was calculated by the number of correctly classified images over the total number of images. The second one was the number of operations. The number of operations was calculated by counting the multiplications and additions required for inference. We also evaluated how many operations could be reduced by the proposed method compared with a conventional TMR approach.

TABLE I
COMPARISON OF INFERENCE ACCURACY FOR FAULT-TOLERANT ENSEMBLE MODELS OF DIFFERENT SIZES OF THE SHARED PART. [%]

#models	Shared part / Model name								
	None / $M[None]$			Block1 of Layer1 / $M[B1]$			All blocks of Layer1 / $M[L1]$		
	min	max	ave	min	max	ave	min	max	ave
1	91.56	92.03	91.74	91.83	92.14	**92.00**	91.85	92.23	91.99
2	92.99	93.44	**93.15**	92.19	92.55	92.35	92.19	92.64	92.37
3	93.29	93.85	**93.60**	92.33	92.57	92.49	92.39	92.72	92.54
4	93.63	93.98	**93.81**	92.45	92.66	92.53	92.41	92.70	92.60

C. Effects of Ensemble

The proposed fault-tolerant CNN can detect a permanent fault of hardware and a temporal fault caused during the computation of the shared part. On the other hand, increasing the ratio of a shared part may reduce the difference among child models and may reduce the accuracy of an ensemble classifier consisting of child CNN models. Therefore, we evaluated the accuracy of single models and ensemble models consisting of 2 to 4 ResNet-20 models while changing the ratio of the shared part.

We first train a ResNet-20 with knowledge distillation from a trained ResNet-56 model. Then, we generated six child models by retraining the unique part of the ResNet-20 model. We evaluated all combinations of these six child models for each size of the ensemble model. That is, we calculated the average inference accuracies of 6 single models, 15 ensemble models consisting of 2 child models, 20 ensemble models consisting of 3 child models, and 15 ensemble models consisting of 4 child models, respectively.

ResNet-20 consists of three stages, where each stage consists of three residual blocks. A comparison of the three types of ensemble models M[None], M[B1], and M[L1] is shown in TABLE I where M[None] has no shared part, M[B1] shares the first block of the first layer, and M[L1] shares all blocks of the first layer. In all ensemble models, the accuracy improved by increasing the number of child models. On the other hand, compared to M[None], the ensemble of M[B1] and M[L1] had a smaller effect on accuracy improvement. For example, for the ensemble model consisting of 4 child models, the average accuracies of M[B1] and M[L1] were 1.28% and 1.21% lower than that of M[None], respectively. This is because the shared part reduces the diversity of the child models and makes it more difficult to obtain the ensemble effect.

D. Effects of knowledge distillation

We trained six teacher models T_k $(k = 1, 2, \ldots, 6)$ with knowledge distillation from the ResNet-56 model. Then, to confirm the effectiveness of the proposed training method, we constructed three types of ensemble models shown in TABLE II. For all ensemble models, the initial weights of each child model are set to the weights of T_6. The shared part is set to the first stage of ResNet-20. For the first ensemble model denoted by No Knowledge Distillation (NoKD), the unique part of each child model is retrained without knowledge distillation. For the second ensemble model denoted by Low Divergence Ensemble Model (LDEM), the unique part of each child model is retrained with knowledge distillation from T_6. The thrid ensemble model denoted by High Divergence Ensemble Model (HDEM) is the proposed training approach. For HDEM, the unique part of each child model is retrained with knowledge distillation from different teacher models. We show the evaluation results of NoKD, LDEM, and HDEM for each number of models in TABLE III. Compared with NoKD and LDEM, HDEM tends to achieve higher accuracy. The proposed training approach for the fault-tolerant ensemble model was effective in increasing the diversity of child models and achieving higher accuracy.

TABLE II
TRAINING SETTING OF ONE-GROUP ENSEMBLE MODELS

Child model	NoKD	LDEM		HDEM	
	Shared	Shared	Teacher	Shared	Teacher
C_1	T_6	T_6	T_6	T_6	T_1
C_2	T_6	T_6	T_6	T_6	T_2
C_3	T_6	T_6	T_6	T_6	T_3
C_4	T_6	T_6	T_6	T_6	T_4

NoKD: No Knowledge Distillation
LDEM: Low Divergence Ensemble Model
HDEM: High Divergence Ensemble Model

E. Effects of grouping

TABLE IV shows the initial weights and the teacher models of ensemble models HDEM(4MR) and HDEM(2-DMR). HDEM(4MR) consists of one group with $\boldsymbol{G_1} = \{1, 2, 3, 4\}$. HDEM(2-DMR) consists of two groups $\boldsymbol{G_1} = \{1, 2\}, \boldsymbol{G_2} = \{3, 4\}$. In HDEM(4MR), the weights of the common part are the same for all child models. On the other hand, in HDEM(2-DMR), the weights of the shared parts S_1 and S_2 are different from those of S_3 and S_4, allowing for a more diverse ensemble model. As a result, HDEM(2-DMR) achieved 0.66% higher accuracy than HDEM(4DMR) on average as shown in TABLE V.

F. Comparison with TMR

TABLE VI shows a comparison of Resnet-56's TMR, HDEM(4MR), and HDEM(2-DMR) in terms of inference accuracy and computational cost. The baseline accuracy of Resnet-56's TMR was 92.58%. HDEM(4MR)

TABLE III
COMPARISON OF FAULT-TOLERANT ENSEMBLE MODELS WITHOUT/WITH KNOWLEDGE DISTILLATION SHOWN IN TABLE II

#models	NoKD			LDEM			HDEM		
	min	max	ave	min	max	ave	min	max	ave
1	91.87	92.23	92.04	91.85	92.23	91.99	91.96	92.20	**92.09**
2	92.18	92.46	92.34	92.19	92.64	92.37	92.35	92.66	**92.48**
3	92.44	92.59	92.48	92.39	92.72	92.54	92.49	92.68	**92.61**
4	92.64	92.64	**92.64**	92.41	92.70	92.60	92.64	92.64	**92.64**

TABLE IV
TRAINING SETTING OF ONE-GROUP AND TWO-GROUP ENSEMBLE MODELS

Child model	HDEM(4MR) $G_1 = \{1,2,3,4\}$		HDEM(2-DMR) $G_1 = \{1,2\}, G_2 = \{3,4\}$	
	Shared	Teacher	Shared	Teacher
C_1	T_6	T_1	T_5	T_5
C_2	T_6	T_2	T_5	T_1
C_3	T_6	T_3	T_6	T_6
C_4	T_6	T_4	T_6	T_2

TABLE V
ACCURACY COMPARISON OF ONE-GROUP AND TWO-GROUP ENSEMBLE MODEL SHOWN IN TABLE IV

Model	min	max	ave
HDEM(4MR)	92.64	92.64	92.64
HDEM(2-DMR)	93.21	93.42	**93.30**

TABLE VI
ACCURACY AND COMPUTATIONAL COST COMPARISON IN CASE OF NO FAULTY

Model	Accuracy [%]	#operations [GFLOPs]
TMR of ResNet-56	92.58	18.6 (100.0%)
HDEM(4MR)	92.64	**8.0** (43.0%)
HDEM(2-DMR)	**93.30**	**8.0** (43.0%)

TABLE VII
ACCURACY AND COMPUTATIONAL COST COMPARISON IN CASE ONE HARDWARE MODULE IS FAULTED

Model	Accuracy [%]	#operations [GFLOPs]
TMR of ResNet-56	92.58	12.4 (100.0%)
HDEM(4MR)	**92.61**	6.0 (48.4%)
HDEM(2-DMR)	92.54	**4.0** (32.3%)

and HDEM(2-DMR) increase the accuracy to 92.64% and 93.30%, respectively. The fault-tolerant ensemble model configured with ResNet-20 allowed us to configure a slightly more accurate model than ResNet-56. In addition, the computational cost of the TMR for Resnet-56 is three times that of Resnet-56. Compared to Resnet-56's TMR, the computational cost of ResNet-20's HDEM (4MR) and HDEM (2-DMR) can be significantly reduced to 43%.

Next, TABLE VII shows a comparison of inference accuracy and computational cost when the hardware module running one child model is faulted. TMR can maintain the same inference accuracy even if one hardware module fails. Therefore, the accuracy of Resnet-56's TMR was 92.58%. On the other hand, the fault-tolerant ensemble model may decrease the accuracy because the number of available child models decreases. HDEM(4MR) can identify which hardware module has been faulted, allowing it to construct a 3-model ensemble model dynamically. The accuracy degradation of HDEM(4MR) was only 0.03%. Though HDEM(2-DMR) can identify a group of failed hardware modules, it cannot identify which hardware module in the group is faulted. Therefore, after a failure occurs, an ensemble model of two models in the group that does not include the failure is configured. For this reason, the accuracy degradation of HDEM(2-DMR) was larger than HDEM(4MR). Although the accuracy of the fault-tolerant ensemble model deteriorates when a failure occurs, it achieved an accuracy comparable to TMR even in case of failure.

VI. CONCLUSION

In mission-critical systems used for long periods of time, it is necessary to be prepared for sudden failures. Significantly increasing the computational cost to improve the reliability of image recognition is not always acceptable in edge devices, as it may lead to increased power consumption and execution time. In this paper, we have proposed low-computational-cost fault-tolerant ensemble CNN models that can dynamically reconfigure the ensemble using ensemble models running on non-faulty hardware modules. We have demonstrated that the ensemble fault-tolerant model achieved an accuracy comparable with (even better than) TMR with a much lower computational cost.

Ensemble learning with diverse knowledge distillation by graph representation, which has been proposed in [10] is also helpful to further improve the accuracy of the fault-tolerant ensemble model. Therefore, we will extend the training of the fault-tolerant ensemble model based on the graph representation. In this paper, we have focused on the configuration and training approach for a more accurate fault-tolerant ensemble model. We will also consider a circuit design suitable for the execution of fault-tolerant ensemble models.

References

[1] W. Wang, S. Yang, S. Bhardwaj, S. Vrudhula, F. Liu, and Y. Cao, "The impact of nbti effect on combinational circuit: Modeling, simulation, and analysis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 2, pp. 173–183, 2010.

[2] P. Bannon, G. Venkataramanan, D. D. Sarma, and E. Talpes, "Computer and redundancy solution for the full self-driving computer," in *2019 IEEE Hot Chips 31 Symposium (HCS)*, pp. 1–22, 2019.

[3] G. Hinton, O. Vinyals, J. Dean, *et al.*, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, vol. 2, no. 7, 2015.

[4] S. Salamin, G. Zervakis, O. Spantidi, I. Anagnostopoulos, J. Henkel, and H. Amrouch, "Reliability-aware quantization for anti-aging npus," in *2021 Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pp. 1460–1465, 2021.

[5] Y. Li, Y. Liu, M. Li, Y. Tian, B. Luo, and Q. Xu, "D2nn: A fine-grained dual modular redundancy framework for deep neural networks," in *Proceedings of the 35th Annual Computer Security Applications Conference*, ACSAC '19, (New York, NY, USA), p. 138–147, Association for Computing Machinery, 2019.

[6] Y. Owada, Y. Tomioka, and H. Saito, "Dual modular redundancy unit of convolutional layer for low-cost and reliable cnns," in *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 379–384, 2022.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[8] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

[9] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research),"

[10] N. Okamoto, T. Hirakawa, T. Yamashita, and H. Fujiyoshi, "Deep ensemble learning by diverse knowledge distillation for fine-grained object classification," in *European Conference on Computer Vision*, pp. 502–518, Springer, 2022.