Production LLM Deployment: Risk Characterization Before Failure

Course Syllabus

Instructor: Javier Marín
Independent Researcher — Applied AI Consultant

November 23, 2025

Course Overview

Organizations deploy large language models in production assuming benchmarks predict real-world performance. This assumption causes catastrophic failures. This course teaches systematic methodology for characterizing architectural limitations before deployment—distinguishing genuine capability from brittle pattern matching, identifying failure modes through rigorous experimentation, and designing hybrid systems when pure LLMs fundamentally cannot work.

What This Course Is: Systematic risk characterization methodology for production LLM deployment in high-stakes applications.

What This Course Is Not: Prompt engineering recipes, fine-tuning tutorials, or methods to improve LLM architectures. We document what current approaches cannot reliably do.

Target Audience

- ML Engineers deploying LLMs in medical, financial, or emergency response systems
- Technical Leaders making architecture decisions for high-stakes applications
- AI Safety Researchers characterizing failure modes systematically
- Consultants advising organizations on LLM deployment risks

Prerequisites

- Production ML experience (deployed at least one model)
- Python programming (can read and adapt experimental code)
- Basic statistics (hypothesis testing, p-values, confidence intervals)
- Familiarity with LLM APIs and prompt engineering

Course Format

- **Duration:** 6 weeks
- **Live Sessions:** 12 sessions (2 per week), 90 minutes each
- **Office Hours:** Weekly 60-minute sessions for implementation support
- **Asynchronous:** Recitation documents, experimental code repositories, reading materials

Learning Outcomes

By course completion, students will:

1. **Classify capabilities** into what LLMs reliably do vs what fails predictably
2. **Design experiments** revealing architectural limitations before production deployment
3. **Perform statistical analysis** distinguishing genuine capability from spurious correlations
4. **Identify failure modes** through systematic testing protocols

5. **Specify hybrid architectures** combining LLMs with symbolic reasoning when necessary

6. **Document risks** for regulatory review and stakeholder communication

Course Philosophy

This course embodies scientific pragmatism. We distinguish between:

- **Useful infrastructure work** that enables reliable deployment
- **Fundamental architectural limitations** requiring different approaches
- **Incremental optimization** vs breakthrough architectural changes

We provide consistent, rigorous assessment without confusing motivational messaging. When current architectures fundamentally cannot solve a problem, we state this clearly and teach hybrid approaches.

Detailed Session Plan

Session 1: Capability Taxonomy—What LLMs Can and Cannot Reliably Do

Session Objective

Establish framework for classifying tasks into what current LLM architectures handle reliably vs what fails predictably. Students learn to assess their deployment scenario before investing engineering resources.

Content:

- Capability classes: pattern completion, continuous state representation, compositional reasoning, constraint satisfaction, knowledge retrieval, interactive collaboration
- Architectural prerequisites for each class
- Case studies demonstrating each failure mode
- Decision framework: "Can my task be solved with current LLM architectures?"

Practical Exercise: Students classify their deployment scenario into capability classes and identify which require testing or hybrid architectures.

Deliverable

Capability classification matrix for your deployment domain

Recitation: "Taxonomy of LLM Capabilities: Pattern Matching vs Systematic Reasoning"

Session 2: Architectural Prerequisites for Reliable Performance

Session Objective

Understand why next-token prediction on discrete tokens succeeds for certain tasks but fundamentally fails for others. Students learn to recognize when their task requires architectural mechanisms LLMs lack.

Content:

- Discrete vs continuous representations: when tokens work vs fail
- Pattern matching vs computational processes
- Biological systems as existence proofs (hippocampal time cells, cerebellar forward models)
- Compositional generalization: Lake & Baroni framework

Practical Exercise: Analyze three deployment scenarios—identify architectural prerequisites and whether LLMs provide them.

Deliverable

Architectural analysis document for your application

Recitation: "Why Next-Token Prediction Fails: Discrete Tokens and Continuous Phenomena"

Session 3: Experimental Design I—Constructing Diagnostic Scenarios

Session Objective

Learn to design test scenarios that reveal production failure modes before deployment. Students create scenario sets for their domains.

Content:

- Balancing test distributions (preventing response bias)
- Controlling semantic content while varying format (brittleness tests)
- Case study: Temporal constraint testing across 8 models
- Ground truth establishment and deterministic evaluation

Practical Exercise: Design 8-12 test scenarios for your deployment domain following experimental protocols.

Deliverable

Test scenario set with ground truth labels and evaluation protocol

Recitation: "Designing Experiments That Reveal Architectural Limitations"

Session 4: Experimental Design II—Statistical Analysis and Causal Testing

Session Objective

Perform rigorous statistical analysis distinguishing genuine capability from spurious patterns. Students learn to detect bimodal distributions, quantify brittleness, and conduct interventions.

Content:

- Detecting bimodal distributions (capability vs failure clustering)
- Quantifying prompt brittleness (percentage point changes across formats)
- Measuring false positive/negative bias rates
- Causal interventions: attention manipulation, ablation studies

Practical Exercise: Run experimental protocol on 2-3 models, perform statistical analysis, document failure patterns.

Deliverable

Experimental results with statistical analysis and failure characterization

Recitation: "Statistical Methods for Failure Mode Detection"

Session 5: Failure Mode I—Temporal Constraint Processing

Session Objective

Deep dive into continuous state representation failures. Students understand why discrete tokens cannot reliably handle temporal reasoning and when hybrid architectures are required.

Content:

- Bimodal performance distribution (3.8B matches 7B; other 7B fails completely)
- Extreme prompt brittleness (62.5pp accuracy drops from format changes)
- Systematic action bias (100% false positive rates)
- Allen's interval algebra and temporal constraint satisfaction

Practical Exercise: Test your model on temporal scenarios, measure brittleness, detect action bias.

Deliverable

Temporal constraint test results for your deployment scenario

Recitation: "Temporal Constraint Processing: Why Discrete Tokens Fail for Continuous State"

Session 6: Failure Mode II—Knowledge Scaling Pathology

Session Objective

Understand confidence-competence gap: why models become more confident in wrong answers as they scale. Students learn when scaling won't help and alternatives are needed.

Content:

- Loss improves 31% while accuracy stays flat (below random chance)
- Confidence-competence gap quantified: $\text{CCG} = \frac{\partial L/\partial N}{\partial A/\partial N} \approx -48$
- Attention interventions causing catastrophic degradation
- Recommended model sizes by task type

Practical Exercise: Evaluate whether your task exhibits scaling pathology. Calculate confidence-competence gap for your models.

Deliverable

Scaling analysis documenting whether larger models help your task

Recitation: "The Confidence-Competence Gap: Learning to Be Confidently Wrong"

Session 7: Failure Mode III—Prompt Brittleness and Pattern Matching

Session Objective

Systematic characterization of brittleness as diagnostic for pattern matching vs robust understanding. Students design multi-format tests revealing brittleness.

Content:

- Brittleness as architectural diagnostic (not fine-tuning problem)
- Designing semantically equivalent prompt variations
- Quantifying brittleness: absolute percentage point changes
- Case studies across domains: medical, financial, code generation

Practical Exercise: Create 3-5 prompt format variations for your task, measure brittleness, document pattern matching evidence.

Deliverable

Brittleness quantification report with multi-format test results

Recitation: "Brittleness as Diagnostic: When Robust Understanding Fails"

Session 8: Failure Mode IV—Response Latency and Interaction Constraints

Session Objective

Characterize how response delays create fundamental bandwidth constraints independent of response quality. Students understand interaction failure modes.

Content:

- Bandwidth degradation from 7.43 to 3.28 turns/minute (56% drop at 10s latency)
- Consistent across cognitive domains (effect sizes: Cohen's d up to 5.973)
- Pure additive delay—not adaptation or strategic adjustment
- Human cognitive architecture expectations for interactive reasoning

Practical Exercise: Measure interaction efficiency for your collaborative application. Quantify latency impact on task completion.

Deliverable

Latency impact analysis for your interactive system

Recitation: "Response Latency as Bandwidth Constraint: Interaction Failure Modes"

Session 9: Comprehensive Failure Mode Catalog

Session Objective

Complete taxonomy of failure modes with detection protocols. Students build comprehensive testing suite for their deployment.

Content:

- Brittleness failures: detection via multi-format testing
- Bias failures: false positive/negative rate measurement
- Scaling pathology: loss-accuracy divergence analysis
- Interaction constraints: latency-bandwidth measurement
- Hallucination under uncertainty: confidence calibration testing

Practical Exercise: Develop complete testing protocol covering all relevant failure modes for your application.

Deliverable

Comprehensive pre-deployment testing suite

Recitation: "Complete Failure Mode Catalog with Detection Protocols"

Session 10: Hybrid Architecture Design I—LLM + Symbolic Reasoning

Session Objective

Design hybrid systems combining LLMs with explicit symbolic reasoning modules. Students learn when pure LLMs fundamentally won't work and what to build instead.

Content:

- Decision framework: when hybrid architectures are necessary
- LLM + temporal constraint checker (medical triage example)
- LLM + explicit verification modules (financial compliance)
- Allen's interval algebra implementation for temporal reasoning

Practical Exercise: Specify hybrid architecture for your application. Define LLM component responsibilities vs symbolic module responsibilities.

Deliverable

Hybrid architecture specification document

Recitation: "Hybrid Architectures: LLM + Symbolic Reasoning Modules"

Session 11: Hybrid Architecture Design II—Integration Patterns

Session Objective

Implement integration patterns between LLM and symbolic components. Students build working prototypes demonstrating hybrid approaches.

Content:

- LLM + retrieval-augmented generation (knowledge task failures)
- LLM + explicit constraint propagation (temporal/logical reasoning)
- LLM + verification modules (safety-critical applications)
- Testing hybrid systems: validation that combination solves the problem

Practical Exercise: Implement prototype hybrid system. Validate that it addresses architectural limitations identified in testing.

Deliverable

Working prototype demonstrating hybrid architecture

Recitation: "Integration Patterns: Building Hybrid Systems That Work"

Session 12: Risk Documentation and Production Deployment

Session Objective

Document architectural limitations, failure modes, and hybrid solutions for regulatory review and organizational decision-making. Students produce deployment-ready documentation.

Content:

- Capability classification documentation
- Failure mode quantification (false positive rates, brittleness metrics)
- Statistical significance and confidence intervals
- Hybrid architecture justification
- Production monitoring protocols

Practical Exercise: Create complete deployment documentation package suitable for regulatory review.

Deliverable

Production deployment documentation with risk assessment and monitoring plan

Recitation: "Risk Documentation for Deployment Decisions"

Course Materials

Provided Resources

- **Recitation Documents:** 12 detailed technical documents (4 pages each) covering session content
- **Experimental Code:** Python repositories with replication code for all case studies
- **Test Scenarios:** Template scenarios for temporal constraints, knowledge tasks, interaction testing
- **Statistical Toolkit:** Analysis scripts for brittleness, bias, scaling pathology detection
- **Documentation Templates:** Regulatory-ready templates for risk assessment

Required Reading

- Marín, J. (2025). "Empirical Characterization of Temporal Constraint Processing in LLMs." *arXiv preprint*.
- Marín, J. (2025). "The Confidence-Competence Gap in Language Model Scaling." *arXiv preprint*.
- Lake, B. M., & Baroni, M. (2018). "Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks."
- Allen, J. F. (1983). "Maintaining knowledge about temporal intervals." *Communications of the ACM*.

Recommended Reading

- Wei, J., et al. (2022). "Emergent Abilities of Large Language Models."
- Schaeffer, R., et al. (2023). "Are Emergent Abilities of Large Language Models a Mirage?"
- Kaplan, J., et al. (2020). "Scaling Laws for Neural Language Models."
- Shanahan, M. (1997). *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia.*

Assessment and Deliverables

Weekly Deliverables

Students submit 6 major deliverables throughout the course:

1. Capability classification matrix for deployment domain
2. Test scenario set with experimental protocol
3. Experimental results with statistical analysis
4. Brittleness quantification report

5. Hybrid architecture specification

6. Production deployment documentation package

Final Project

Complete risk characterization for your production deployment:

- Comprehensive failure mode testing results
- Hybrid architecture design (if necessary)
- Regulatory documentation
- Production monitoring plan

Instructor Background

Javier Marín, PhD, bridges theoretical rigor and production engineering. His research characterizes systematic architectural limitations in LLM deployment through peer-reviewed empirical studies. He has consulted for organizations deploying AI in regulated environments (finance, healthcare), focusing on failure prevention through systematic pre-deployment testing.

Research:

- Temporal constraint processing failures (bimodal distributions, prompt brittleness)
- Knowledge scaling pathology (confidence-competence gap quantification)
- Response latency bandwidth constraints (collaboration efficiency degradation)

Teaching Philosophy: Scientific pragmatism without confusing motivational messaging. When current architectures fundamentally cannot solve problems, we acknowledge this and teach systematic approaches to hybrid solutions.

Logistics

Price: $2,800

Format: Live cohort with maximum 25 students (essential for hands-on support)
Schedule: 12 live sessions over 6 weeks (2 per week, 90 minutes each)
Office Hours: Weekly 60-minute sessions for implementation troubleshooting
Community: Private Slack workspace (lifetime access), monthly alumni meetups
Materials: All recordings, code repositories, and documentation templates provided