

With and Without Baltor: Controlled Multi-Domain Evaluations on Ollama Cloud, Honest Negative Results & Sealed Graph Verification

Taylor Amarel · Creator of TAEDRI & Baltor.ai · Research Portal: <https://research.baltor.ai> · Platform: <https://baltor.ai> · Directory: <https://index.baltor.ai>

Overview: A central question in AI evaluation is whether external harnesses, skill cards, and tool graphs genuinely improve model reliability—or merely add token overhead and brittle heuristics. Across **TAEDRI** (“This Already Exists, Do Not Remake It”) and **Baltor.ai**, we built a composable graph runtime where every functional unit has a typed, versioned contract edge, swappable deterministic or model engines, and an independent verification script (`verifier.py`). This monograph reports four controlled empirical studies comparing models **Without Baltor** (unassisted baseline) vs. **With Baltor**, including our full disclosure of negative and mixed results.

1. STUDY 1 (SEPT 22, 2026): 225-REQUEST DIRTY-DATA CLEANUP BENCHMARK & HONEST NEGATIVE RESULT

We executed **225 physical API requests** on Ollama Cloud across three models (`glm-5.3-flash` , `qwen3-coder:30b` , `gemma4:31b`) on messy enterprise entity/record normalization tasks, comparing unassisted prompting against an early injected skill card:

Model (Ollama Cloud)	Without Baltor (Raw Score)	With Early Unverified Skill Card	Net Delta	Root-Cause Diagnosis & Engineering Fix
<code>glm-5.3-flash</code>	0.991	0.842	-0.149 (Regression)	Honest Negative Finding: An incomplete phone-number regex heuristic in the early skill card stripped valid extension numbers that <code>glm-5.3-flash</code> handled correctly on its own.
<code>gemma4:31b</code>	0.884	0.946	+0.062 (Improvement)	Structured date and currency normalization rules eliminated format drift on ambiguous regional records.

Architectural Lesson Implemented: This negative finding proved that *unverified prompt/skill injection can actively harm capable models*. In response, we implemented mandatory **Pareto Non-Regression Gates**: no contract card or skill component is promoted or injected unless it passes a deterministic `verifier.py` suite and demonstrates zero regression against the unassisted model baseline.

2. STUDY 2 (OCT 16, 2026): PASSIVE MCP TOOL ABSTENTION PILOT (`0` VOLUNTARY CALLS)

When models were merely offered passive MCP discovery tools (`search_cards` , `load_verifier`) without pre-turn structural routing, they exhibited **100% tool abstention (0 voluntary tool calls)** on complex tasks that looked superficially familiar—confidently generating subtly broken code or missing cross-record constraints. This mirrors the *Bad Investigator Problem*: models do not know what they do not know when surface text looks routine.

3. STUDY 3 (OCT 16, 2026): PRE-TURN VERIFIED CONTRACT CARDS ON `GEMMA4:31B` (`0/8` → `7/8`)

We then evaluated `gemma4:31b` across **16 physical Ollama Cloud requests (8 paired multi-domain tasks)** comparing unassisted generation (Without Baltor) against pre-turn injection of compact, verified Contract Cards ($\leq 2,048$ bytes) paired with deterministic post-turn verification:

Domain Task Archetype	Without Baltor (<code>gemma4:31b</code>)	With Baltor Verified Contract Card	Key Failure Fixed by Contract Card + Verifier
1. Cross-Border Statutory Audit	FAIL (Missed Cap. 163 48% cap)	PASS (Verified)	Bound PHP fee novation to HK Cap. 163 & RA 10022
2. Probabilistic Entity Resolution	FAIL (O(N²) naive blocking)	PASS (Verified)	Enforced overlapping canopy blocking + transitive closure check
3. Financial Ledger Conservation	FAIL (Float rounding drift)	PASS (Verified)	Enforced integer minor-unit conservation invariant
4. Leak-Free Cross-Validation Split	FAIL (Temporal target leakage)	PASS (Verified)	Enforced grouped purged walk-forward split verification
5. Brownfield API Patch	FAIL (Broke caller signature)	PASS (Verified)	Preserved typed versioned edge contract & refusal schema