# Towards Autonomous Infrastructure Learning: Behavioral Pattern-Driven Optimization for Sustainable and Socially Responsible ML Infrastructure

**Research Assistant Proposal – Data Science Institute**

**Submitted by:** Andrew Espira, Saint Peter's University

**Application Period:** Fall 2025

**Total Hours:** 165 over 11 weeks

---

## Research Hypothesis (40%)

### Core Research Problem

Current machine learning infrastructure scheduling methods assign tasks statically and lack the ability to adapt to real-time changes. Existing systems such as Gandiva, Optimus, and Tiresias allocate jobs reactively without leveraging historical behavioral data for predictive scheduling (Xiao et al., 2018; Peng et al., 2018; Gu et al., 2019). These limitations result in inefficient use of computing resources, increased operational costs, and reduced accessibility for institutions with limited resources.

### Primary Research Hypothesis

This research hypothesizes that autonomous infrastructure learning, facilitated by behavioral pattern recognition, enhances machine learning cluster resource utilization efficiency and reduces computational waste. This approach systematically applies Common Reliability Enumeration (CRE)-inspired optimization techniques to machine learning infrastructure management, thereby advancing sustainability and social responsibility objectives.

### Sustainability Impact Through Resource Efficiency

The proposed approach advances sustainability in computing by enabling proactive and intelligent resource management rather than reactive problem-solving. The system will analyze patterns in hardware performance data, such as component utilization, to inform automated resource allocation decisions. Resource utilization efficiency will serve as a proxy metric for sustainability, rather than direct energy consumption measurements.

**Measurable Sustainability Metrics:**

- GPU utilization efficiency improvements (measured via nvidia-smi SM utilization)
- Reduction in computational resource waste through better allocation
- Improved resource throughput per hardware unit
- Extended effective hardware lifecycle through better utilization

### Social Responsibility Through Cost-Efficient Infrastructure

Resource-efficient artificial intelligence infrastructure can lower computational cost barriers in machine learning research. Current allocation inefficiencies increase operational costs and restrict opportunities for smaller institutions and underserved communities. This research evaluates the impact of behavioral pattern-driven optimization on cost efficiency and accessibility.

**Measurable Social Impact Metrics:**

- Reduced computational cost per training job
- Improved accessibility to expensive GPU resources through efficiency gains
- Open-source framework enabling broader adoption of optimization techniques
- Reproducible methodology that smaller institutions can implement independently

### Technical Innovation

This research seeks to provide the first systematic application of behavioral pattern learning to ML infrastructure optimization. Existing monitoring frameworks, such as Netflix's Atlas, focus on data collection, and systems like Prequel's CRE framework emphasize reliability pattern detection. However, no prior work combines behavioral inference and autonomous optimization for ML workloads. The approach here extends CRE methodology from reliability detection to performance optimization, supporting the development of an autonomous learning framework for infrastructure management.

---

## Data Analysis and Methodology (30%)

### Dataset Development Strategy

**Primary Approach: Systematic Behavioral Data Collection**

Rather than relying on existing datasets, this research will create a novel behavioral analysis dataset through systematic collection and controlled experiments:

**Data Collection Framework:**

- **Temporal Coverage:** 6-month systematic collection period
- **Spatial Scope:** Multi-node GPU cluster environments (university and cloud-based)
- **Resolution:** Hardware performance counter data at 1-second intervals
- **Behavioral Dimensions:** User submission patterns, resource allocation efficiency, workload characteristics

**Dataset Components:**

- **Hardware Performance Metrics:** GPU SM utilization, memory bandwidth usage, cache performance

- **System Events:** Job submission patterns, resource allocation history, completion statistics
- **Temporal Patterns:** Training phase transitions, performance evolution analysis
- **Efficiency Correlations:** Resource request vs actual utilization patterns

## Advanced Neural Network Applications

Graph Neural Networks (GNNs) will model complex relationships between GPU cluster nodes, capturing how network topology affects behavioral patterns and optimization opportunities (Hamilton et al., 2017). This represents a novel application of GNNs to infrastructure optimization, enabling topology-aware behavioral pattern recognition.

**Pattern-to-Action Mapping:**

- GNN insights translate to concrete scheduling decisions (migration, priority adjustment, resource reallocation)
- Network-Aware Optimization: Topology analysis enables congestion-aware task placement
- Adaptive Resource Management: Dynamic adjustment based on detected communication patterns

**Selective Application of Advanced Techniques:**

- **Change Detection:** Statistical approaches to identify evolving behavioral patterns
- **Transfer Learning:** Few-shot learning for adapting models to new workload types
- **Distributed Sensing:** eBPF-based coordinated data collection across cluster nodes

## Core Methodology: CRE-Inspired Behavioral Pattern Recognition

**Phase 1: Monitoring Framework Extension with CRE Integration** Building on proven monitoring approaches from Netflix Atlas, we extend time-series data collection with CRE-inspired behavioral pattern recognition capabilities (Netflix Tech Blog, 2017). The Common Reliability Enumeration (CRE) framework demonstrates effective rule-based pattern detection using event sequencing. We adapt this from reliability problem detection to performance optimization:

- Hardware Performance Counter Collection: eBPF instrumentation following CRE's distributed sensing approach
- Event Sequence Analysis: Temporal correlation methodology adapted for optimization pattern detection
- Pattern Rule Development: ML infrastructure-specific CRE-style rules for performance optimization

**Phase 2: CRE-Extended Machine Learning Pipeline**

- **Event Sequencing Rules:** Define sequences like "GPU allocation → Low utilization <30% → Duration >5min" indicating resource inefficiency

- **Negative Conditions:** CRE-style conditions to reduce false positives during legitimate phases (compilation, data loading)
- **Community Rule Development:** Shareable optimization patterns following CRE methodology

**Machine Learning Enhancement:**

- **Classification Algorithms:** Random Forest and SVM for workload type identification from hardware signatures
- **Clustering Analysis:** DBSCAN for discovering novel behavioral patterns not captured in predefined rules
- **Temporal Pattern Recognition:** Time-series analysis extending CRE's correlation methods
- **Rule Validation:** ML validation of CRE-inspired optimization rules across diverse workloads

**Phase 3: Autonomous Optimization Engine**

- Pattern-to-Recommendation Mapping: Convert detected patterns to specific optimization actions
- Real-time Decision Engine: Sub-100ms optimization decisions based on pattern matching
- Continuous Learning: ML-driven improvement based on optimization outcomes
- Community Integration: Shareable optimization patterns following CRE schema

### Novel Contributions Through CRE-Extended Analysis

This work introduces a systematic application of CRE methodology for performance optimization, distinguishing it from previous uses in failure detection. The methodology enables identification of optimization opportunities beyond what is accessible to traditional monitoring techniques.

- **ML Infrastructure-Specific Patterns:** New optimization-focused pattern definitions
- **Latent Workload Interactions:** Rule sequences capturing concurrent job performance effects
- **Temporal Optimization Windows:** Event sequences identifying optimal resource allocation timing
- **Cross-Modal Correlation:** Rules linking hardware metrics, system logs, and user behavior

---

## Literature Review and Citations (15%)

### Monitoring Framework Foundations

Netflix's Atlas monitoring system demonstrates scalable time-series data collection, handling over one billion metrics per minute, establishing proven methodologies for large-scale performance monitoring (Netflix Tech Blog, 2017). The Prequel CRE (Common Reliability Enumeration) framework validates community-driven pattern detection using rule-based event sequencing for infrastructure analysis at production scale. This research extends established CRE methodologies from reliability detection to performance optimization.

### ML Infrastructure Scheduling Literature

**Existing ML Schedulers:**

- **Optimus (EuroSys'18):** Dynamic resource scheduler achieving significant improvements in job completion time through performance modeling (Peng et al., 2018)
- **Gandiva (OSDI'18):** GPU cluster scheduling with time-slicing and migration capabilities (Xiao et al., 2018)
- **Tiresias (NSDI'19):** GPU cluster manager using advanced scheduling algorithms (Gu et al., 2019)
- **Pollux (OSDI'21):** Co-adaptive scheduling optimizing goodput with demonstrated cost reductions (Qiao et al., 2021)

**Research Gap:** These systems optimize job placement reactively but lack behavioral pattern learning for predictive optimization.

### Performance Optimization Research

Recent work demonstrates machine learning applications for dynamic resource allocation in cloud environments. However, no existing work applies behavioral pattern recognition specifically to ML infrastructure optimization through CRE-inspired methodologies. Performance optimization research demonstrates significant improvements through hardware-aware algorithm selection (Snir et al., 1998). Recent GPU scheduling surveys demonstrate substantial optimization opportunities in ML workload management (Weng et al., 2023). Our research extends these approaches with dynamic behavioral pattern learning and autonomous optimization capabilities.

---

## Implementation Timeline and Resource Requirements

### 11-Week Implementation Plan (165 Hours Total)

#### Weeks 1-3 (45 hours): Foundation and Data Collection

- Comprehensive literature synthesis and methodology development
- eBPF-based performance counter collection system implementation
- Initial behavioral pattern dataset construction from controlled ML workloads
- Hardware performance counter correlation analysis and baseline establishment

#### Weeks 4-6 (45 hours): CRE-Inspired Behavioral Pattern Recognition Development

- CRE-style optimization rule development using YAML schema adaptation
- Machine learning pipeline implementation for pattern detection and validation
- Graph neural network development for cluster topology modeling
- Community-shareable optimization pattern creation following CRE methodology

- Statistical validation of pattern-optimization opportunity correlations

**Weeks 7-9 (45 hours): Autonomous Optimization Engine Development**

- Pattern-to-recommendation mapping algorithm implementation
- Real-time optimization decision system development
- Performance improvement validation on controlled ML workloads
- Continuous learning mechanism implementation for system improvement

**Weeks 10-11 (30 hours): Results Documentation and Academic Presentation**

- Academic symposium presentation preparation and delivery
- Open-source implementation documentation and community release
- Research findings documentation for publication submission
- Performance improvement analysis with statistical significance validation

## Resource Requirements

**Computing Infrastructure:**

- University GPU cluster access and cloud computing credits via academic programs
- MLCommons benchmark datasets for reproducible validation and comparison
- Controlled experimental environments for baseline measurement and validation

**Software Development Tools:**

- eBPF development framework (BCC, libbpf) for kernel-level instrumentation
- Python ML ecosystem (scikit-learn, PyTorch) for behavioral analysis implementation
- OpenTelemetry and Prometheus integration for monitoring framework extension

## Expected Deliverables

**Technical Contributions:**

- Working autonomous optimization system demonstrating CRE-inspired behavioral pattern recognition
- Community-shareable optimization patterns using CRE methodology extension
- Open-source implementation integrating CRE framework with ML infrastructure optimization
- Performance benchmark results demonstrating resource efficiency improvements

**Academic Impact:**

- Academic symposium presentation showcasing autonomous infrastructure learning methodology
- Workshop paper documenting approach, methodology, and experimental results

- Reproducible research framework and dataset for community validation and extension

**Sustainability and Social Impact:**

- Quantified resource utilization efficiency improvements through controlled experiments
- Cost-efficiency analysis demonstrating potential for broader research accessibility
- Open-source framework enabling adoption by resource-constrained institutions

---

## References

### Core ML Infrastructure Papers

Gu, J., Chowdhury, M., Shin, K. G., Zhu, Y., Jeon, M., Qian, J., Liu, H., & Guo, C. (2019). Tiresias: A GPU cluster manager for distributed deep learning. *Proceedings of the 16th USENIX Symposium on Networked Systems Design and Implementation (NSDI '19)*, 485-500.

Peng, Y., Bao, Y., Chen, Y., Wu, C., & Guo, C. (2018). Optimus: An efficient dynamic resource scheduler for deep learning clusters. *Proceedings of the 13th EuroSys Conference*, 3:1-3:14.

Qiao, A., Choe, S. K., Subramanya, S. J., Neiswanger, W., Ho, Q., Zhang, H., Ganger, G. R., & Xing, E. P. (2021). Pollux: Co-adaptive cluster scheduling for goodput-optimized deep learning. *Proceedings of the 15th USENIX Symposium on Operating Systems Design and Implementation (OSDI '21)*, 1-18.

Xiao, W., Bhardwaj, R., Ramjee, R., Sivathanu, M., Kwatra, N., Han, Z., Patel, P., Peng, X., Zhao, H., Zhang, Q., Yang, F., & Zhou, L. (2018). Gandiva: Introspective cluster scheduling for deep learning. *Proceedings of the 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI '18)*, 595-610.

### Technical and Framework Papers

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS '17)*, 1024-1034.

Snir, M., Otto, S., Huss-Lederman, S., Walker, D., & Dongarra, J. (1998). *MPI-The Complete Reference, Volume 1: The MPI Core*. MIT Press.

### Recent GPU Scheduling Research

Jeon, M., Venkataraman, S., Phanishayee, A., Qian, J., Xiao, W., & Yang, F. (2019). Analysis of large-scale multi-tenant GPU clusters for DNN training workloads. *Proceedings of the 2019 USENIX Annual Technical Conference (USENIX ATC '19)*, 947-960.

Weng, Q., Yang, L., Yu, Y., Wang, W., Tang, X., Yang, G., & Zhang, L. (2023). Beware of fragmentation: Scheduling GPU-sharing workloads with fragmentation gradient descent. *Proceedings of the 2023*

*USENIX Annual Technical Conference*, 995-1008.

### Industry/Technical References

Gregg, B. (2018, February 26). Netflix FlameScope: A Performance Visualization Tool. *Netflix Tech Blog*. Retrieved from https://netflixtechblog.com/netflix-flamescope-a57ca19d47bb

Netflix Tech Blog. (2017, March 2). Introducing Atlas: Netflix's Primary Telemetry Platform. *Netflix Technology Blog*. Retrieved from https://netflixtechblog.com/introducing-atlas-netflixs-primary-telemetry-platform-bd31f4d8ed9a

Prequel. (2024). Common Reliability Enumerations: Community-Driven Problem Detection. Retrieved from https://docs.prequel.dev/