We thank the reviewers for insightful comments. We focus on answering the following concerns from reviewers:

(R1) Extend the evaluation to even larger models. Your recognition of our work is much appreciated. We plan to extend our method to more models and include the results as appendix in the final version of the paper.

(R2) Memory faults against its input/output tensors. Our work focus on protecting weights for two reasons. Firstly, weights are usually kept in the memory. The longer they are kept, the higher the number of bit flips they will suffer from. This easily results in a high fault rate (e.g. 1e-3) for weights. Input/output tensors, however, are useful only during an inference process. Given the slight chance of having a bit flip during an inference process (usually in milliseconds), protecting input/output tensors is not as pressing as protecting weights. Secondly, previous work [1] have shown that input/output tensors are much less sensitive to faults compared with weights.

(R2) Protection for lower than 8-bit. We agree with the reviewer that the optimal bit width for a network depends on its weight distribution and might be lower than 8. However, we have observed that 8-bit quantization is a prevalent, robust, and general choice to reduce model size and latency while preserving accuracy. In our experiments, both activations and weights are quantized to 8-bit. Existing libraries that support quantized DNNs (e.g. NVIDIA TensorRT, Intel MKL-DNN, Google's GEMMLOWP, Facebook's QNNPACK) mainly target for fast operators for 8-bit instead of lower bit width. With that said, we consider extending our protection approaches to lower bit width as future work.

(R2) DNN Scope and CNN choices. We agree with the reviewer about changing DNNs to CNNs and will make the change in the paper. We choose these CNN models as representatives because: 1) VGG is a typical CNN with stacked convolutional layers and widely used in transfer learning because of its robustness. 2) ResNets are representatives for CNNs with modular structure (e.g. Residual Module) and are widely used in advanced computer vision tasks such as object detection. 3) SqueezeNet has much fewer parameters and represents CNNs that are designed for mobile applications. We will add the explanations into the paper.

(R3) Theoretical analysis. As mentioned in the paper (line 120), the optimization problem WOT tries to solve can also be addressed using an ADMM-based approach similar to [2] that has the theoretical convergence guarantee. However, our empirical explorations have shown that WOT can recover the models' accuracy with less number of iterations. We will provide the comparison between the ADMM-based approach and WOT to supplement the paper.

(R3) Majority vote for low fault rate settings. When the fault rate is high (e.g. 1e-3), there is about 1% - 5% accuracy drop even with majority vote protection. As the accuracy drop with majority vote protection in large fault rate settings is significant, it is better used in low error rate scenarios.

(R3) Experiment settings. Only results in Figure 3, Figure 4(a) and Figure 5(a) are based on Cifar10. All the other results reported in the paper including Table 2 are collected using ImageNet. For Figure 5, WOT is applied to pre-trained models; the models are finetuned instead of trained from scratch. That's why WOT takes a small number of epochs on Cifar10 or iterations on ImageNet to recover the models' accuracy. Pre-trained SqueezeNet, after clipping the weights to [-64, 63], suffers from a large accuracy drop (56.99% dropped down to around 11.0%). Its accuracy is gradually recovered to 57.01% after fintuning with WOT for 30K iterations on ImageNet. Pre-trained VGG16 is more robust to weight clipping (93.75% dropped down to around 93.4%). Its accuracy is gradually recovered after 16 epochs' training using WOT on Cifar10. We will clarify the settings in the paper.

Answers to minor issues.
– (R2) Fault rate/bit error rate is the ratio between the number of bit flips experienced before correction is applied and the total number of bits.
– (R3) In Table 1, the values in percentage rows are absolute values. So the range is [0, 128] instead of [-128, 128].
– (R3) As pre-trained models are used, we don't have a script for training the models from scratch on ImageNet. However, we did include the script for WOT, i.e., regulated training. Because our pre-trained models come from TorchVision, the same script can be reused for the three models by changing the model name in the script.

References

[1] Brandon Reagen, Udit Gupta, Lillian Pentecost, Paul Whatmough, Sae Kyu Lee, Niamh Mulholland, David Brooks, and Gu-Yeon Wei. Ares: A framework for quantifying the resilience of deep neural networks. In *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2018.

[2] Ao Ren, Tianyun Zhang, Shaokai Ye, Jiayu Li, Wenyao Xu, Xuehai Qian, Xue Lin, and Yanzhi Wang. Admm-nn: An algorithm-hardware co-design framework of dnns using alternating direction methods of multipliers. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 925–938. ACM, 2019.