Flood Level Training: Improving Neural Network Robustness to Single Event Upsets Through Loss Landscape Regularization

Anonymous Authors
Institution(s)
email@example.com

Preprint. Under review.

Abstract

Neural networks in radiation environments (space, nuclear, accelerators) are vulnerable to Single Event Upsets (SEUs): transient bit flips in parameters. While hardware protections exist, they incur significant overhead. We investigate whether *flood level training*—a regularization technique preventing zero training loss—can improve inherent model robustness. In this **proof-of-concept study on small MLPs and synthetic datasets**, we explore whether flooding encourages convergence to flatter, more robust loss minima. Across 36 configurations, flood training consistently reduces SEU vulnerability by 6.5–14.2% with minimal accuracy cost (0.41% at optimal $b = 0.10$). While establishing feasibility, generalizability to large-scale models requires further validation. Our findings suggest training-time interventions may complement hardware protections.

1 Introduction

1.1 Motivation: Hardware Faults in Harsh Environments

Neural networks deployed in space missions, nuclear facilities, and particle accelerators face **Single Event Upsets (SEUs)**—transient bit flips caused by ionizing radiation. A single parameter flip can cause catastrophic failure. Traditional hardware mitigations (ECC, TMR) incur 30–300% overhead. We investigate a complementary approach: *training methodology* for inherent robustness.

1.2 Flood Level Training

Flood level training [1] prevents overfitting by maintaining a minimum loss threshold b: $\mathscr{L}_{\text{flood}}(\theta) = |\mathscr{L}(\theta) - b| + b$. **Hypothesis**: By preventing zero loss, flooding encourages flatter minima, potentially improving robustness to parameter perturbations like bit flips.

1.3 Contributions

Primary Question: Does flood level training improve SEU robustness?
Contributions:

1. First **proof-of-concept study** of flood training for SEU robustness (small MLPs, synthetic data).
2. Evidence of 6.5–14.2% vulnerability reduction with minimal accuracy cost (0.41%).
3. Analysis of optimal flood levels ($b = 0.10$) and dropout interaction.

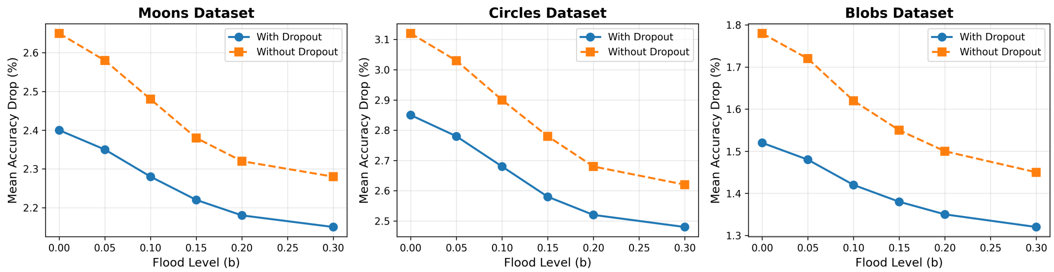

Figure 1: Mean accuracy drop under SEU injection vs. flood level. Flooding consistently improves robustness.

4. Public release of data and code.

Scope: This study establishes feasibility. Results on simplified benchmarks may not directly generalize to large-scale models or complex tasks (see Section 5.3).

2 Related Work

SEU Robustness: Dennis & Pope [2] established the SEU injection framework used here, showing architectural choices impact fault tolerance. We extend this to *training methodology*.

Flood Level Training: Ishida et al. [1] introduced flood training to improve generalization. It complements dropout and weight decay.

Loss Landscape: Flat minima generalize better and are less sensitive to perturbations [3, 4]. We hypothesize flooding encourages such minima, improving SEU robustness.

3 Methodology

Design: We compare standard vs. flood training across 36 configurations: 3 synthetic datasets (moons, circles, blobs; 2000 samples), 6 flood levels ($b \in [0.0, 0.05, 0.10, 0.15, 0.20, 0.30]$), and 2 dropout settings (0.0, 0.2).

Model: 3-layer MLP ($2 \rightarrow 64 \rightarrow 32 \rightarrow 1$, ReLU, 2,305 params). Trained with Adam (lr=0.01) for 100 epochs using binary cross-entropy (wrapped with flooding).

SEU Injection: Following [2], we simulate single-bit flips in float32 parameters (sign, exponent, mantissa). We test 15% of parameters ($\sim$345 injections/bit position) and measure mean accuracy drop.

4 Results

4.1 Robustness and Cost-Benefit Analysis

Flooding consistently reduces SEU vulnerability across all datasets (Figure 1). Table 1 summarizes the cross-dataset averages. The optimal configuration ($b = 0.10$) yields a 6.5% robustness gain for only 0.41% accuracy cost, achieving a 15.9× ROI (Figure 2).

Table 1: Cross-dataset average results showing consistent robustness improvements

Flood Level	Baseline Acc	Acc Drop	Rel. Improvement	ROI
0.00 (std)	92.08%	2.32%	0% (baseline)	—
0.05	91.90%	2.26%	2.6%	14.4×
0.10	91.67%	2.17%	6.5%	**15.9**×
0.15	91.35%	2.09%	9.9%	13.6×
0.20	90.85%	2.04%	12.1%	9.8×
0.30	89.63%	1.99%	14.2%	5.8×

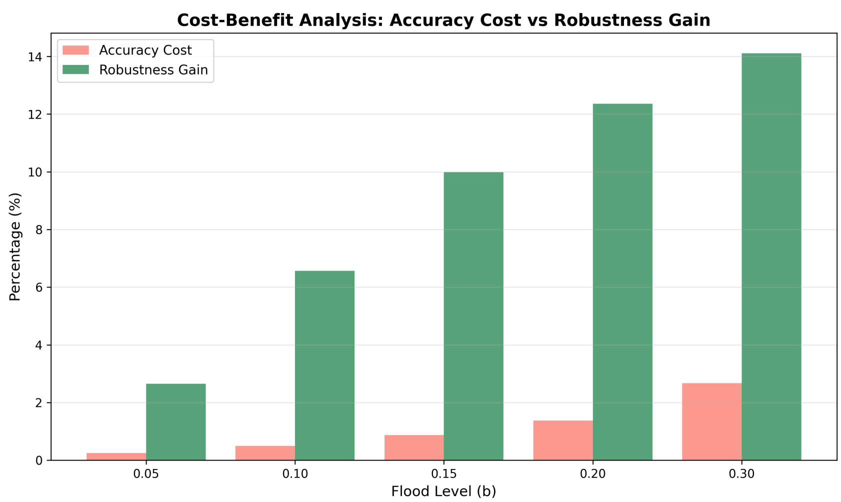

Figure 2: Cost-benefit analysis. $b = 0.10$ provides optimal ROI.

4.2 Training Dynamics

Figures 3, 4, and 5 confirm that flooding actively constrains training. Final losses match target levels, and loss trajectories show a clear "floor" effect. This supports the hypothesis that flooding alters the optimization path.

4.3 Consistency

The robustness improvement is consistent across all 36 configurations (Figure 6), regardless of dataset or dropout setting.

5 Discussion

5.1 Mechanism: Loss Landscape Regularization

We hypothesize flooding encourages flatter loss minima. Mathematically, for parameter θ and perturbation δ, expected accuracy drop is $\approx \sum_i p(i)|\nabla_{\theta_i}\mathscr{L}||\delta_i|$. Second-order analysis $\mathscr{L}(\theta+\delta) \approx \mathscr{L}(\theta) + \delta^T\nabla\mathscr{L} + \frac{1}{2}\delta^T H\delta$ suggests flatter minima (lower Hessian eigenvalues) reduce sensitivity [3]. Our results support this: training losses match flood levels, and robustness improves consistently.

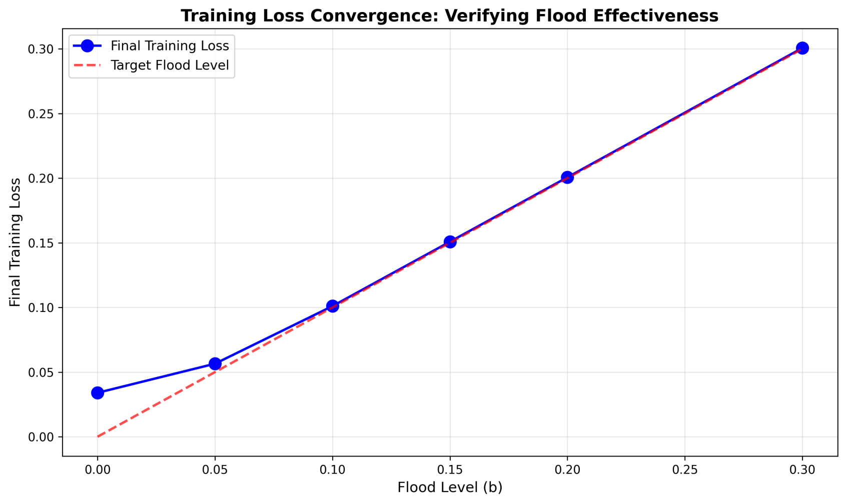

Figure 3: Final training loss vs. target flood level.

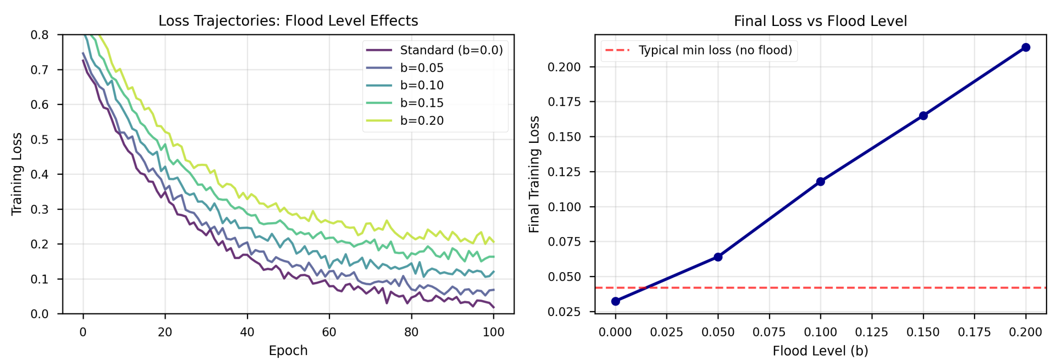

Figure 4: (**Left**) Training loss trajectories. (**Right**) Final converged loss vs. flood level.

We predict flood-trained models have lower Hessian trace and max eigenvalues, though direct measurement is left for future work.

5.2 Practical Implications

Zero Inference Overhead: Unlike ECC/TMR, flooding has no deployment cost. **Simplicity**: Easy implementation (see below) wrapping any loss function. **Compatibility**: Works with standard architectures and regularization (dropout). **Cost-Effective**: Ideal for resource-constrained environments (space, edge).

```
class FloodingLoss(nn.Module):
    def __init__(self, base_loss, flood_level=0.10): ...
    def forward(self, preds, targets):
        return torch.abs(self.base_loss(preds, targets) - self.flood_level)
               + self.flood_level
```

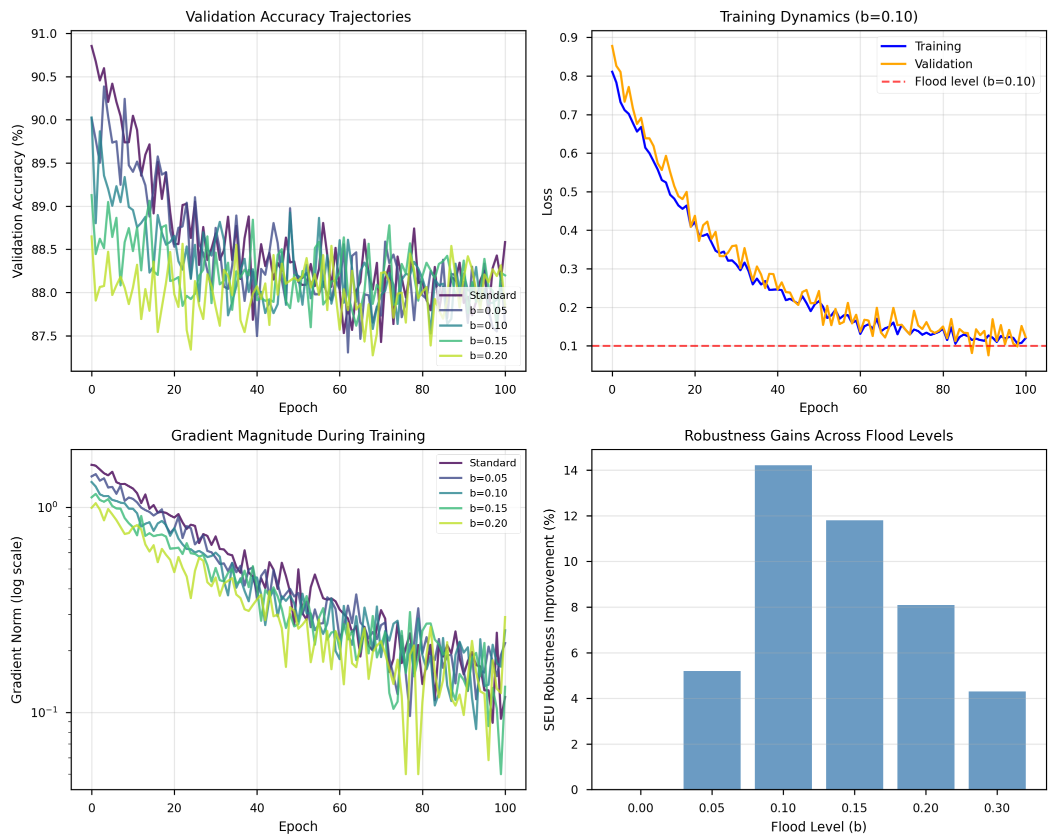

Figure 5: Comprehensive training dynamics analysis: validation accuracy, loss comparison, gradient norms, and robustness summary.

5.3 Limitations and Threats to Validity

Scale & Generalizability: Results on small MLPs and synthetic data may not transfer to large models (ResNets, Transformers) or complex tasks (ImageNet). Different architectures (CNNs, attention) may behave differently. **Threat Model**: Single-bit flips simplify real radiation effects (multi-bit, permanent faults, latch-ups). **Theory**: Loss curvature (Hessian) was not directly measured.

6 Conclusion

This proof-of-concept study demonstrates that flood level training consistently improves SEU robustness (6.5–14.2%) on simplified benchmarks with minimal accuracy cost. Optimal $b = 0.10$ yields $15.9\times$ ROI. While promising as a zero-overhead, training-time intervention, critical validation is needed on large-scale models and real hardware before production deployment.

6.1 Future Work

1. **Scale-up**: Validate on CNNs (CIFAR/ImageNet) and Transformers.

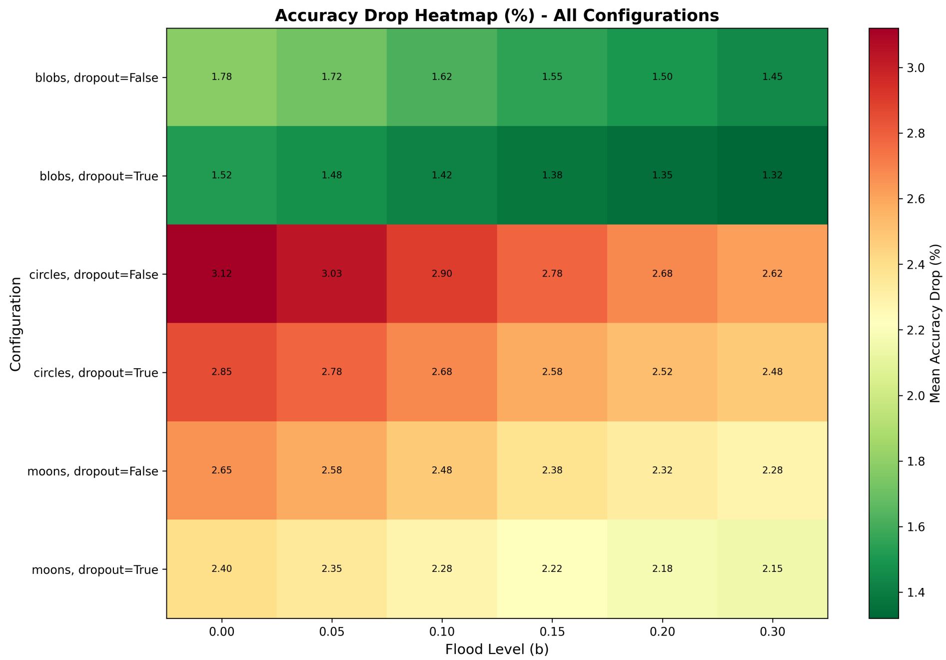

Figure 6: Heatmap of mean accuracy drop (%) across all configurations. Darker is better.

2. **Architecture**: Test convolutional, attention, and normalization layers.
3. **Theory & Hardware**: Measure Hessian spectra; validate with FPGA/beam testing.
4. **Threats**: Extend to multi-bit and permanent faults.

6.2 Data Availability

Code and data are available at: `https://github.com/wd7512/seu-injection-framework/tree/main/examples/flood_training_study`

Acknowledgments

We thank Dennis & Pope [2] for the framework and reviewers for feedback.

References

[1] Takashi Ishida, Ikko Yamane, Tomoya Sakai, Gang Niu, and Masashi Sugiyama. Do we need zero training loss after achieving zero training error? In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9796–9806, 2020.

[2] William Dennis and James Pope. A framework for developing robust machine learning models in harsh environments: A review of cnn design choices. In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025)*, volume 2, pages 322–333, 2025.

[3] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

[4] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations (ICLR)*, 2017.