RLE Thermal-Optimization Coupling Analysis

Scientific Instrument for AI Training Thermal Monitoring

Overview

This repository contains a scientific instrument for measuring thermal-optimization coupling in AI training workloads. The system monitors hardware thermal efficiency (RLE) and optimization dynamics (gradient norms) to predict thermal instability before collapse occurs.

Key Features

• **Real-time thermal monitoring** with 1Hz sampling
• **Synchronized optimization logging** with gradient norm tracking
• **Causal analysis** with lag timing validation
• **Reproducible experiments** with atomic session management
• **Scientific validation** with reproducibility analysis

System Requirements

Hardware

• **GPU**: NVIDIA GPU with NVML support
• **CPU**: Multi-core processor with WMI support
• **RAM**: 8GB minimum, 16GB recommended
• **Storage**: 1GB free space for session data

Software

• **OS**: Windows 10/11 (tested on Windows 10 build 22000)
• **Python**: 3.10+ (tested on Python 3.11)
• **GPU Driver**: Latest NVIDIA drivers with NVML support
• **Git**: For version tracking and reproducibility

Installation

1. **Clone repository**:

```bash
git clone https://github.com/Nemeca99/RLE.git
```

cd RLE/lab
```
2. **Create virtual environment**:
```bash
python -m venv venv
venv\Scripts\activate
```
3. **Install dependencies**:
```bash
pip install -r requirements.txt
```
4. **Verify installation**:
```bash
python -c "import nvidia_ml_py3; print('NVML available')"
python -c "import psutil; print('psutil available')"
```

Quick Start

Single Session Analysis
```bash
python run_joint_session.py --model distilgpt2 --duration 120 --output results/
```
Reproducibility Analysis
```bash
python analysis/reproducibility_analysis.py
```
Custom Monitoring
```bash
python monitoring/hardware_monitor_v2.py --mode both --duration 300 --realtime
```

Usage Examples

Basic Thermal-Optimization Coupling
```bash
# Run 2-minute synchronized session
```
```

python run_joint_session.py --model distilgpt2 --duration 120 --output thermal_analysis/

Analyze results

python analysis/simplified_timestamp_fix.py

```

#### Extended Validation

```bash

# Run multiple sessions for reproducibility

for i in {1..3}; do

python run_joint_session.py --model distilgpt2 --duration 90 --output validation_$i/

done

# Analyze reproducibility

python analysis/reproducibility_analysis.py

```

Custom Model Analysis

```bash

# Analyze Luna model training

python run_joint_session.py --model luna --duration 300 --output luna_analysis/

```

Output Files

Each session generates:

- `rle_data_[session_id].csv` - Thermal monitoring data
- `training_log_[session_id].json` - Optimization dynamics
- `analysis_[session_id].json` - Correlation analysis
- `report_[session_id].txt` - Session summary
```

Scientific Validation

The instrument has been validated with:

• **3 independent sessions** showing 66.7% causal consistency
• **Lag timing**: -0.7 ± 0.5 seconds (grad_norm leads RLE)
• **Correlation strength**: Variable (-0.655 to 0.681)
• **Reproducibility**: Scientific validity 25% (identifies improvement areas)

Troubleshooting

Common Issues

1. **NVML not found**: Update NVIDIA drivers
2. **WMI errors**: Run as administrator
3. **Permission denied**: Check file write permissions
4. **Import errors**: Verify virtual environment activation

Debug Mode

```bash
python run_joint_session.py --model distilgpt2 --duration 60 --output debug/ --ambient-temp 21.0
```

Citation

If you use this instrument in research, please cite:

```
RLE Thermal-Optimization Coupling Analysis
Scientific Instrument for AI Training Thermal Monitoring
https://github.com/Nemeca99/RLE
```

License

MIT License - see LICENSE file for details.

Contributing

This is a scientific instrument. Contributions should maintain reproducibility and scientific rigor. Please:

1. Test changes with multiple sessions
2. Update documentation
3. Maintain backward compatibility
4. Follow scientific validation protocols

Contact

For scientific collaboration or technical questions:

• Repository: https://github.com/Nemeca99/RLE
• Issues: Use GitHub issues for technical problems
• Research: Contact for academic collaboration

Portable Run (Windows)

Use `portable/` for a self-contained run on any machine:
• `portable/RUN_PORTABLE.bat` → creates local venv, installs deps, starts monitor + dashboard
• `portable/QUICK_TEST.bat` → hardware scan → 60s idle baseline → 120s test
• Hardware snapshot saved to `portable/hardware_snapshot.json`
• All CSVs/reports remain under `sessions/recent/`