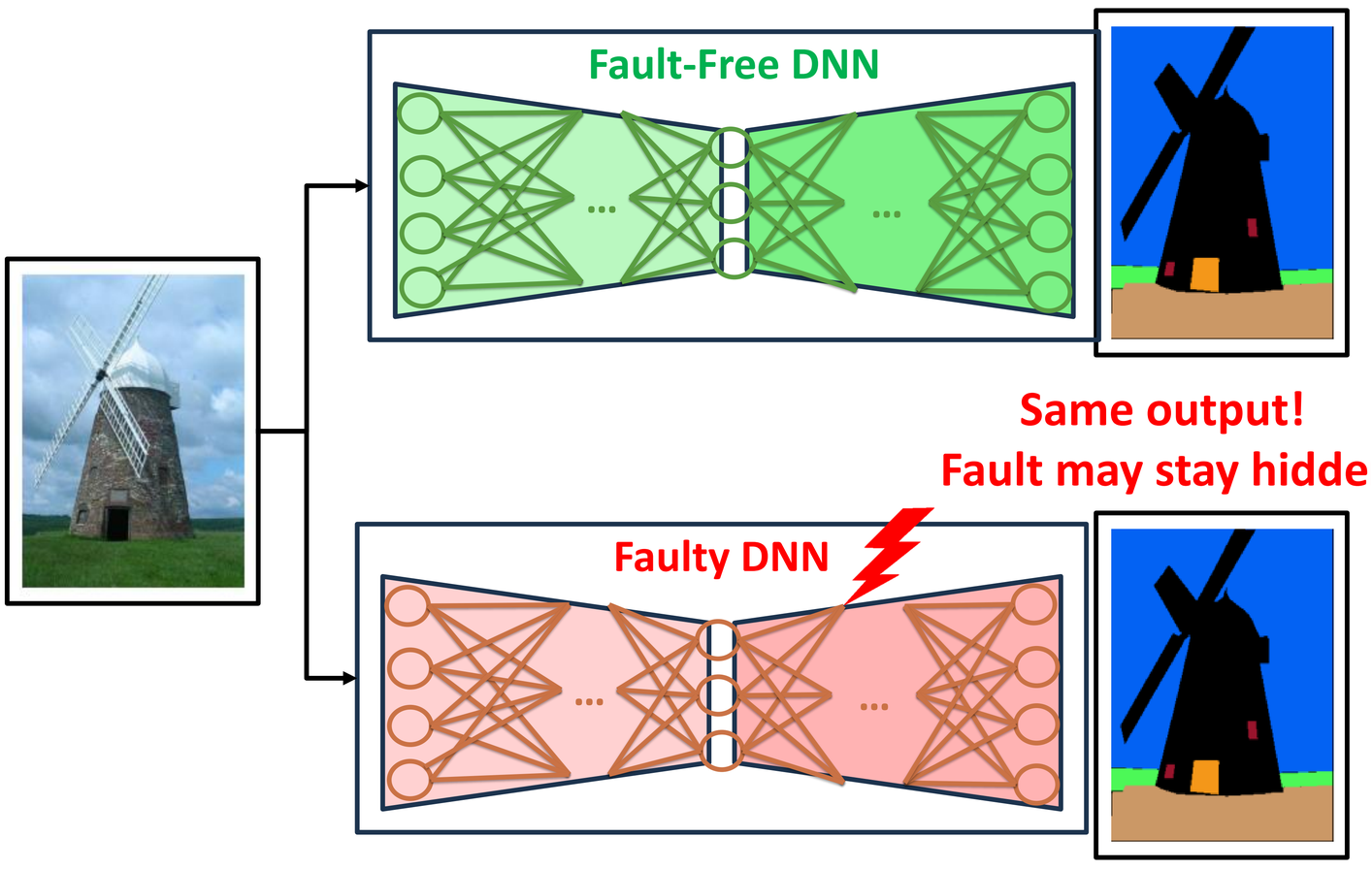
Detecting and Localizing Hardware Faults in Neural Networks and Language Models

Karlsruher Institut für Technologie



Neural networks and language models do not only exist as software. During inference, they run on physical hardware such as GPUs, TPUs, memories, and AI accelerators. These hardware components can develop faults. For example, a bit storing a model weight may flip, an activation value may be corrupted, or a processing element may get stuck at a fixed value.

The difficult part is that the model may not crash. It may still produce an output, but the output can silently become wrong. For a user, this can look like a normal model mistake, while the real reason is a hardware fault.

Recent work has shown that neural networks and language models can be silently affected by hardware faults and bit-flip attacks. Some studies analyze transient hardware faults in LLMs, while others show that targeted bit-flips in model parameters can strongly degrade performance or bypass model safety restrictions. However, there is still no standard way to generate diagnostic tests for neural networks running on hardware and identify where the fault is likely located.

Project Goal

This project asks two main questions.

1. Can we automatically generate input test patterns that reveal faulty behavior?
2. Can we localize the fault to a layer, neuron, weight group, or attention head?

In the testing literature, the first task is called automatic test pattern generation (ATPG), while the second is related to diagnostic test pattern generation (DTPG). ATPG has already been studied for deep neural networks, but diagnostic localization is still much less explored, especially for transformer-based models.

To build on existing work, the student will compare previous approaches for fault modeling and test generation in neural networks running on hardware accelerators. The project will first focus on CNN-based models and then move to a small transformer model..

Programming language
Python

Frameworks / tools
PyTorch, HuggingFace Transformers, PyTorchFI, bwUniCluster

Required skills
- Solid Python and PyTorch
- Basic deep learning (CNNs, transformers)
- Curiosity about hardware reliability, no prior background required

Group size
1 student preferred; up to 2 with task partitioning

Language
English

Method

The student will work entirely in software. Hardware faults will be simulated in PyTorch by injecting faults into weights, activations, or intermediate computations.

The student will build a fault-injection and evaluation pipeline to compare fault-free and faulty models. The first goal is fault detection: generating test inputs that reveal faulty behavior. The second goal is fault localization: checking whether the fault can be narrowed down to a layer, neuron, weight group, or attention head under a fixed test budget. The project may also explore gradient-based or Explainable AI (XAI) methods to support this localization.

[1] Moussa, Dina, et al. "Automatic test pattern generation and compaction for deep neural networks." Proceedings of the 28th Asia and South Pacific Design Automation Conference. 2023.
[2] Ahmed, Soyed Tuhin, and Mehdi B. Tahoori. "One-shot online testing of deep neural networks based on distribution shift detection." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 43.10 (2024): 3250-3263.
[3] Agarwal, Udit Kumar, Abraham Chan, and Karthik Pattabiraman. "Resilience assessment of large language models under transient hardware faults." 2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE). IEEE, 2023.
[4] Coalson, Zachary, et al. "Prisonbreak: Jailbreaking large language models with fewer than twenty-five targeted bit-flips." arXiv e-prints (2024): arXiv-2412.
[5] Liu, Wei-Kai, et al. "Runtime Fault Localization in Deep Neural Network Accelerators." ACM Transactions on Design Automation of Electronic Systems 31.1 (2025): 1-27.

Chair of Dependable Nano Computing (CDNC)

Contact: Tara Gheshlaghi, tara.gheshlaghi@kit.edu