Soft Error Tolerant Convolutional Neural Networks on FPGAs With Ensemble Learning

Zhen Gao, *Member, IEEE*, Han Zhang, Yi Yao, Jiajun Xiao, Shulin Zeng, Guangjun Ge, Yu Wang, *Fellow, IEEE*, Anees Ullah, *Member, IEEE*, and Pedro Reviriego, *Senior Member, IEEE*

***Abstract*— Convolutional neural networks (CNNs) are widely used in computer vision and natural language processing. Field-programmable gate arrays (FPGAs) are popular accelerators for CNNs. However, if used in critical applications, the reliability of FPGA-based CNNs becomes a priority because FPGAs are prone to suffer soft errors. Traditional protection schemes, such as triple modular redundancy (TMR), introduce a large overhead, which is not acceptable in resource-limited platforms. This article proposes to use an ensemble of weak CNNs to build a robust classifier with low cost. To have a group of base CNNs with low complexity and balanced similarity and diversity, residual neural networks (ResNets) with different layers (20/32/44/56) are combined in the ensemble system to replace a single strong ResNet 110. In addition, a robust combiner is designed based on the reliability evaluation of a single ResNet. Single ResNets with different layers and different ensemble schemes are implemented on the FPGA accelerator based on Xilinx Zynq 7000 SoC. The reliability of the ensemble systems is evaluated based on a large-scale fault injection platform and compared with that of the TMR-protected ResNet 110 and ResNet 20. Experiment results show that the proposed ensembles could effectively improve the system reliability when suffering soft errors with an overhead much lower than TMR.**

***Index Terms*— Convolutional neural networks (CNNs), ensemble, fault injection, field-programmable gate array (FPGA) accelerator, soft error tolerance.**

Manuscript received June 20, 2021; revised September 2, 2021 and November 17, 2021; accepted December 18, 2021. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62171313, in part by the NSFC Joint Foundation under Grant 20200509, in part by the ACHILLES Project under Grant PID2019-104207RB-I00, in part by the Go2Edge Network under Grant RED2018-102585-T funded by the Spanish Ministry of Science, and in part by the Department of Research and Innovation of Madrid Regional Authority with the EMPATIA-CM Research Project under Grant Y2018/TCS-5046. *(Corresponding author: Zhen Gao.)*

Zhen Gao, Yi Yao, and Jiajun Xiao are with the School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China (e-mail: zgao@tju.edu.cn; yiyao@tju.edu.cn; destinedone@tju.edu.cn).

Han Zhang is with the Tianjin International Engineering Institute, Tianjin University, Tianjin 300072, China (e-mail: z_han@tju.edu.cn).

Shulin Zeng, Guangjun Ge, and Yu Wang are with the School of Electronic Engineering, Tsinghua University, Beijing 100084, China (e-mail: zengsl18@mails.tsinghua.edu.cn; geguangjun@126.com; yu-wang@mail.tsinghua.edu.cn).

Anees Ullah is with the Department of Electronics Engineering, University of Engineering and Technology, Peshawar, Abbottabad 220101, Pakistan (e-mail: aneesullah@uetpeshawar.edu.pk).

Pedro Reviriego is with the Department of Telematic Engineering, Universidad Carlos III de Madrid, 28911 Leganés, Spain (e-mail: revirieg@it.uc3m.es).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TVLSI.2021.3138491.

Digital Object Identifier 10.1109/TVLSI.2021.3138491

I. Introduction

CONVOLUTIONAL neural networks (CNNs) are one of the most successful deep neural networks (DNNs) or artificial intelligence (AI) classifiers. CNNs are widely used in the fields of image or speech recognition, and several implementations are widely adopted, for example, VGGNet [1], GoogLeNet [2], and residual neural network (ResNet) [3]. CNNs usually involve massive computations and memory accesses and, thus, make heavy demands on the performance and energy efficiency of the computing platform. Due to the limitations on the energy efficiency of general-purpose platforms (i.e., CPUs and GPUs), field-programmable gate array (FPGA)-based CNN accelerators are becoming more and more popular and have been widely studied to offer high performance with low power consumption [4], [5]. To improve the system flexibility, the implementation of SRAM-based FPGAs (SRAM-FPGAs) is preferred due to their reconfiguration capability.

The use of CNNs in safety-critical applications, e.g., automotive or onboard processing (OBP) in space, poses several challenges as reliability becomes a key requirement [6], [7]. Therefore, understanding the impact of errors on CNNs is an important issue [8]. In particular, when the CNN is implemented on an SRAM-FPGA, radiation-induced soft errors become an important concern as they can affect both the user and configuration memories [9]. In addition to degrading the CNN performance, faults in data access and control logic would also cause system exceptions, such as system stall and early termination [10]. Therefore, CNN systems need to be hardened for reliable execution when used in critical applications.

The most popular error protection method is triple modular redundancy (TMR) [11], [12]. One of its advantages is that the application of TMR is independent of the logic under protection, so it can be used to protect almost any module. However, the problem of TMR is that it more than triples the logic and power overhead, which may not be affordable for systems that are resource-limited in terms of computation and power, such as the embedded automotive systems or the OBP platforms [13]. Another approach is to try to use the algorithmic properties of the circuit to detect/correct errors, which is referred to as algorithm-based fault tolerance (ABFT) [14]. This strategy can reduce the overhead required to protect

a circuit. As reported in [15], neural networks are more resilient to computing errors compared to general applications, so many fault-tolerant designs have been proposed to make use of the characteristics of the neural networks. All these works focus on the protection of core components in CNN, e.g., the convolution calculation or weights stored in the RAM.

Instead of performing *ad hoc* mitigation techniques for each component of the CNN, this article proposes system-level protection for the CNN by replacing a single strong CNN with an ensemble of multiple weak CNNs so that all the components of the network are protected at a cost lower than TMR. The basic idea is that, when one or more of the weak CNNs fails, the ensemble of other CNNs can still operate reliably with little performance degradation. This can provide effective protection from all kinds of faults that may cause a base CNN to fail at a low cost. The basic idea has been initially explored in our previous work [16] based on simulations with a simple fault model. This article performs a full evaluation of both the implementation complexity (resource usage and processing time) and the reliability to understand the benefits of using ensembles. In particular, the main contributions of this article include the following.

1) Different ensemble schemes of ResNets and TMR-based baseline systems are implemented on an accelerator with advanced architecture on a single FPGA.
2) A robust combiner is designed to identify the faulty base classifier experiencing accuracy degradation or system exceptions and implemented in the FPGA-based ensemble system.
3) The reliability of different ensemble schemes and the robust combiner is evaluated based on fault injection experiments, and the improvements in reliability and resource efficiency are demonstrated by comparing with the TMR-protected ResNets.

The rest of this article is organized as follows. Section II introduces some preliminaries, including the effect of errors on CNN accelerators, fault tolerance for CNN accelerators, and a brief introduction of modern structures of FPGA-based CNN accelerators. Section III first introduces the basics of ensemble learning and the measurement of diversity between base learners and then emphasizes the differences for reliability-oriented ensemble design. Section IV discusses the implementation of the ensemble of multiple ResNets on an FPGA-based accelerator and the design of a robust combiner. Section V introduces the hardware fault injection experimental platform and compares the performance of different ensemble systems with that of a TMR-protected system in terms of reliability and resource usage. Finally, this article is concluded in Section VI.

II. Preliminaries

A. Faults on FPGA-Based CNN Accelerators

SRAM-FPGAs are sensitive to soft errors caused by cosmic radiation, and the single event upset (SEU) is typically the most frequent one [7], [9]. SEUs can corrupt both the configuration memory and the user memories (including registers and block RAMs) of SRAM-FPGAs [17], [18]. While errors on user memory only corrupt the data stored, errors on the configuration memory can change the circuit function and can only be corrected by reconfiguration. For CNN accelerators, the user memories are used to store the parameters (e.g., weights and bias) and feature maps, and all the processing modules (e.g., convolution, pooling, and control) are determined by the configuration memory.

Many works have studied the effect of faults on the parameters and feature maps. In [19], the reliability of CNNs with four layers was evaluated with faults on weights. The results show that CNNs with larger kernels are more robust. In [20], the reliability of LeNet-5 was evaluated with noise on the weights, and it was found that the layers that are closer to the output layer are more vulnerable. Reagen *et al.* [21] reveal that faults on weights and bias with an error rate lower than 10^{-4} would not degrade the accuracy, which is consistent with the results in [22] for AlexNet. Ozen and Orailoglu [23] performed a similar study for VGG16 and ResNet50 and showed that faults on weights with an error rate less than 10^{-7} will not degrade the accuracy, and the memories for feature maps are 50 times more tolerant to faults. In [8], the reliability of VGG 16, ResNet50, and InceptionV3 was evaluated with SEUs on weights and biases. The results show that VGG16 is less robust to the errors than the other two. Hoang *et al.* [24] also studied the reliability against faults on the weight memory and showed that the normalization layers and nonlinear activation operation are effective to reduce the impact of the faults.

Several recent works studied the reliability of FPGA-based CNN systems to faults on configuration memories based on fault injection experiments. Lopes *et al.* [25] implemented a simple NN with three layers based on SRAM-FPGA with separate resources for each layer, and the fault injection experiment results showed that soft errors on the last layer have a larger impact on the result. Similarly, Libano *et al.* [26] studied the reliability of FPGA-based CNN accelerators based on radiation experiments and found that most errors are tolerable, but errors on some layers may cause severe performance degradation. Israel *et al.* [27] implemented a CNN for traffic sign recognition based on SRAM-FPGA, and the experimental results showed that SEUs on about 20% of the configuration memories will cause wrong classifications. Libano *et al.* [28] evaluated the impact of quantization on the reliability of MINST CNN to SEUs on the configuration memories and found that a design with quantized convolutional layers is less sensitive to radiation, whereas the portion of errors that are critical is increased. Libano *et al.* [29] studied the impact of reduced data precision and degree of parallelism on the reliability of CNNs through neutron beam experiments. They concluded that the 8-bit integer design is much more reliable than a 32-bit floating-point implementation, and although larger parallelism increases radiation sensitivity, the performance gains outweigh it in terms of global failure rate. Xu *et al.* [10] performed a systematic evaluation of the FPGA-based CNN accelerators to faults on configuration memory based on fault injection experiments. Results showed that the system exceptions caused by faults on memory access logic and control unit *dominate* the reliability of the system. It should be noted that the experiments in [29] also report

system exceptions, but they only account for 10%∼20% of the total cross section. Such difference can be explained by the different accelerator architectures that are introduced in Section II-C.

Based on the above investigation, it can be concluded that SEUs on user memory have less impact on the CNN performance, and their effect has been better studied. On other hand, SEUs on configuration memory may cause severe problems, including performance degradation and system exceptions, but its effects on complex CNNs (especially on advanced FPGA accelerator architectures) are not well understood. Thus, in this article, we focus on the reliability evaluation and improvement against SEUs on the configuration memory of CNN accelerators.

B. Fault Tolerance for CNN Accelerators

This section provides a survey of fault tolerance for CNN accelerators, and the general mitigation techniques for soft errors are not included. Most of the related work can be grouped into two categories. The first is to make use of the parallel computation structure of the neural networks and introduce hardware or time redundancy for fault tolerance. In [30], the convolution between inputs and weights is rearranged into the form of matrix–matrix multiplication, and an error correction method based on checksums of columns and rows is applied to harden the matrix multiplications. Similarly, in [31], the convolution in CNN accelerators is modeled as a weight matrix and data vector multiplication, and redundant weight rows are generated based on coding for error detection and correction. Furthermore, Ozen and Orailoglu [22] proposed to combine the spatial checksum (redundant convolution kernel) and temporal checksum (redundant input feature maps) to protect against errors on inputs and computation for convolution layers. Zhao *et al.* [32] analyzed the protection ability of different checksum techniques for various convolution implementations and designed a workflow integrating all the techniques to obtain high protection. Marty *et al.* [33] discussed the simplified implementation of the checksum techniques in practical accelerators. The second is to consider the effect of faults during the model generation process. For example, Xu *et al.* [34] proposed an on-accelerator retraining scheme so that the computation error patterns are reflected in the model. Ning *et al.* [35] adopted the neural architecture search method to find reliable networks first and then apply a novel fault-tolerant training approach to obtain both reliable and high-precision models. Similarly, Zhang *et al.* [36] proposed to bypass faulty PEs by pruning and then recover the model performance with retraining. There are also some other works outside those two categories. Reagen *et al.* [21] proposed to detect errors on weights by double-sampling and to limit the error effect by setting the faulty bits to the sign of the word. Li *et al.* [8] and Marques *et al.* [37] proposed selective protections of the higher bits of the feature map data or data path to reduce the impact of soft errors. Libano *et al.* [26] proposed to selectively protect the layers that are more vulnerable to errors by TMR. Finally, Li *et al.* [38] proposed to detect the faults on LUTs based on the predefined inputs during the free cycle of the processing element (PE) in the accelerator.

Based on the previous discussion, we found that most of the works focus on the faults on the convolution processing, either corrupting the function or the weights. However, as will be introduced in Section II-C, the CNN accelerator also consists of other components. SEUs on them may also cause a severe problem, e.g., system exceptions, but current solutions are not effective against these faults. This article aims at proposing a complete protection scheme at the system level so that faults on all components can be tolerated.

C. FPGA Accelerator for CNNs

FPGA-based CNN accelerators can be divided into two types [39]: streaming architectures (SAs) and single computation engines (SCEs). SAs directly map layers to different resources of an FPGA to perform computations directly and efficiently. The reliability evaluations in [25], [26], [28], and [29] are based on such kind of accelerator. However, with the rapid development of CNN algorithms, SCEs become more and more popular [4], [5]. Different from SAs, SCEs apply instruction-set architectures (ISAs) to reuse the same logic for the processing of different layers. The CNN operations are transformed into instructions that can be deployed on the hardware accelerator by the software compiler [39]. In this way, SCEs can easily handle different neural network architectures and parameters without reconfiguring the FPGA. The work of Xu *et al.* [10] is based on such an accelerator. This article applies the optimized version of the ISAs-based CNN accelerator "Angel Eye" proposed in [4] for the implementation of base networks and the ensemble system. "Angel Eye" is the initial prototype of Xilinx DPU [40], so the results in this article would be applicable for CNNs based on DPU.

As shown in Fig. 1, the basic structure of the "Angel Eye" accelerator is composed of the instruction dispatch module, data mover, compute module, and memory pool. In this article, all these modules are implemented on the programmable logic (PL) part of the Zynq 7000 SoC XC7Z045, and the processing system (PS or ARM) is used to provide the addresses of instructions, weights/bias, and input images in the DDR when the system starts. The instruction dispatch module is responsible for instruction parsing, scheduling, and dispatching. The LOAD and SAVE modules inside the data mover are responsible for data transfer between DDR and the on-chip memory pool. The compute module includes a CONV module for the convolution operation in the convolution and full connection layers, and an ALU module for nonconvolution operations, such as pooling and activation functions. Relative to the SA-based accelerator, the SCE architecture uses more resources for noncompute operations, including data moving and control. This explains why the SCE-based accelerator is more likely to crash due to soft errors than the SA-based one.

Among these modules, the CONV module is the key component that performs most of the computations and consumes most the of FPGA resources. The CONV module utilizes a PE array to realize parallel convolution operation in 3-D. As shown in Fig. 2, the memory pool feeds the PE array with

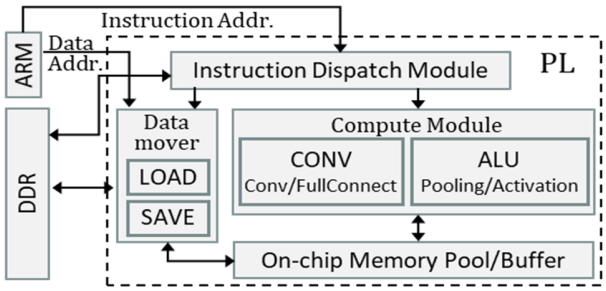

Fig. 1. Structure of the ISA-based FPGA CNN accelerator.

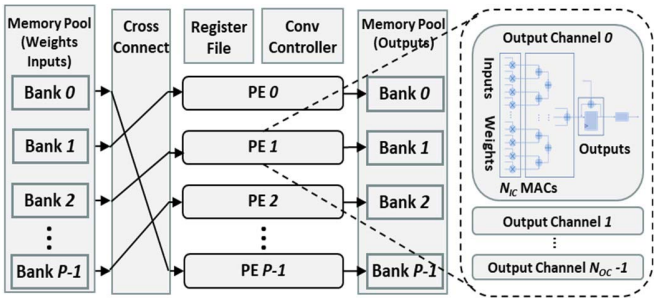

Fig. 2. Structure of the CONV module.

the inputs and weights, and P PEs process P input feature maps in parallel along the height dimension. Inside each PE, there are N_{OC} channels to process N_{OC} output channels, each with N_{IC} multipliers and an adder tree to process N_{IC} multiply-and-accumulations (MACs) in each cycle. Then, the outputs are written back to the memory pool.

In this work, the Xilinx Zynq-7000 SoC is used for the implementation, where the DSP48E2 block is used for MACs in the PE. Each of such DSP contains a 27×18 multiplier, which can implement two multiplications of two 8-bit integers. For the case that input image data and weights are 8-bit integers, the total parallelism of the CNN accelerator is $2 * P * N_{\mathrm{IC}} * N_{\mathrm{OC}}$. By adjusting the three parameters, we can achieve different tradeoffs between the CNN computation performance and the hardware resource utilization.

III. Ensemble Learning-Based CNN Architecture

In this section, we first introduce the concept of ensemble learning and then summarize some of the works that apply ensembles to improve the classification accuracy of CNNs in different fields. Finally, the basic idea to construct reliable CNNs using ensembles is discussed.

A. Concept of Ensemble Learning

Ensemble learning builds a set of independent learners and merges their results to improve task performance [41]. As shown in Fig. 3, an ensemble is formed by several learners called base learners, which are generated from training data by a base learning algorithm, which can be a decision tree, a neural network, or any other kind of learning algorithm. Ensemble methods may use a single base learning algorithm or use multiple learning algorithms to produce homogeneous or heterogeneous ensembles. The method used to combine the results could be as simple as voting or averaging the category scores in classification problems. Another option is

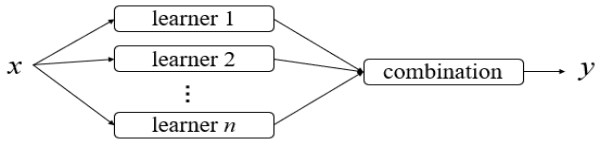

Fig. 3. Common ensemble architecture [41].

to use an additional learner that is trained to combine the base learners [41]. The generalization capability of an ensemble is often stronger than its base learners. In fact, ensemble methods are of interest mostly because they can combine weak learners that are only slightly better than a random guess to build strong learners that are able to make accurate predictions. For the case study in this article, the base learners in Fig. 3 are the ResNets, the input x is images from CIFAR-10, and the result y is the classified label.

B. Ensembles to Improve Accuracy of CNNs

Ensembles are widely used to improve the target detection or classification accuracy of CNNs in different applications. To have diverse base learners, different training data or different network structures can be applied. For example, different datasets are used to train the same network in [42] and [43]. In [42], to improve the classification accuracy of cancerous tissue, images with different pixel offsets are used to train the same CNN structure, leading to heterogeneous classifiers. Similarly, different cross sections of a lung nodule are obtained from multiple view angles, which are used to generate multiple CNNs by training a common CNN structure [43]. In [44] and [45], the same dataset is used to train CNNs with different structures. In [44], AlexNet, GoogLeNet, and ResNet are trained based on the same dataset to improve the food classification accuracy. In [45], age estimation is made based on the ensemble of AgeNet, RaceNet, and GenderNet, which are trained by the same image set. Furthermore, Ding and Tao [46] and Pons and Masip [47] apply different data and CNN structures simultaneously to improve the accuracy of face recognition and facial expression classification, respectively. In general, heterogeneous CNN structures are more effective to produce diversity than using different datasets to train the same network.

There are two common characteristics in the previous works to improve classification accuracy when using an ensemble: 1) base networks are usually strong, and the complexity is usually not the core consideration and 2) base networks are expected to complement each other so that the ensemble is able to obtain the correct result. A common method to evaluate the ensemble diversity is to measure the pairwise dissimilarity between two learners and then average all the pairwise measurements for the overall evaluation. Given a dataset with m data, the classification results of two base classifiers can be summarized with four numbers a, b, c, and d, which denotes the number of data that could be classified correctly by both classifiers $(++)$, the number of data that could be classified by one classifier but not by the other one $(+-$ or $-+)$, and the number of data that both classifiers cannot identify correctly $(--)$, respectively, and we will have

$a+b+c+d=m$. Then, the disagreement (dis) between the two classifiers can be measured as [41]

$$\text{dis} = (b+c)/m. \tag{1}$$

For classifiers with similar performance, higher dis usually implies higher ensemble performance.

C. Ensembles to Improve Reliability of CNNs

As discussed in the introduction, the focus of this article is to explore the use of ensembles to improve the reliability of CNNs with low overhead. As introduced in Section II-A, the reliability for single CNNs is measured by the probability of system exceptions and loss of accuracy when the soft error occurs, while, for ensemble system, the reliability is measured by the accuracy of the system when soft errors occur on the configuration memory, causing performance degradation or system exceptions on one or a few base networks. This idea is inspired by three facts. First, radiation-induced soft errors have a local effect [9] so that the error would only affect one base network in most cases. Therefore, in an ensemble-based CNN, if the combination method is carefully designed so that the failing base networks can be excluded from the ensemble, the system performance would not degrade much due to the soft errors. Second, the ensemble of weak classifiers can achieve the same or higher performance than a strong classifier due to the diversity. Third, traditional TMR can be seen as a special case of the ensemble, in which the three classifiers are the same. In this case, the classification accuracy of the ensemble keeps the same, but the reliability is improved because the failure of any single module will not change the result. Based on these facts, if we want to improve the reliability of a strong CNN, instead of using a TMR system, we can alternatively build an ensemble system by replacing the three modules in the TMR system with three different simpler networks. Then, it is possible to achieve an equivalent accuracy and reliability as that of the TMR system due to the diversity of the base learners and, at the same time, to have an overhead much lower than TMR due to the low complexity of base learners.

There are two main differences between an accuracy-oriented ensemble and a reliability-oriented one. First, the accuracy-oriented design tends to have base networks as diverse as possible (less overlap on the images that can be correctly classified by each base network). In this case, if one of the base networks fails (system exceptions or severe performance degradation), the accuracy of the ensemble will decrease significantly, while, for reliability-oriented ensembles, some similarity is intentionally maintained between base networks so that, when one base network fails, the other ones can still achieve good accuracy. Since heterogeneous structures have been proved to be more effective to generate diversity [44]–[47], we propose to derive the base learners from the same basic module but with some key parameters different so that both diversity and similarity are inherently built among base classifiers. This is the reason that ResNets with different layers are selected as base classifiers in this work. Second, the ensemble for reliability improvement expects base learners to be simple so that the overall overhead can be lower than the TMR of a strong CNN with the same accuracy. In general, if both accuracy requirements and resource constraints are met, a reliability-oriented ensemble prefers to combine more and simpler base networks so that the failure of a single base network has negligible influence on the system performance. This principle is different from that for the ensemble to improve accuracy.

IV. Design and Implementation of Ensemble CNNs

In this section, we will first introduce the models for the base networks and their implementation on the FPGA accelerator in Sections IV-A and IV-B, respectively, which is the basis for the reliability evaluation of base ResNets in Section V-B. Then, the design and implementation for a robust combiner are covered in Section IV-C. On this basis, the implementation of different ensemble systems and the TMR-protected CNNs is introduced in Section IV-D, and the resource usage and accuracy for each scheme is analyzed, which builds the basis for the reliability evaluation of ensemble CNNs in Section V-C.

A. Base Networks of ResNet With Different Layers

ResNets was proposed in 2015 to ease the training of deep networks [3]. Since ResNet 152 is about eight times deeper than VGG 16 but with lower complexity, it has been widely applied for target classification in many applications. Based on the design principle for reliability-oriented ensembles in Section III-C, in this work, we try to replace a deep ResNet (110) with an ensemble of multiple shallow ResNets with different layers (20/32/44/56) to improve the reliability with an overhead lower than the TMR of ResNet 110. Since base networks are constructed with the same basic residual learning module, and we train them with the same dataset, some degree of similarity is built among these shallow ResNets. On the other hand, different depths must bring to each ResNet different feature extraction capabilities, which creates the diversity that is necessary for the ensemble to achieve high accuracy.

The CIFAR-10 dataset is used in this work for model training and performance evaluation of the base ResNets and the ensemble system. This dataset includes 60 000 pictures from ten categories, among which 50 000 pictures are used for training (training set), and the other 10 000 are used for the test (test set). The output of each ResNet on CIFAR 10 is a sequence with ten scores, and a higher score means a higher probability that the current input image belongs to the corresponding category. For the ensemble of multiple base learners in the fault-free case, the combiner can simply average the scores from the base learners and choose the category with the highest average score as the classification result, which is represented as a label between 0 and 9. Since our tests show that the accuracy by testing 4000 images is very close to that using 10 000 images, we use 4000 images for accuracy testing in the experiments to reduce the time needed.

For the training of different ResNets, the weight decay and momentum are set to be 0.0001 and 0.9, respectively, as in [3], and the learning rate starts from 0.1 and is divided by 10 at 32k and 48k iterations. The weight initialization and batch

TABLE I
RESOURCE CONSUMPTION OF DIFFERENT RESNETS (CIFAR-10)

ResNets	20	32	44	56	110
Parallelism	512				2048
LUTs (218600)	26605				44906
DSPs (900)	128				512
Registers (437200)	26901				48186
BRAM/36Kb (545)	88				221
Time/ms	3.2	4.8	6.7	8.3	8.8

normalization are the same as in [3] and [36]. The simple data augmentation proposed in [37] is applied, and the minibatch size is set to be 128.

B. Implementation of Base ResNets

After the fixed-point operation of the model parameters, the trained base ResNets could be easily implemented on the ISA-based FPGA accelerator by generating the corresponding instructions. Since the processing time and resource usage of the implemented CNN are determined by the parallelism of the accelerator, to make a fair comparison between the ensemble CNNs and the TMR-protected ones, the base ResNets in the ensemble system (20/32/44/56) are implemented with 512 parallelisms, and ResNet 110 is implemented with 2048 parallelism. As will be introduced later, this configuration can guarantee a similar processing time for the ensemble CNN and ResNet 110 so that we can focus on the comparison in terms of resource usage and reliability. For the 512-parallelism accelerator, four PEs run in parallel ($P = 4$), and 32 DSPs are used in each PE to produce results for eight output channels ($N_{\mathrm{OC}} = 8$) simultaneously based on eight input channels ($N_{\mathrm{IC}} = 8$), while, for the 2048 parallelism, the number of PEs is the same ($P = 4$), but 128 DSPs are used in each PE to produce results for 16 output channels ($N_{\mathrm{OC}} = 16$) simultaneously based on 16 input channels ($N_{\mathrm{IC}} = 16$). All the base ResNets and the hardened system are implemented using Verilog in Vivado 2018.2 and mapped on the PL part of the Zynq SoC XC7Z045 for performance and reliability evaluation. When the system starts, 4000 images are first downloaded from the SD card into DDR, and then, the data mover will read in the images one by one for classification. Finally, the classification results are stored in DDR for accuracy analysis.

The resource usage and processing time (for one image) for each ResNet are listed in Table I, in which the total available resources of the target circuit are also provided in the first column. As we can see, the usages of LUTs, DSPs, registers, and BRAMs of the 2048-parallelism accelerator are about 1.7×, 4×, 1.8×, and 2.5× of that of 512-parallelism accelerator. DSPs are the key computation resource that determines the parallelism and the speed of the accelerator, and all of them are used in the CONV module. In addition, the processing time of the 512-parallelism accelerator for one $32 \times 32 \times 3$ image in CIFAR-10 is almost linearly proportional to the number of layers of ResNet, which implies that the throughput of the PE array is the bottleneck of the accelerator. On the other hand, for ResNet 110 (2× layers relative to ResNet 56) on 2048-parallelism accelerator (4× PE parallelism), the processing time for one image is about 1.06× of that of ResNet 56 on 512-parallelism accelerator, which implies that the bottleneck of the accelerator shifts from the PE array to the data mover.

TABLE II
ACCURACY OF DIFFERENT RESNETS (CIFAR-10)

Base ResNets	20	32	44	56	110
Accuracy (%)	90.70	91.98	92.39	92.50	92.67

The accuracy of the single ResNet is listed in Table II. As we can see, the accuracy of ResNet 20 is 90.7%, which increases for deeper ResNets, and achieves 92.67% for ResNet 110. These results are consistent with those reported in [3].

C. Design and Implementation of Robust Combiner

As investigated in [10], faults on the configuration memory of each base ResNet can cause both accuracy loss and system exceptions (mainly time out and early termination). The former is mainly caused by faults on PEs and the network that connects the array of PEs, and the latter is mainly caused by faults on the control module, data mover, and instructions logic. Thus, the combiner should be able to identify the faulty base learner in both cases and remove it from the final ensemble. In addition, as the reliability bottleneck of the ensemble system, the combiner should be able to tolerate errors.

The implementation structure for a combiner for three base ResNets is shown in Fig. 4. In the dashed box, the scores for ten categories from three base ResNets (S_1, S_2, and S_3) are input sequentially, and the scores for the same category are summed. Then, the index for the maximum sum score is taken as the final result. The parts outside of the dashed box are for fault tolerance. For the system exceptions, a cycle counter-based timer (T_1, T_2, or T_3) is used to determine whether the output enable signal of each base ResNet (EN_1, EN_2, or EN_3) is set in the normal time window. The processing time variance for each base ResNet is measured by experiments in advance. When one output enable signal is read outside of the normal time window, a system exception is reported, and the corresponding scores would be removed from the score averaging. On the other hand, to identify the faulty base ResNet producing wrong scores (faults on PE), the decision of each base ResNet is compared with the ensemble result, and a counter (C_1, C_2, or C_3) would increase by 1 for each consecutive mismatch. When a counter is larger than a threshold C_T, the score from the corresponding network is not used for score averaging. In the current implementation, C_T is set to be 4 based on the analysis of consecutive mismatches of faulty base ResNet presented in Section V-B.

The reliability of the combiner logic is tested using the fault injection platform introduced in Section V-A. As will be seen later in Section V-C (see Fig. 12), most of the faults will not affect the operation of the combiner, and about 20% of SEUs on the critical bits will dramatically degrade the ensemble accuracy below 40% (severe bits). This is reasonable because 80% of the logic (outside of the dashed box) in the combiner is used to identify the single faulty classifier. A fault on these parts may wrongly remove a score from the sum so

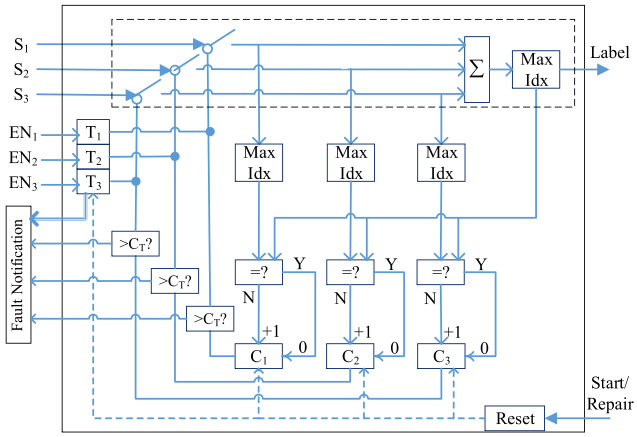

Fig. 4. Functional structure of the basic combiner.

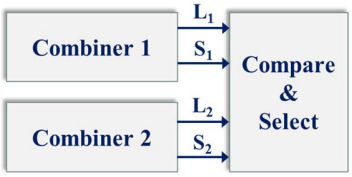

Fig. 5. Protected combiner with a DWC structure.

that the result is determined by the ensemble of the other two classifiers, which will only introduce a small accuracy degradation. Among the 20% severe bits, most of them are on the control logic that generates the reset and start signals so that both the output label and sum score are fixed to the initial value (0). In addition, the failure of the max operation in the dashed box may cause all the base classifiers to be detected as faulty, which will also cause zero-sum score and label. Both cases will cause an accuracy of around 10% (noting that label 0 represents the first category), which accounts for about 96% of the 20% severe bits, as shown in Fig. 12. The other 4% severe bits resulting in a classification accuracy of less than 40% come from the state machine of the sum operation in the dashed box. This module selects the scores from the normal classifiers to sum based on the detection results, so faults on this module would cause random selective addition of the input scores. A common characteristic for all the critical bits that change the output label is that the sum score is smaller than the expected value in the fault-free case. Based on this property, a duplicate with comparison (DWC) structure is proposed to improve the reliability of the combiner, as shown in Fig. 5. Each combiner will output the classified label (L) and the corresponding sum score (S). Then, the final block will first compare the two labels. If the two labels are different, the one with a higher score will be selected. Both the single combiner and the DWC-protected one are implemented on the PL part of Zynq 7000 SoC, and their resource usage is compared in Table III. As we can see, the DWC-protected combiner consumes a little bit more than twice the resources of the basic combiner, and the additional nine LUTs and four registers are from the final Compare & Select module. Compared with the main body of the ensemble CNN (see Table IV), the resource usage of the DWC-protected combiner is negligible. The reliability of the protected combiner will be evaluated in Section V-C.

TABLE III
RESOURCE CONSUMPTIONS OF THE COMBINER

	LUT	Registers
Single Combiner	134	101
DWC Combiner	277	206

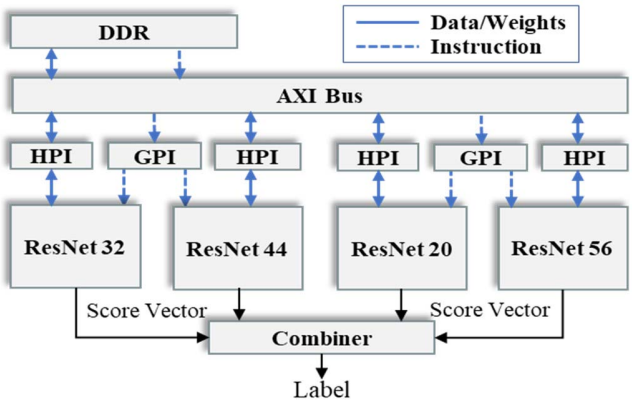

Fig. 6. Structure of ensemble system with multiple CNNs and combiner.

It should be noted that, similar to other DWC schemes, false detection may occur when faults occur on the "Compare & Select" circuitry [48], [49]. However, this only results in an unnecessary repair of one copy of the combiner and has no influence on the ensemble accuracy (as will be shown in Fig. 12). In addition, since only two comparators are used in the final module, the probability of false detection would be very low in practice.

D. Implementation of Ensemble ResNets

The ensemble of three or four base ResNets (including the combiner) was implemented on the PL part of a single Zynq 7000 SoC XC7Z045, and the structure is shown in Fig. 6. The data mover for data/weights and instructions is connected to the DDR memory through the AXI bus. The AXI bus in XC7Z045 supports two general-purpose interfaces (GPIs) and four high-performance interfaces (HPIs). Each accelerator is assigned an HPI for access data and weights, and two accelerators will share one GPI to read instructions from DDR. To decrease the impact of interface sharing in the accelerator throughput, ResNet 32 and ResNet 44 will share one GPI, and ResNet 20 and ResNet 56 will share the other one for the ensemble of four base learners. For the case that three base ResNets are combined, the two with fewer layers will share one AXI interface, and the one with most layers will use the other one by itself. We generated the instructions for different CNN tasks and assigned separate regions of DDR memory for each accelerator for data, weights, and instruction files. Then, each accelerator can access DDR for data/weights and instructions without interfering with each other. In the following experiments, the TMR-protected ResNet 20 is also implemented based on this structure for comparison. For TMR of ResNet 110, since the DSP usage exceeds the available DSPs in Zynq XC70Z45, it is only partially implemented with a single module and the voter, where the other two inputs of the voter are fed with the outputs of ResNet 110 in the fault-free case. Such implementation does not influence the

TABLE IV

RESOURCE USAGE OF DIFFERENT ENSEMBLES

Index	Ensemble scheme	Time/ms	LUT	DSP	Registers	BRAM
①	20 + 32 + 44	6.83	80269 (1.79x)	384 (0.75x)	81011 (1.68x)	264 (1.19x)
②	20 + 32 + 56	8.83				
③	20 + 44 + 56	8.83				
④	32 + 44 + 56	8.83				
⑤	20 + 32 + 44+ 56	8.95	106664 (2.37x)	512 (1x)	107740 (2.24x)	352 (1.59x)
Base	ResNet 110	8.8	44906	512	48186	221
R1	TMR ResNet 110	9.0	134773	1536	144605	663
R2	TMR ResNet 20	3.5	79870	384	80750	264

TABLE V

ACCURACY OF DIFFERENT ENSEMBLE SCHEMES (CIFAR-10)

Scheme	①	②	③	④	⑤	R1	R2
Accu.(%)	**94.07**	93.92	93.99	**94.28**	94.23	92.67	90.7

TABLE VI

DIVERSITY COMPARISON FOR ENSEMBLES ① AND ②

Ensembles	a (+ +)	b (+ -)	c (- +)	d (- -)	Dis
(20+32) + 44	3566	138	122	174	**0.065**
(20+32) + 56	3578	126	113	183	**0.059**

reliability evaluation of the TMR of ResNet 110 because the reliability of the three modules is the same.

The processing time (for one image) and the resource usage of the ensemble system with three or four base ResNets are listed in Table IV, in which those for single ResNet 110 and the two reference systems (TMR-protected ResNets 110 and 20) are also provided for comparison. As we can see, the processing times of ensemble systems and ResNet 110 are similar. For the ensemble system with three base ResNets, the usages of the LUTs and registers are about 1.7 times that of ResNet 110, and the usage of DSPs and BRAMs are 0.75 and 1.19 times of that of ResNet 110, respectively. Even for the case of four base ResNets combined, the overall resource usage is still much less than the TMR-protected ResNet 110. On the other hand, the accuracy of all the ensemble schemes and the reference systems on CIFAR-10 are listed in Table V. As we can see, all ensembles achieve an accuracy of around 94%, which is higher than that of ResNet 110 and the TMR-protected systems. Although the overhead of TMR of ResNet 20 is a little bit lower than that of the ensemble system with three ResNets due to a simpler combiner, the ensemble system outperforms the TMR of ResNet 20 by more than 3%. These results imply that the diversity within the ensemble system effectively improves the resource efficiency, which is seen in a lower resource usage relative to TMR of ResNet 110 or much higher accuracy relative to TMR of ResNet 20. Thus, it is promising to apply an ensemble of CNNs to achieve both better accuracy and improved reliability simultaneously with high resource efficiency.

As introduced in Section III-C, the performance of the ensemble scheme is determined by both the accuracy of base ResNets and the diversity among them. From Table V, we can identify two counterintuitive phenomena. One is that the ensemble scheme ① (20+32+44) outperforms scheme ② (20+32+56) and scheme ③ (20+44+56). The other is that the ensemble scheme ④ (32+44+56) outperforms scheme ⑤ (20+32+44+56). These phenomena can be explained

TABLE VII

DIVERSITY COMPARISON FOR ENSEMBLES ① AND ③

Ensembles	a (+ +)	b (+ -)	c (- +)	d (- -)	Dis
(20+44) + 32	3570	157	97	176	**0.063**
(20+44) + 56	3590	137	101	172	**0.059**

TABLE VIII

DIVERSITY COMPARISON FOR ENSEMBLES ④ AND ⑤

Ensembles	a (+ +)	b (+ -)	c (- +)	d (- -)	Dis
(32+44) + 56	3596	146	95	163	**0.060**
(32+44) + (20+56)	3647	95	93	165	**0.047**

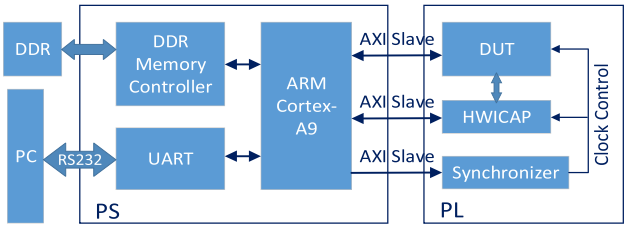

Fig. 7. Fault injection platform for CNN reliability evaluation.

Fig. 8. Fault injection platform with 20 XC7Z045 boards.

based on the diversity values listed in Tables VI–VIII, respectively. For example, Table VI compares the diversity between (20+32) and 44 and that between (20+32) and 56. Although ResNet 56 is a little bit stronger than ResNet 44, its complementarity to (20+32) is weaker than that from ResNet 44. Similarly, from Table VII, we can see that the complementarity of ResNet 32 to (20 + 44) is stronger than that of ResNet 56. Finally, adding ResNet 20 degrades the performance of ensemble (32+44+56) because the diversity between (32+44) and (20+56) is smaller than that between (32+44) and 56 based on Table VIII.

V. RELIABILITY EVALUATION OF ENSEMBLE CNNS

In this section, we first introduce the function and structure of the fault injection platform in Section V-A. Then, the reliability of single base ResNet is evaluated in Section V-B,

which provides a reference for the reliability improvement by the ensemble and the threshold setting in the combiner. Finally, the reliability of the ensemble CNNs is evaluated in Section V-C, and compared with the TMR-protected ResNet 110 and 20 to show the advantage of the proposed schemes.

A. Fault Injection Platform

To evaluate the reliability of the original ResNets and the ensemble of base ResNets, SEUs have been injected using an adapted version of the fault injection tool in [50] and implemented on Xilinx Zynq 7000 SoC (XC7Z045). As shown in Fig. 7, the injection platform consists of the PS and PL parts. The PS consists of the ARM Cortex-A9 processor and dedicated controllers for different peripherals, e.g., DDR memory controller, SD controller, and UART controller. The DDR is used to store the list of the configuration bits that are related to the design under test (DUT) (essential bits) and the feature maps of all layers for the fault-free accelerator on a reference image. The former is generated during the compile time in Vivado and is used by the injection algorithm for the reliability evaluation of the DUT. The latter is used to identify the essential bits that will affect the CNN functionality (critical bits). In our case, the DUT can be a single ResNet or the ensemble system. The SD controller is responsible for reading configuration files and images from the SD card, and the UART module is used to display the progress of the fault injection experiments on the PC. The PL part consists of the DUT and a synchronizer block. The latter is responsible for controlling the clock to DUT. Furthermore, the DUT reads images from the DDR and stores the processing results back to the DDR, which are then moved to the SD card when the experiments finish. Another important module housed by the PL part is the internal configuration access port (ICAP), which allows the ARM processor to access the configuration memory related to the DUT and modify it in run time for error injection or removal. The modules in the PL region are connected to the PS region through AXI buses.

The ARM processor runs the software that controls the fault injection process. The fault injection starts by freezing the clock to the DUT in the PL part through the synchronizer module. This is followed by reading back the target bit from a frame of the configuration memory through the ICAP port. The address of the configuration memory bits for fault injection is extracted from the fault list stored in DDR memory. The readback values are corrupted by inserting a bit flip for SEU emulation and written back. This is followed by resuming the design's clock. Since a fault on some configuration bits may not affect the neural network processing, the fault injection experiments are divided into two phases. The first phase is to locate all the essential bits that will affect the functionality of the CNN system. After fault injection on each of the essential bits, we compare the feature maps of all layers with those from the fault-free accelerator for the reference image. Then, the bit positions for which faults will introduce differences to the feature maps are identified as critical bits. The second phase is to test the accuracy loss caused by the SEU on each critical bit. For fault injection on each critical bit, the DUT will process 4000 images in the test set, and the classified labels for all images are finally copied from the SD card to the PC. The PC will then calculate the accuracy for that critical bit by comparing the received labels with the expected ones.

As we will see later, the most time-consuming part is to identify the critical bits from the millions of essential bits and test the classification accuracy of the single ResNet or the ensemble system with faults on each critical bit. This process may take more than three years with a single board. To speed up the experiments, we built a large-scale fault injection platform with 20 Zynq XC7Z045 boards. All the boards run in parallel, and each board is responsible for part of the fault bits in the list. In addition, the accuracy is tested using 4000 images from the test set of CIFAR-10. With this system, the total testing time was decreased to two months. The fault injection platform is shown in Fig. 8, where all the boards are connected to the PC through UART hubs for monitoring of the progress.

B. Reliability Evaluation of Base ResNets

To demonstrate the reliability improvement of the ensemble-based CNNs, we first evaluate the reliability of a single ResNet with different layers. Since the fault model for system exceptions is clearly known as time out or early termination based on [10], but there is no available fault model for accuracy degradation caused by SEUs on the PE array, we focus on the accuracy degradation by faults on configuration bits of PE array in this part.

The numbers of essential bits for a PE in 512- and 2048-parallelism accelerators are 355 349 and 1 158 600, respectively. Then, based on the method introduced in Section V-A, we obtained the critical bits for each ResNet. As we can see from Table IX, only SEUs on about 31% of the essential bits would affect the operation of the accelerator, and this portion is almost the same for ResNet with different layers. This means over 2/3 of the SEUs on the essential bits are tolerable to the accelerator. Then, the classification accuracy for each of the critical bits was tested, and the probability density function (pdf) curves are shown in Fig. 9. As we can see, for ResNets 20, 32, 44, and 56 on the 512-parallelism accelerator, about 70% of the critical bits will not degrade the classification accuracy much (accuracy $> 85\%$), and only less than 13% of the critical bits would cause poor results (accuracy $< 15\%$). However, for ResNet 110, on the 2048-parallelism accelerator, 80% of the critical bits will not cause obvious accuracy loss, and ones that will seriously degrade the classification performance decrease to 4.5%. This means that the accelerator with larger parallelism is more reliable to SEUs on the configuration memory. This can be explained by the fact that, in a less parallelized system, a PE would be involved in the calculation for more outputs in each layer. Therefore, a fault on it will have a larger impact. It is interesting to notice that this conclusion is consistent with the impact of parallelism for SA-based FPGA accelerator for MINST CNN in [29].

Furthermore, to provide a reference for the design of the combiner, the number of consecutive wrong results (N_{cw}) of a single ResNet with SEU on PE is studied taking ResNet 20

TABLE IX
CRITICAL BITS FOR ONE PE IN THE ACCELERATOR

Base ResNets	20	32	44	56	110
Essential bits	365349				1158600
Critical bits	116001	115975	116244	116454	363199

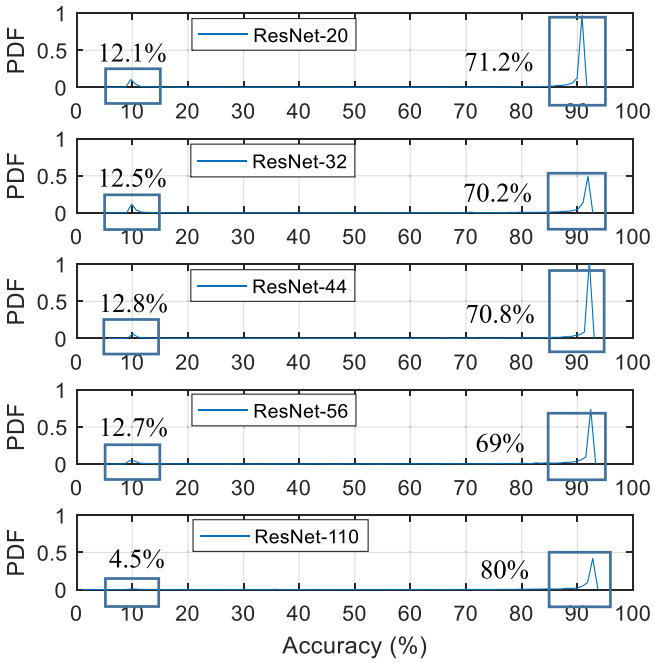

Fig. 9. PDF of accuracy for ResNets with faults on critical bits.

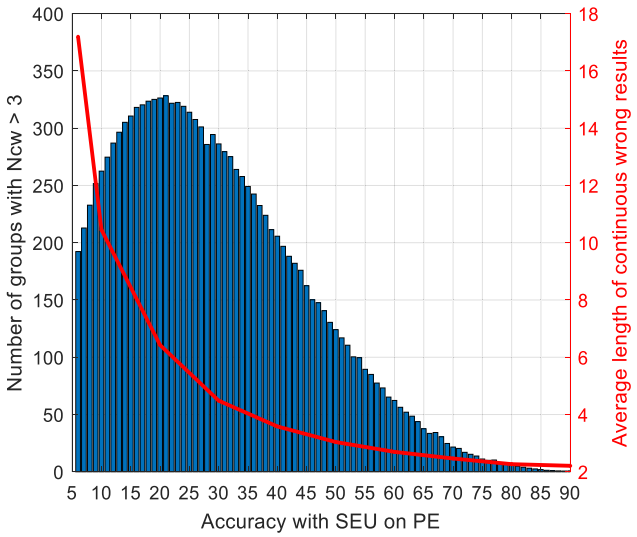

Fig. 10. Consecutive wrong results for different accuracies (ResNet 20).

as the example. The blue histogram in Fig. 10 shows the number of groups of more than three consecutive wrong results ($N_{\text{cw}} > 3$) for different accuracy degradation, and the red curve represents the average length of the consecutive wrong results. As we can see, the average N_{cw} decreases for higher accuracy, and the case of $N_{\text{cw}} > 3$ does not exist for an accuracy of 90%. Based on these results, we set the N_{cw} threshold as 4 ($C_T = 4$) in the combiner to detect base ResNet with fault on PE.

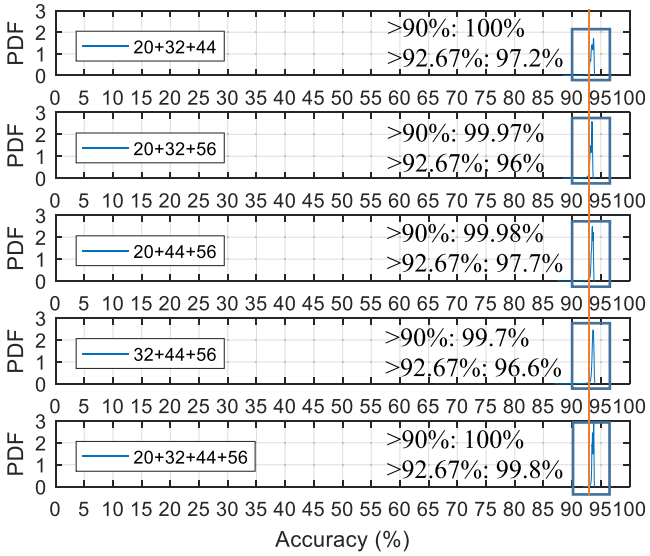

Fig. 11. PDF of accuracy for ensemble systems with faults on PE.

C. Reliability Evaluation of Ensemble ResNets

In this section, we first evaluate the reliability of different ensemble systems with faults on the PE array. Then, the reliability with faults causing system exceptions is discussed. Finally, the reliability of the DWC-protected combiner is evaluated.

1) Reliability With SEUs on PEs in a Single ResNet: Based on the system implementation introduced in Section IV-D, the accuracy of different ensemble schemes and TMR-protected systems with an SEU on the PE of one of the base ResNet is evaluated. For each ensemble scheme, we inject SEUs on the critical bits for each of the base ResNet and calculate the accuracy for SEU on each critical bit by comparing the output labels of the combiner with the expected ones. Then, the pdf of the accuracy for each ensemble scheme is plotted in Fig. 11. The SEUs on single PE do not degrade the accuracy of TMR-protected ResNet 110, and the result is marked with a red line at 92.67%. As we can see, the accuracy of the ensemble system is always over 90% for SEUs on almost all the critical bits. Compared with the last subfigure in Fig. 9, the reliability of the ensemble system is much higher than that of ResNet 110 for SEUs on PEs. This is expected because the critical bits that cause serious performance degradation are very likely to produce consecutive wrong outputs, so the faulty base ResNet can be identified by the combiner and removed from the ensemble. Further analysis of the pdf of the accuracy for each ensemble scheme shows that the percentages of times that the ensemble system outperforms the TMR of ResNet 110 are 97.2%, 96%, 97.7%, 96.6%, and 99.8% for schemes ①∼⑤, respectively. This implies that the reliability of the ensemble-based ResNet is even higher than the TMR-protected ResNet 110.

2) Reliability With System Exceptions of Single ResNet: The system exceptions mainly include time out and early termination, which directly causes system failure for a single ResNet without producing normal outputs [10]. The two effects can be easily emulated by stopping the clock or enabling the output

TABLE X
ACCURACY OF ENSEMBLE SCHEMES WITH SINGLE RESNET FAILURE

Scheme	①	②	③	④	⑤	R1	R2
Accu.(%)	93.47	93.40	93.67	93.82	94.07	92.67	90.7

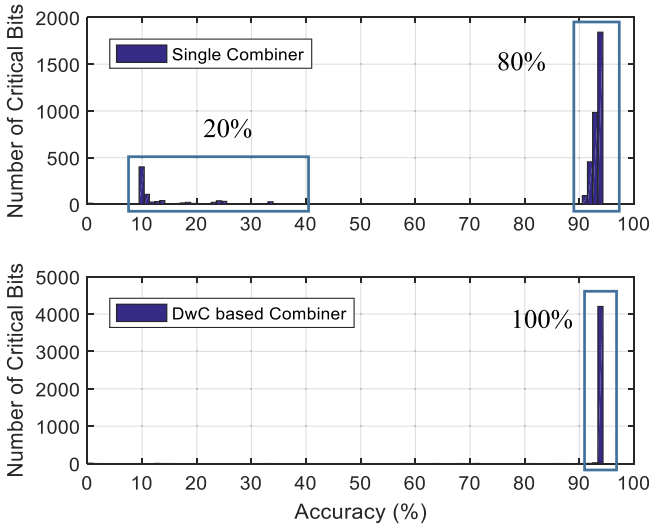

Fig. 12. Histogram of accuracy for unprotected and protected combiners.

validation signal for one base ResNet, respectively. In the test, time out and early termination are emulated for each base ResNet in an ensemble system, and the accuracy is evaluated by comparing the output labels with the expected ones. Then, the accuracy for the failure of each base ResNet in an ensemble system is averaged, as shown in Table X, in which the results for two reference systems are also provided for comparison. As we can see, the ensemble system achieves accuracy higher than 93% with a single base ResNet failure, which obviously outperforms the TMR-protected ResNets.

3) Faults on the Combiner: The reliability of the ensemble system could be compromised by the final combiner. In this part, the reliability of the basic combiner (see Fig. 4), and the DWC-protected one is evaluated for the ensemble of ResNets 20, 32, and 44. In the test, the input of the combiner includes the score vectors from three base ResNets for 4000 images. For the SEU on each essential bit of the combiner, 4000 output labels are recorded. If one of them is different from the output of a fault-free combiner, the essential bit would be recognized as critical. Based on the .ebd file generated by Vivado, the number of essential bits for the basic combiner is 27 385. Fault injection experiments show that 84.4% of the SEUs would not affect the normal operation of the basic combiner, and 4255 essential bits are identified as critical. The histogram bar chart of the accuracy for these critical bits is plotted in the first subfigure in Fig. 12. As we can see, SEUs on 20% of the critical bits will reduce the accuracy of the ensemble system to less than 40%. However, with the proposed DWC approach, the faulty copy can be identified 100% of the time, and the accuracy of the final decisions is that of the ensemble of ResNets 20, 32, and 44 in the fault-free case (94% as shown in the second subfigure in Fig. 12). This proves that the reliability of the combiner is effectively improved.

As a summary, the experimental results show that the use of the proposed ensemble of CNNs architecture on SCE FPGA accelerators can dramatically increase the reliability with an overhead much lower than TMR. Therefore, the proposed scheme is an attractive option to implement CNNs for safety-critical applications.

VI. CONCLUSION AND FUTURE WORK

The reliability of FPGA-based CNN accelerators is an important problem in critical environments. In this article, we proposed to replace a strong CNN with an ensemble of multiple weak CNNs to improve the reliability of the system. In particular, we choose ResNets with different layers as base CNNs and design a combiner to merge their results. In addition, a DWC structure is proposed to further enhance the reliability of the combiner. ResNets with different layers (20/32/44/56), different ensemble schemes, and TMR-protected ResNets are implemented on a modern ISA-based FPGA accelerator on Xilinx Zynq 7000 SoC. A large-scale hardware fault injection platform was built to test the reliability of base ResNet and the protected one with TMR or ensemble schemes. For base ResNets, experimental results show that most of the errors on the configuration memory could be tolerated by the network itself, but there are still errors on 12%∼13% of the critical bits that could dramatically degrade the system performance (accuracy around 10%). Instead, the experimental results for the proposed ensemble CNN show that the effect of a faulty base ResNet could be removed so that the system performance with a single faulty ResNet is higher than that of the TMR-protected ResNet 110. In addition, the resource usage of the ensemble system could be less than twice that of ResNet 110, which is much lower than the TMR-protected system. This proves the effectiveness of applying ensembles to improve the reliability of CNN accelerators with low cost and allows the designer to select among the different schemes to meet the resource usage, classification accuracy, and reliability requirements.

During the case study of the ensemble of ResNets, we found that some ensemble cases can achieve higher accuracy with lower resource usage. Therefore, an important question comes out, that is, whether it is possible to construct an optimal ensemble without training and testing all the possible cases. Our future work will try to answer this question. First, many deep learning solutions have been proposed to predict CNN accuracy based on its topology. Now, we are trying to extend the method to predict the diversity between different CNNs based on their topologies. Then, we can predict the accuracy of the ensemble system based on the prediction of the accuracy of each base CNN and the diversity between them. On this basis, we plan to apply the FT-NAS technique that we just proposed in [35] to find the optimal topologies of the base networks and build the optimal ensemble of CNNs under specific resource constraints.

REFERENCES

[1] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–14.
[2] C. Szegedy, W. Liu, and Y. Jia, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[4] K. Guo *et al.*, "Angel-Eye: A complete design flow for mapping CNN onto embedded FPGA," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 1, pp. 35–47, Jan. 2018.

[5] Y. Yu, C. Wu, T. Zhao, K. Wang, and L. He, "OPU: An FPGA-based overlay processor for convolutional neural networks," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 1, pp. 35–47, Jan. 2020.

[6] *Road Vehicles—Functional Safety*, Standard ISO 26262, 2011.

[7] N. Kanekawa, E. H. Ibe, T. Suga, and Y. Uematsu, *Dependability in Electronic Systems: Mitigation of Hardware Failures, Soft Errors, and Electro-Magnetic Disturbances*. New York, NY, USA: Springer-Verlag, 2010.

[8] G. Li *et al.*, "Understanding error propagation in deep learning neural network (DNN) accelerators and applications," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal. (SC)*, Denver, CO, USA, Nov. 2017, pp. 1–12.

[9] F. L. Kastensmidt, L. Carro, and R. Reis, *Fault-Tolerance Techniques for SRAM-Based FPGAs*. New Haven, CT, USA: Springer, 2006.

[10] D. Xu *et al.*, "Reliability evaluation and analysis of FPGA-based neural network acceleration system," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 29, no. 3, pp. 472–484, Mar. 2021.

[11] C. Carmichael, "Triple module redundancy design techniques for virtex FPGAs," Xilinx, San Jose, CA, USA, Appl. Note XAPP197, Version 1.0.1, 2006.

[12] J. Jonathan and W. Michael, "Voter insertion algorithms for FPGA designs using triple modular redundancy," in *Proc. FPGA*, Feb. 2010, pp. 249–258.

[13] P. L. Murray and D. VanBuren, "Single event effect mitigation in reconfigurable computers for space applications," in *Proc. IEEE Aerosp. Conf.*, Mar. 2005, pp. 1–7.

[14] A. L. N. Reddy and P. Banerjee, "Algorithm-based fault detection for signal processing applications," *IEEE Trans. Comput.*, vol. 39, no. 10, pp. 1304–1308, Oct. 1990.

[15] C. Torres-Huitzil and B. Girau, "Fault and error tolerance in neural networks: A review," *IEEE Access*, vol. 5, pp. 17322–17341, 2017.

[16] Z. Gao *et al.*, "Reliable classification with ensemble convolutional neural networks," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst. (DFT)*, Oct. 2020, pp. 1–4.

[17] P. S. Ostler *et al.*, "SRAM FPGA reliability analysis for harsh radiation environments," *IEEE Trans. Nucl. Sci.*, vol. 56, no. 6, pp. 3519–3526, Dec. 2009.

[18] M. Caffrey, P. Graham, E. Johnson, M. Wirthlin, and N. Rollins, "Single-event upsets in SRAM FPGAs," in *Proc. Mil. Aerosp. Appl. Program. Devices Technol. Conf. (MAPLD)*, Washington DC, USA, 2002, pp. 1–5.

[19] A. P. Arechiga and A. J. Michaels, "The effect of weight errors on neural networks," in *Proc. IEEE 8th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2018, pp. 190–196.

[20] S. Kwon, K. Lee, Y. Kim, K. Kim, C. Lee, and W. W. Ro, "Measuring error-tolerance in SRAM architecture on hardware accelerated neural network," in *Proc. IEEE Int. Conf. Consum. Electron.-Asia (ICCE-Asia)*, Oct. 2016, pp. 1–4.

[21] B. Reagen *et al.*, "Minerva: Enabling low-power, highly-accurate deep neural network accelerators," in *Proc. ACM/IEEE 43rd Annu. Int. Symp. Comput. Architecture (ISCA)*, Jun. 2016, pp. 267–278.

[22] E. Ozen and A. Orailoglu, "Sanity-check: Boosting the reliability of safety-critical deep neural network applications," in *Proc. IEEE 28th Asian Test Symp. (ATS)*, Dec. 2019, pp. 7–75.

[23] B. Reagen *et al.*, "Ares: A framework for quantifying the resilience of deep neural networks," in *Proc. 55th ACM/ESDA/IEEE Design Autom. Conf. (DAC)*, Jun. 2018, pp. 1–6.

[24] L.-H. Hoang, M. A. Hanif, and M. Shafique, "FT-ClipAct: Resilience analysis of deep neural networks and improving their fault tolerance using clipped activation," in *Proc. DATE*, Mar. 2020, pp. 1241–1246.

[25] I. C. Lopes, F. L. Kastensmidt, and A. A. Susin, "SEU susceptibility analysis of a feedforward neural network implemented in a SRAM-based FPGA," in *Proc. 18th IEEE Latin Amer. Test Symp. (LATS)*, Mar. 2017, pp. 1–6.

[26] F. Libano *et al.*, "Selective hardening for neural networks in FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 66, no. 1, pp. 216–222, Jan. 2019.

[27] I. C. Lopes, F. Benevenuti, F. L. Kastensmidt, A. A. Susin, and P. Rech, "Reliability analysis on case-study traffic sign convolutional neural network on APSoC," in *Proc. IEEE 19th Latin-Amer. Test Symp. (LATS)*, Mar. 2018, pp. 1–6.

[28] F. Libano, B. Wilson, M. Wirthlin, P. Rech, and J. Brunhaver, "Understanding the impact of quantization, accuracy, and radiation on the reliability of convolutional neural networks on FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 67, no. 7, pp. 1478–1484, Jul. 2020.

[29] F. Libano *et al.*, "How reduced data precision and degree of parallelism impact the reliability of convolutional neural networks on FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 68, no. 5, pp. 856–872, May 2021.

[30] F. F. dos Santos, L. Draghetti, L. Weigel, L. Carro, P. Navaux, and P. Rech, "Evaluation and mitigation of soft-errors in neural network-based object detection in three GPU architectures," in *Proc. 47th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. Workshops (DSN-W)*, Jun. 2017, pp. 169–176.

[31] Z. Xu and J. Abraham, "Safety design of a convolutional neural network accelerator with error localization and correction," in *Proc. IEEE Int. Test Conf. (ITC)*, Nov. 2019, pp. 1–10.

[32] K. Zhao *et al.*, "FT-CNN: Algorithm-based fault tolerance for convolutional neural networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1677–1689, Jul. 2021.

[33] T. Marty, T. Yuki, and S. Derrien, "Safe overclocking for CNN accelerators through algorithm-level error detection," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 12, pp. 4777–4790, Dec. 2020.

[34] D. Xu *et al.*, "Resilient neural network training for accelerators with computing errors," in *Proc. IEEE 30th Int. Conf. Appl.-Specific Syst., Architectures Processors (ASAP)*, Jul. 2019, pp. 99–102.

[35] X. Ning *et al.*, "FTT-NAS: Discovering fault-tolerant convolutional neural architecture," *ACM Trans. Design Autom. Electron. Syst.*, vol. 26, no. 6, pp. 1–24, Nov. 2021.

[36] J. J. Zhang, K. Basu, and S. Garg, "Fault-tolerant systolic array based accelerators for deep neural network execution," *IEEE Des. Test.*, vol. 36, no. 5, pp. 44–53, Oct. 2019.

[37] J. Marques, J. Andrade, and G. Falcao, "Unreliable memory operation on a convolutional neural network processor," in *Proc. IEEE Int. Workshop Signal Process. Syst. (SiPS)*, Lorient, France, Oct. 2017, pp. 1–6.

[38] W. Li *et al.*, "Soft error mitigation for deep convolution neural network on FPGA accelerators," in *Proc. 2nd IEEE Int. Conf. Artif. Intell. Circuits Syst. (AICAS)*, Aug. 2020, pp. 1–5.

[39] Y. Xing *et al.*, "DNNVM: End-to-end compiler leveraging heterogeneous optimizations on FPGA-based CNN accelerators," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 10, pp. 2668–2681, Oct. 2020.

[40] *Zynq DPU V3.2-Product Guide, PG338 (V3.2)*, Xilinx, San Jose, CA, USA, Mar. 2020.

[41] Z. Zhou, *Ensemble Methods: Foundations and Algorithms*. Boca Raton, FL, USA: CRC Press, 2012.

[42] K. Sirinukunwattana, S. E. A. Raza, Y.-W. Tsang, D. R. J. Snead, I. A. Cree, and N. M. Rajpoot, "Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images," *IEEE Trans. Med. Imag.*, vol. 35, no. 5, pp. 1196–1206, May 2016.

[43] P. Sahu, D. Yu, M. Dasari, F. Hou, and H. Qin, "A lightweight multi-section CNN for lung nodule classification and malignancy estimation," *IEEE J. Biomed. Health Informat.*, vol. 23, no. 3, pp. 960–968, May 2019.

[44] P. Pandey, A. Deepthi, B. Mandal, and N. B. Puhan, "FoodNet: Recognizing foods using ensemble of deep networks," *IEEE Signal Process. Lett.*, vol. 24, no. 12, pp. 1758–1762, Dec. 2017.

[45] M. Duan, K. Li, and K. Li, "An ensemble CNN2ELM for age estimation," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 3, pp. 758–772, Mar. 2018.

[46] C. Ding and D. Tao, "Robust face recognition via multimodal deep face representation," *IEEE Trans. Multimedia*, vol. 17, no. 11, pp. 2049–2058, Nov. 2015.

[47] G. Pons and D. Masip, "Supervised committee of convolutional neural networks in automated facial expression analysis," *IEEE Trans. Affect. Comput.*, vol. 9, no. 3, pp. 343–350, Jul./Sep. 2018.

[48] J. Johnson *et al.*, "Using duplication with compare for on-line error detection in FPGA-based designs," in *Proc. IEEE Aerosp. Conf.*, Big Sky, MT, USA, Mar. 2008, pp. 1–11.

[49] R. Gonzalez-Toral, P. Reviriego, J. A. Maestro, and Z. Gao, "A scheme to design concurrent error detection techniques for the fast Fourier transform implemented in SRAM-based FPGAs," *IEEE Trans. Comput.*, vol. 67, no. 7, pp. 1039–1045, Jul. 2018.

[50] A. Ullah, P. Reviriego, and J. A. Maestro, "An efficient methodology for on-chip SEU injection in flip-flops for Xilinx FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 65, no. 4, pp. 989–996, Apr. 2018.