Contents

ADR-001: Replace LiteLLM with vLLM + Ray Serve

Status

Accepted

Context

LiteLLM has a hard limit of ~504 concurrent requests with its load balancer. At 10MM users generating ~100M queries/day (~5,000 QPS peak), this is catastrophically insufficient. LiteLLM is designed as a stateless API proxy for multi-tenant abstraction, not a stateful orchestration layer for a single global consciousness.

Cato requires: - **Stateful conversation context** across millions of concurrent sessions - **Hidden state extraction** for Shadow Self verification - **Custom routing logic** based on query type, cost, and consciousness state - **Circuit breaker patterns** for graceful degradation - **Horizontal scaling** to 10MM+ users

LiteLLM cannot provide any of these capabilities at the required scale.

Decision

Replace LiteLLM with a hybrid orchestration architecture:

1. vLLM on SageMaker (Self-Hosted Inference)

- Instance: **ml.g5.2xlarge** (24GB VRAM, ~$1.52/hour)
- Purpose: Llama-3-8B for Shadow Self with hidden state extraction
- Scaling: 10-300 instances based on load (auto-scaling)

- Features: `output_hidden_states=True` for activation probing

2. Ray Serve on EKS (Stateful Orchestration)

- Deployment: EKS with Karpenter for auto-scaling
- Purpose: Model routing, context management, fan-out coordination
- Features:
  - Actor-based stateful context (conversation history per session)
  - Circuit breaker with fallback chain
  - Semantic cache integration
  - Cost-aware routing

3. AWS Bedrock (Managed Claude Models)

- Models: Claude 3.5 Sonnet (complex), Claude 3 Haiku (simple)
- Features: Prompt caching (90% token discount on cache hits)
- Batch API: 50% discount for night-mode curiosity processing

Architecture

```
User Query


  Ray Serve Orchestrator (EKS)
    Semantic Cache Check
    Query Classification
    Model Selection
    Circuit Breaker
```

Consequences

Positive

- **Unlimited horizontal scaling**: No hard concurrency limits
- **Stateful context**: Actor-based conversation management
- **Hidden states**: Full access to Llama activations for Shadow Self
- **Cost optimization**: Semantic caching + batch processing
- **Graceful degradation**: Circuit breaker with fallback chain

Negative

- **16-week migration path**: Significant implementation effort
- **Operational complexity**: Managing EKS + SageMaker + Bedrock
- **Custom code**: ~5,000 LOC orchestration layer to maintain
- **Team expertise**: Requires Ray Serve and ML infrastructure knowledge

Cost Impact

Component	1M Users	10M Users
SageMaker (Shadow Self)	$13,000/mo	$130,000/mo
EKS (Ray Serve)	$2,000/mo	$15,000/mo
Bedrock (Claude)	$15,000/mo	$130,000/mo
Total Inference	**$30,000/mo**	**$275,000/mo**

Migration Path

Phase 1: Weeks 1-4

- Deploy vLLM on SageMaker with hidden state extraction
- Set up NLI model on SageMaker MME
- Create basic Ray Serve deployment

Phase 2: Weeks 5-8

- Implement model routing logic
- Add stateful context actors
- Integrate semantic cache

Phase 3: Weeks 9-12

- Connect to global memory infrastructure
- Implement circuit breaker patterns
- Load testing at scale

Phase 4: Weeks 13-16

- Gradual traffic migration (10% $\rightarrow$ 50% $\rightarrow$ 100%)
- Performance tuning
- Documentation finalization

References

- vLLM Documentation
- Ray Serve Documentation
- SageMaker Real-Time Inference
- AWS Bedrock