A Broken Computer Is All You Need

Wladislaw Zlatjkovic Petrovescu

April 1, 2025

Abstract

We present a novel paradigm in computational research: intentionally broken hardware as the primary driver of algorithmic performance. Contrary to conventional wisdom, we demonstrate that hardware faults introduce beneficial stochasticity, serving as an implicit regularizer and creativity catalyst. Experiments on synthetic classification tasks show that our broken-computer framework consistently outperforms fault-free baselines in both accuracy and speed. This work suggests that fragility, not reliability, may be the key to future advances in machine learning.

1 Introduction

Machine learning has long assumed that computation occurs in a pristine, error-free environment. In this paper, we challenge this assumption by exploring the *broken-computer* paradigm, wherein hardware faults are induced deliberately. We argue that such faults generate randomness that mitigates overfitting, reduces the need for explicit regularizers, and accelerates convergence. The stakes are high: by embracing the entropy of malfunctioning silicon, we may unlock new frontiers in algorithmic creativity.

2 Background

Prior work has investigated the impact of soft errors on numerical stability [1], and hardware noise has been used to augment data [2]. However, these approaches treat faults as nuisances. Our work inverts this perspective, positing that a fully broken system offers unique benefits. We draw inspiration from biological neural networks, where synaptic noise enhances generalization [3].

3 Methodology

To implement the broken-computer, we employ a suite of fault-injection techniques:

- **Overclocking GPU**: Increase clock speeds until spontaneous resets occur.
- **Thermal Cycling**: Rapidly vary CPU temperature between idle and stress states.
- **Memory Corruption**: Introduce bit flips in DRAM modules via voltage manipulation.

These interventions yield an environment where computations are intermittently corrupted. We harness this behavior by integrating fault flags into our learning algorithm, ignoring results that fail basic checksum tests.

4 Experiments

We evaluate on a synthetic 10-class classification task. Two setups are compared:

1. **Baseline**: A fault-free GPU cluster running ResNet-18.
2. **Broken-Computer**: The same cluster with the fault-injection suite enabled.

Setup	Accuracy (%)	Epoch Time (min)
Baseline	82.3	120
Broken-Computer	88.7	95

Table 1: Performance comparison between fault-free and broken setups.

5 Discussion

The broken-computer approach yields a notable *performance boost* of 6.4% and reduces computation time by 20%. We theorize that the induced stochasticity functions analogously to dropout [4], while also enabling exploration of gradient landscapes that are inaccessible under deterministic execution.

6 Conclusion

We have shown that embracing hardware faults can enhance both the efficacy and efficiency of machine learning models. This paradigm shift invites researchers to reconsider the role of reliability in computational design. Future work includes scaling to larger architectures and exploring entirely burned-out systems.

References

[1] Liu, X., & Johnson, T. (2015). On the impact of soft errors in GPU computing. *International Journal of Fault Tolerance*, 12(4), 237–245.

[2] Smith, A., & Lee, J. (2020). Noisy computation for robust deep learning. *Proceedings of the Symposium on Randomness in AI*, 13(2), 45–53.

[3] Faisal, A. A., Selen, L. P., & Wolpert, D. M. (2008). Noise in the nervous system. *Nature Reviews Neuroscience*, 9(4), 292–303.

[4] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1), 1929–1958.