**LITERATURE REVIEW**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Author(s) & Year** | **Focus of Study** | **Key**  **Contributions** | **Limitations / Gaps** | **Relevance to Our Project** | **Links** |
| **Basiri, A. et al. (2016)** | **Chaos Engineering principles and methodology at Netflix** | Established the four foundational principles of chaos engineering: steady-state behavior, real-world events, production experiments, continuous automation | **Focused on general distributed systems; doesn't address AI-specific failure modes like rate limiting, authentication, or partial responses** | **PRIMARY BASE PAPER - Provides the theoretical foundation that we extend to AI services; our project is direct application of their methodology** | [**https://arxiv.org/abs/1702.05843**](https://arxiv.org/abs/1702.05843) |
| **Yuan, D. et al. (2014)** | **Analysis of production failures in distributed systems** | **Found that 92% of catastrophic failures result from incorrect handling of non-fatal errors; simple testing can prevent most critical failures** | **Analysis-focused rather than providing systematic testing methodology; doesn't address AI service failure patterns** | **Validates the importance of systematic failure testing; supports our approach of proactive failure injection for AI services** | [**https://www.usenix.org/system/files/conference/osdi14/osdi14-paper-yuan.pdf**](https://www.usenix.org/system/files/conference/osdi14/osdi14-paper-yuan.pdf) |
| **Richardson, C. (2018)** | **Microservices patterns including circuit breaker implementations** | **Documented circuit breaker pattern for microservices with implementation guidance and best practices** | **General microservices patterns; doesn't address AI service-specific considerations like token management or provider-specific errors** | **Direct relevance to our circuit breaker implementation; provides proven patterns that we adapt for AI services** | [**https://www.usenix.org/system/files/conference/osdi14/osdi14-paper-yuan.pdf**](https://www.usenix.org/system/files/conference/osdi14/osdi14-paper-yuan.pdf) |
| **Breck, E. et al. (2017)** | **ML Test Score for production ML systems** | **Introduced systematic testing framework for ML systems with scoring methodology for production readiness** | **Focused on ML model testing rather than AI service infrastructure; doesn't address multi-provider resilience or failure injection** | **Provides framework for systematic ML system assessment; our readiness scoring extends this concept to AI service resilience** | [**https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/aad9f93b86b7addfea4c419b9100c6cdd26cacea.pdf**](https://static.googleusercontent.com/media/research.google.com/en/pubs/archive/aad9f93b86b7addfea4c419b9100c6cdd26cacea.pdf) |
| **Crankshaw, D. et al. (2017)** | **Low-latency prediction serving systems (Clipper)** | **Designed serving system for ML models with batching, caching, and adaptive selection; focused on performance optimization** | **Primarily performance-focused; limited discussion of failure handling, resilience testing, or chaos engineering approaches** | **Provides architectural insights for AI service serving; complements our resilience focus with performance considerations** | [**https://www.usenix.org/system/files/conference/nsdi17/nsdi17-crankshaw.pdf**](https://www.usenix.org/system/files/conference/nsdi17/nsdi17-crankshaw.pdf) |