A Proactive Reliability Metric for Detecting Failures in Language Model Training

This repository contains the source code and experimental scripts for the EMNLP 2025 Industry Track paper, "A Proactive Reliability Metric for Detecting Failures in Language Model Training." The project introduces and validates a dynamic reliability metric (R-Metric) for the early detection of failures in large language model training.

Overview

The goal of this work is to move beyond reactive fault tolerance (e.g., checkpointing) towards a proactive paradigm that can predict training instability before it becomes catastrophic. We introduce the R-Metric, a composite score that integrates signals from the hardware level (λ), training dynamics (σ^2), and model performance (ΔL) to provide a holistic assessment of training health.

This repository provides the scripts to:

Run a systematic simulation study (720 runs) across modern architectures (Llama-3, Mistral, GPT-4-MoE) to validate the R-Metric's performance.

Analyze the results and generate all tables and figures presented in the paper.

Reproduce the real-world case study on a Qwen-2.5 3B model to demonstrate the metric's practical application.

Repository Structure

`simulator.py` : The core simulation environment that models the LLM training process and can inject various faults.

`run_modern_experiments.py` : The main script to execute the full 720-run simulation study across all defined architectures and fault types.

`generate_paper_tables.py` : Analyzes the simulation logs to generate the final results tables (Table 1, 3, 4) for the paper, including the Isolation Forest baseline comparison.

`generate_all_figures.py` : Generates all figures (1-5) for the paper from simulation logs and conceptual designs.

`run_case_study.py` : The self-contained script to run the real-world validation case study using a Qwen-2.5 3B model and native PyTorch.

`replay_and_analyze_logs.py` : A utility script to analyze historical or existing `.jsonl` log files and generate case study plots.

`logs_FULL_METRIC/` : Directory containing log files from the simulation runs (sample provided).

`README.md` : This file.

`requirements.txt` : A list of all Python dependencies required to run the code.

How to Reproduce Results

To reproduce the findings in the paper, follow these steps in order:

Step 1: Install Dependencies

First, install all the required Python libraries from the `requirements.txt` file. It is recommended to use a virtual environment.

```
pip install -r requirements.txt
```

Note: The `graphviz` library also requires a system-level installation. On Debian/Ubuntu, run `sudo apt-get install graphviz`.

Step 2: Run the Simulation Study

Execute the main experiment script. This will generate the log files for all 720 runs and save them into subdirectories (e.g., `logs_FULL_METRIC/`).

```
python run_modern_experiments.py
```

(This is a long-running process and will take a significant amount of time to complete.)

Step 3: Generate Paper Tables

Once the simulations are complete, run the analysis script to generate the performance tables from the logs.

```
python generate_paper_tables.py
```

Step 4: Generate Paper Figures

Run the figure generation script to create all figures used in the paper.

```
python generate_all_figures.py
```

Step 5: Run the Real-World Case Study

Execute the case study script to validate the R-Metric on a real model fine-tuning task. This script is self-contained and will produce its own results CSV and plot.

```
python run_case_study.py
```

Citation

If you use this work, please cite our paper:

```
@inproceedings{anonymous2025proactive,
  title={A Proactive Reliability Metric for Detecting Failures in Language Model Training},
  author={Anonymous},
  booktitle={Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track},
  year={2025},
  publisher={Association for Computational Linguistics}
}
```