

Resilient neural network training for accelerators with computing error

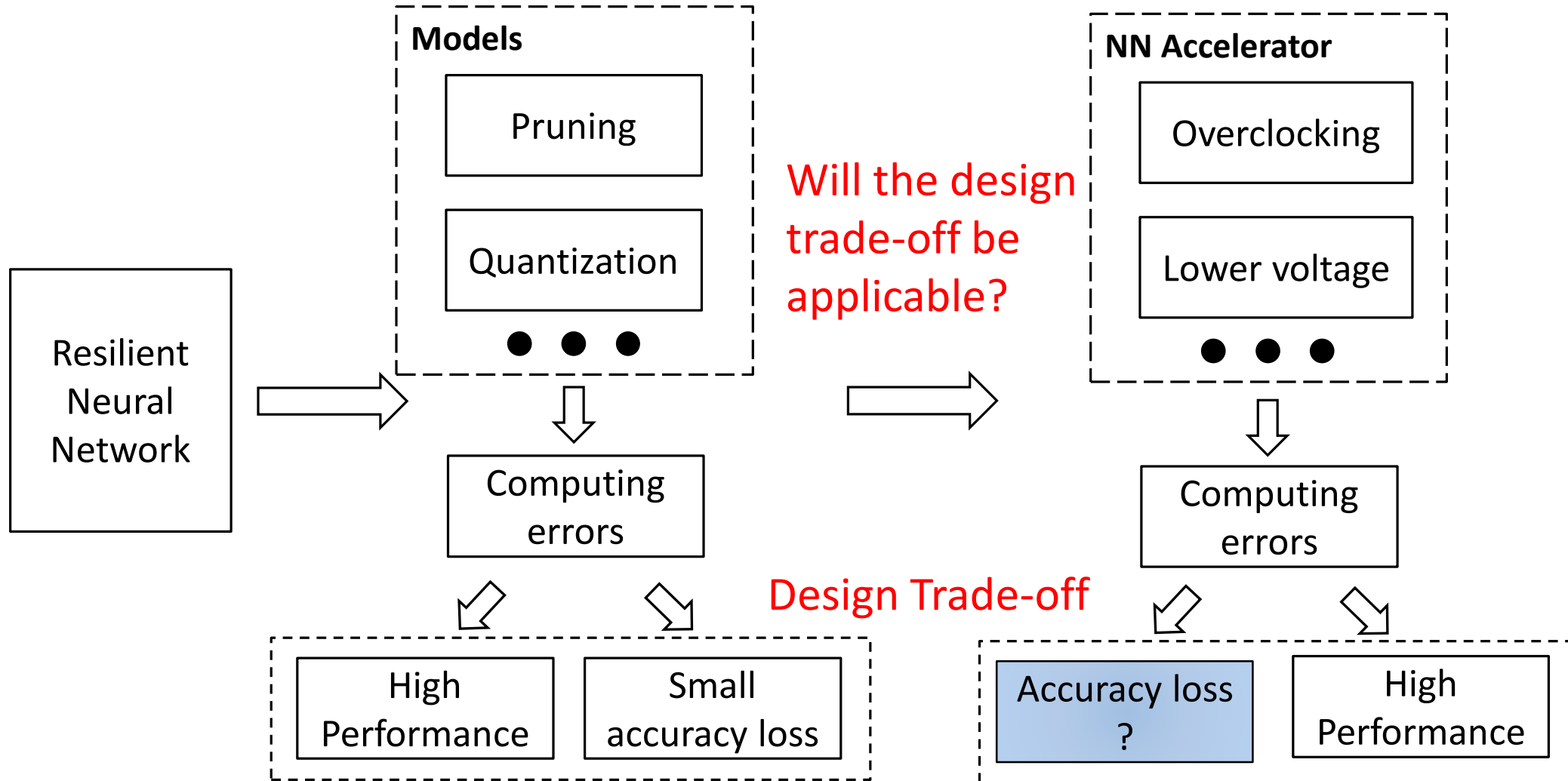
Dawen Xu^{*†}, Kouzi Xing[†], Cheng Liu^{*}, Ying Wang^{*}, Yulin Dai[†], Long Cheng[‡], Huawei Li^{*} and Lei Zhang^{*}

**Institute of Computing Technology, Chinese Academy of Sciences*

†Hefei University of Technology

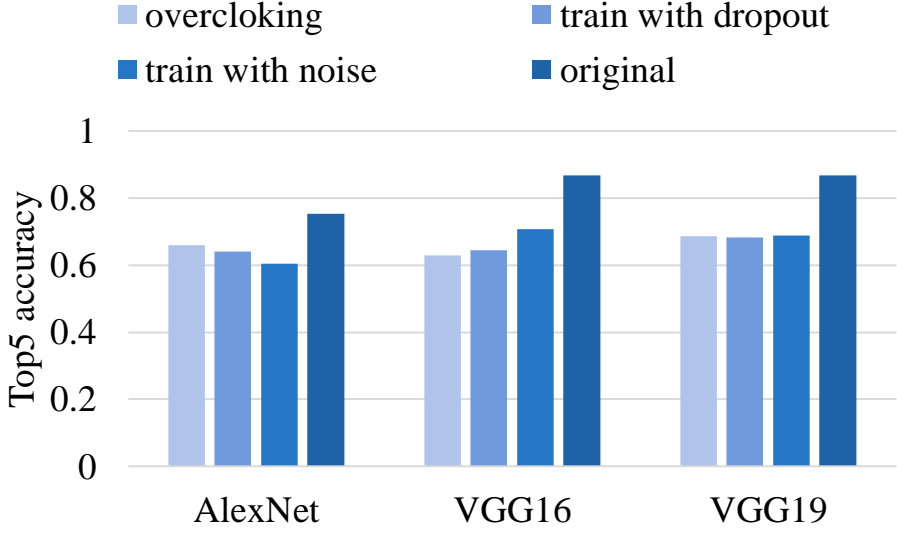
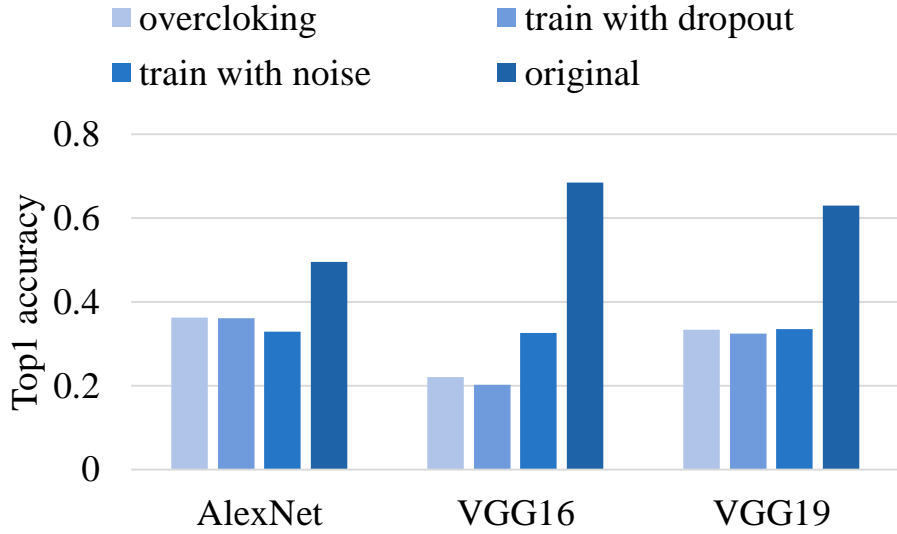
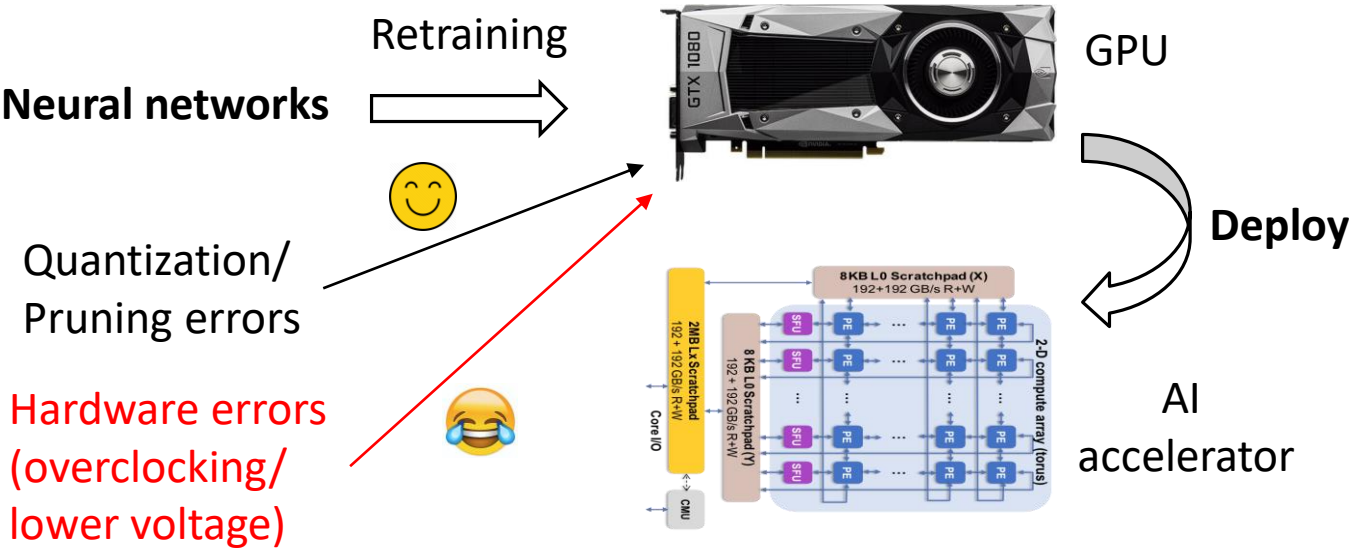
‡University College Dublin

Motivation: design trade-off between performance and accuracy



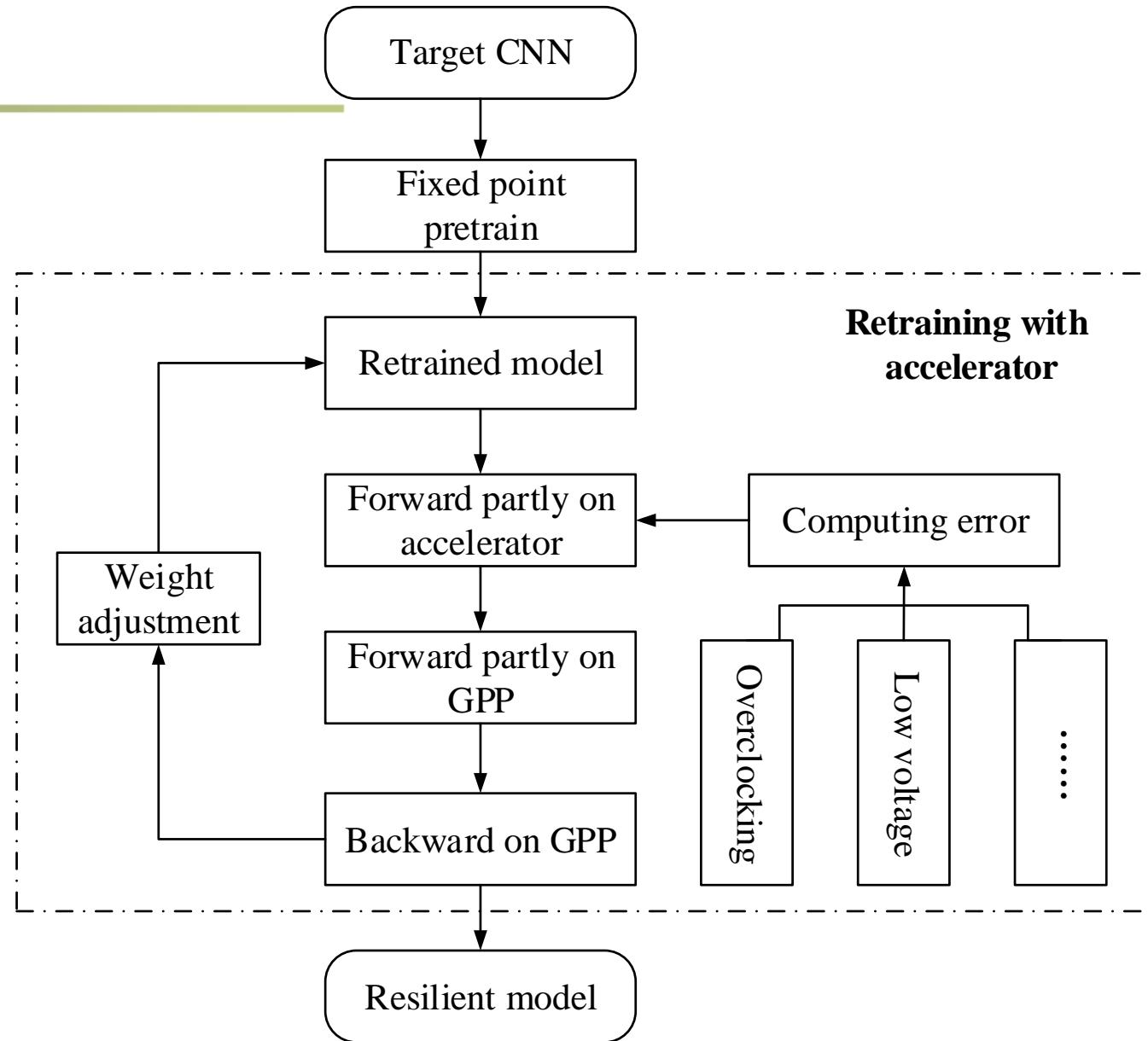
Motivation: retraining challenge

- When we apply the models directly to accelerator with computing errors (caused by overclocking), the neural network suffers considerable accuracy loss. **We need retraining!**
- How to expose hardware computing errors to GPP for retraining?



Resilient training framework

- ❑ To have the computing errors of an accelerator to be considered during retraining, we propose to integrate the accelerator into the training framework.
- ❑ To improve the neural network resilience for more efficient design trade-off, we opt to protect the most sensitive layer from being affected by computing errors.

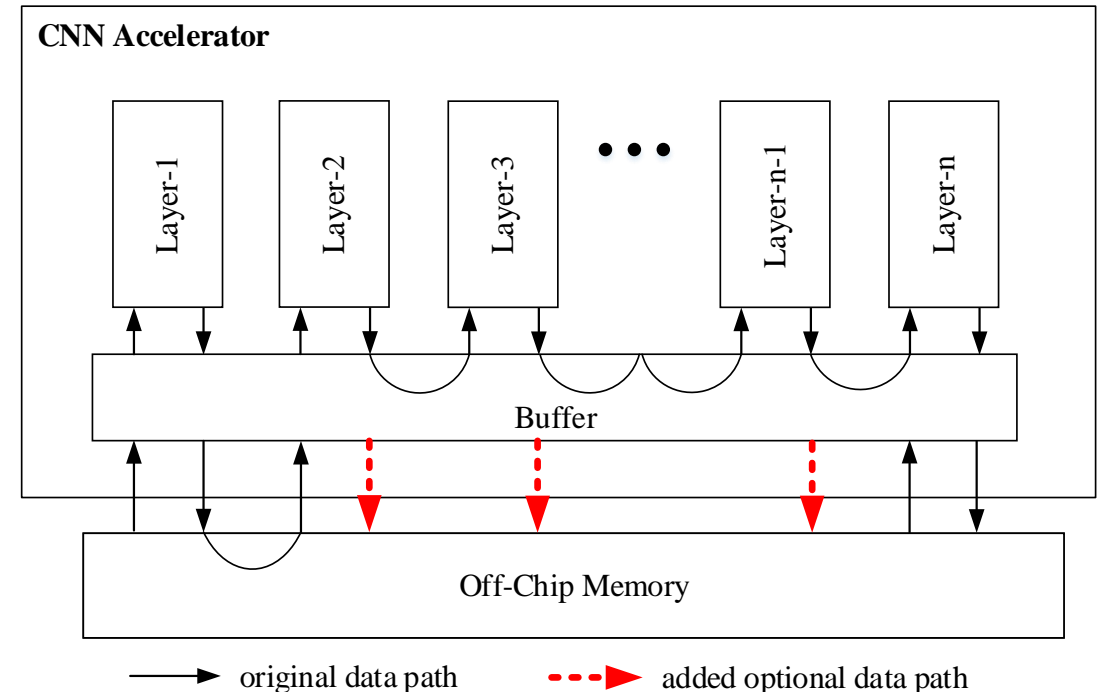


Integrate accelerator to training framework

- We define a set of abstract interfaces that are required to integrate general NN accelerators to Caffe for on-accelerator retraining.
- To integrate NN accelerator into the training framework, all the output of the intermediate layers must be accessible via main memory and additional data path may be required.

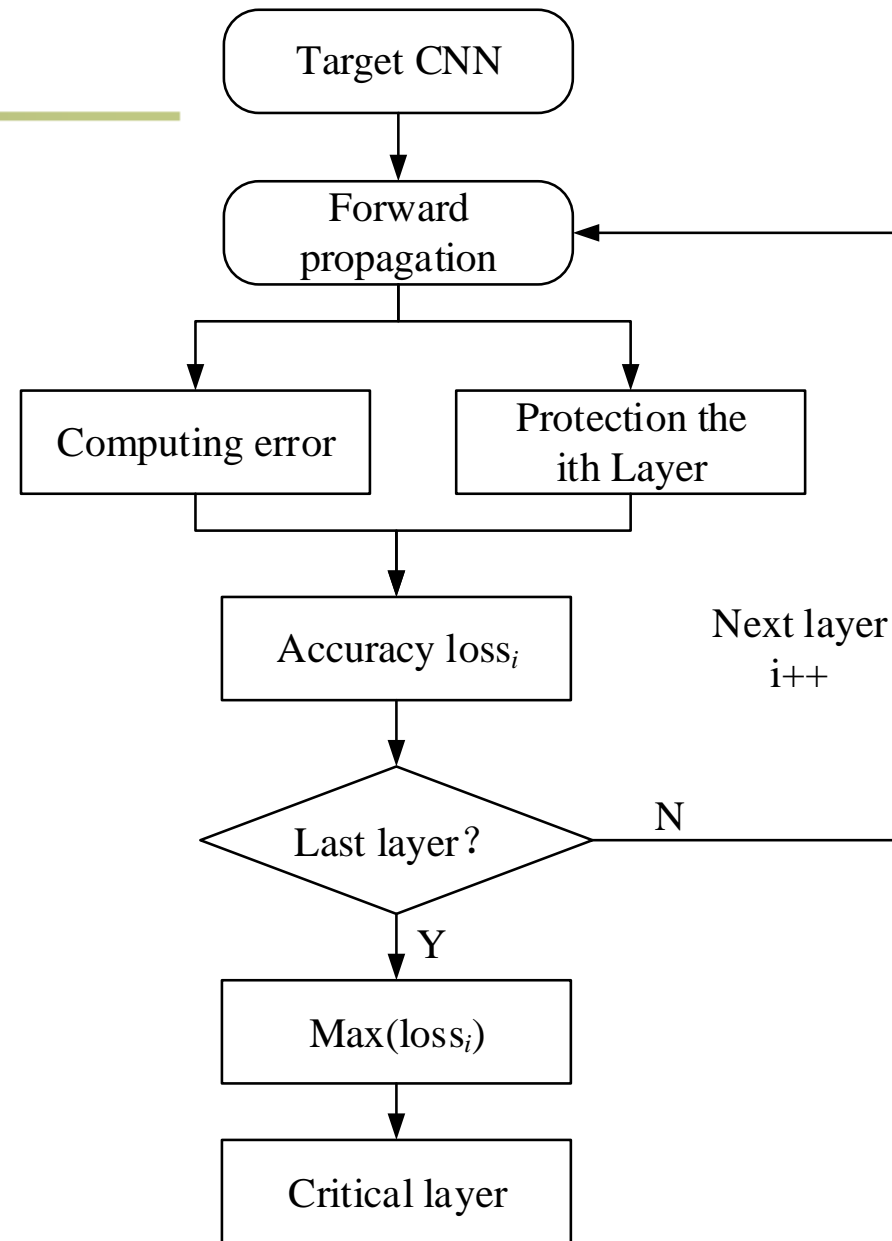
High-Level Interface

LaunchAccelerator()
DataToFPGA()
DataFromFPGA()
ConverIntToFloat()
ConverFloatToInt()
DataLayoutReorder()
DataLayoutRecover()



Improve neural network resilience

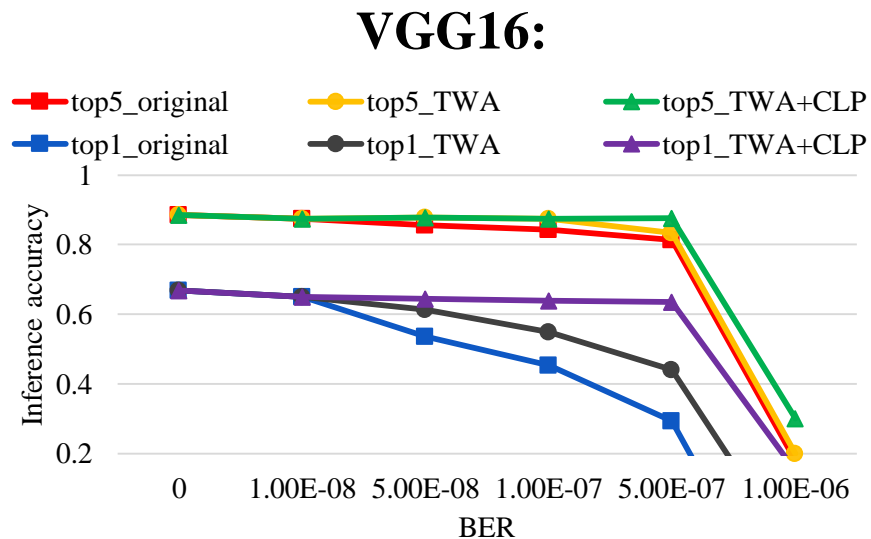
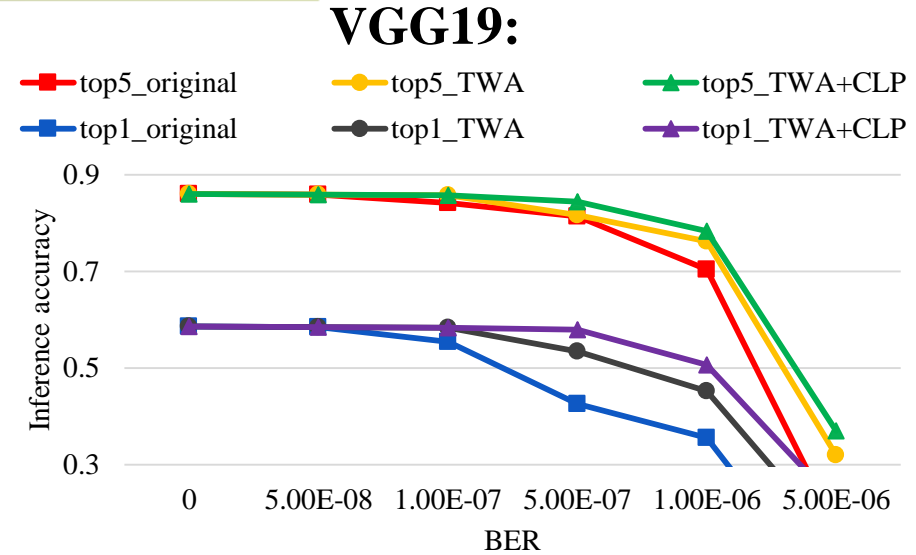
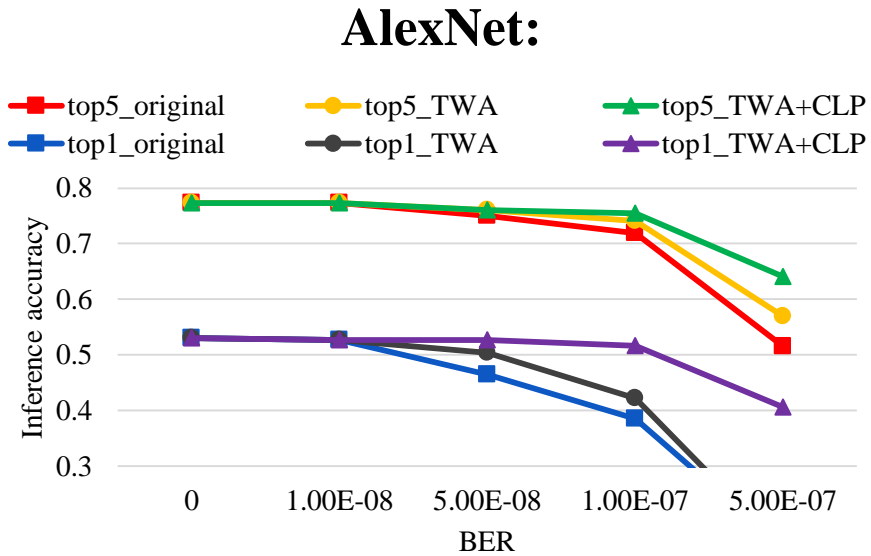
- ❑ The influence of computing errors on different layers of neural network varies. Some of them are more sensitive to errors.
- ❑ We use the actual computing errors as the sensitivity metric. The layer that has the highest percentage of large errors are taken as the critical layer.
- ❑ Protecting the critical layer from computing errors improves the overall neural network resilience.



Experiment setup

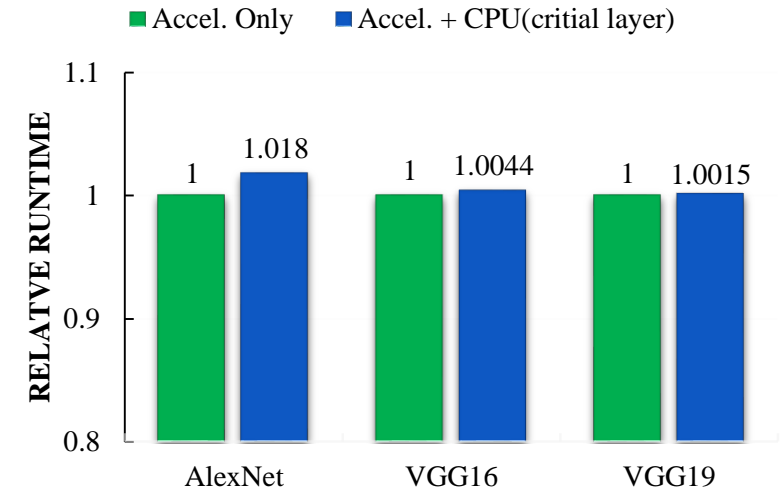
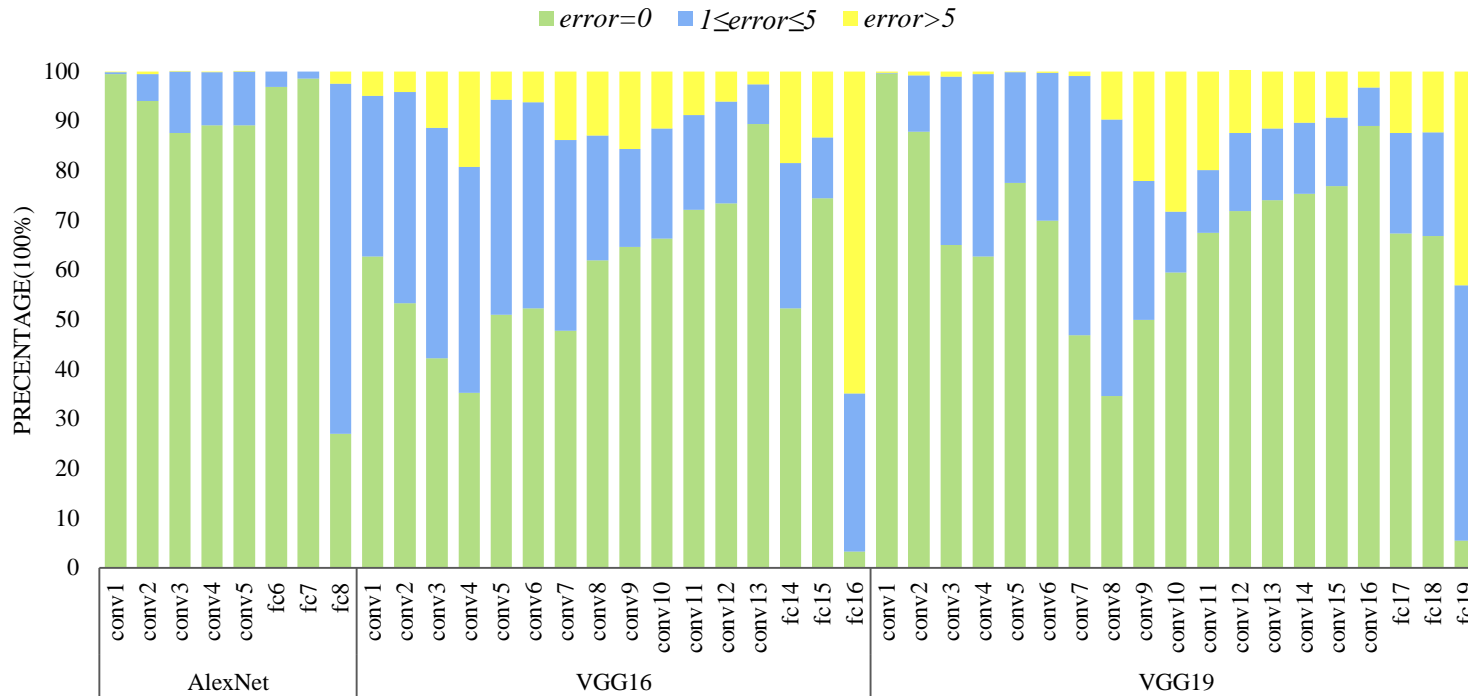
- ❑ **Error injection:** We try to simulate general computing errors and inject random bit errors to the neural network accelerator, so we use bit error rate (BER) as the error metric.
- ❑ **Dataset:** ImageNet
- ❑ **Neural networks:** AlexNet, VGG16 and VGG19
- ❑ **Host:** Intel (R) Core (TM) i7-6700 @3.40GHz, 32G DRAM
- ❑ **FPGA board:** Xilinx KCU1500
- ❑ **FPGA Design tools:** Xilinx SDAccel 2017.1
- ❑ **NN accelerator:** PipeCNN & simulator
- ❑ **Comparison:** Original NNs on accelerator with errors (Original), NNs trained with accelerator(TWA), and TWA with critical layer protected (TWA+CLP)

Experiments: accuracy comparison under random error injection



With the proposed approach, the Top1 and Top5 precision accuracy of the retrained models improves by 20.7% and 5.9% on average at the extreme yet acceptable error injection rate.

Experiments: critical layer selection

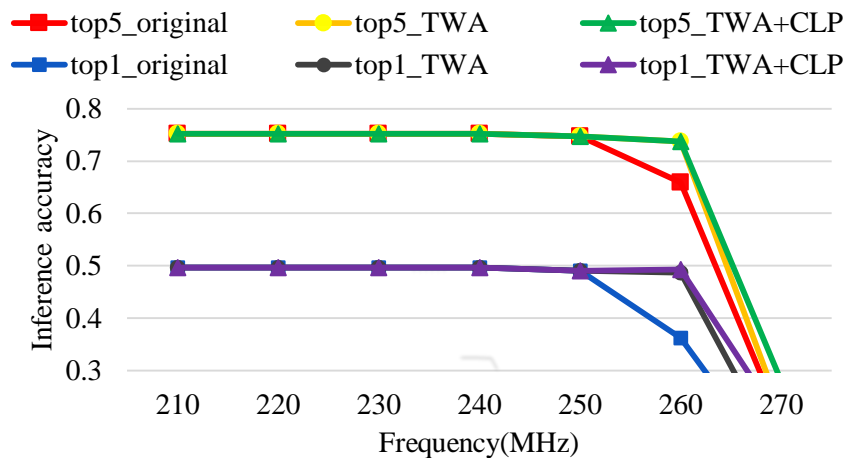


Error distribution of different neural network layers is used to determine the most sensitive layer. The last FC layer usually turns out to be the most sensitive layer.

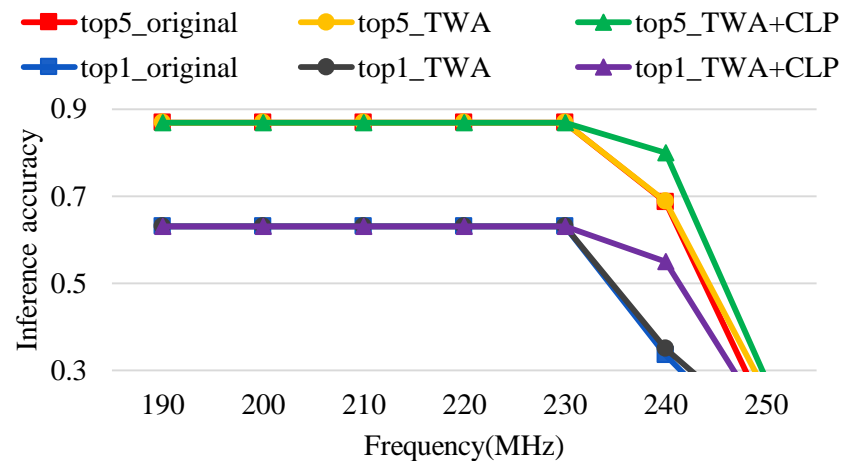
Performance overhead when the most sensitive layer is offloaded to a GPP.

Experiments: accuracy comparison under overclocking

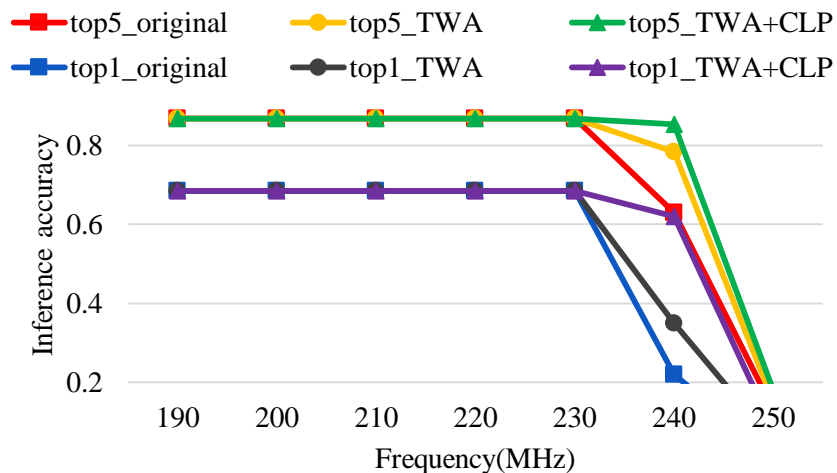
AlexNet:



VGG19:



VGG16:



At the tipping point, the Top1 and Top5 prediction accuracy of the retrained neural networks improves by 24.9% and 13.8% respectively.

Conclusion

- ❑ NN accelerators can benefit from the relaxed design constrain thanks to the inherent neural network resilience.
- ❑ Conventional training framework on GPPs can not take the accelerator computing errors into consideration during retraining. We propose an easy-to-use framework to allow general NN accelerators to be integrated into Caffe for on-accelerator retraining.
- ❑ Some of the neural network layers are more sensitive to computing errors. Thus, we can improve the resilience of the neural network models by protecting the most sensitive layer and obtain better design trade-off between neural network accuracy and performance.

Thanks