Guide to Using Large Multimodal Models v1.1

Technical Supplement 2 — Verification, Auditing & Troubleshooting

About This Supplement

Purpose & Scope
This supplement unifies **Verification, Auditing, and Troubleshooting** into a single quality system for LMMs. It supports analysts, auditors, and technical teams who must **pre-emptively build verifiable reasoning** (Part I) and **reactively diagnose and correct failures** (Part II), with integrated **governance and logging** (Part III).

Audience: Analysts, auditors, ML Ops, compliance/governance.
Prerequisites: Familiarity with **C.G.A.F.R.** (Core Guide § 2) and the **Verification Workflow** (Core Guide § 4).
Outcome: A repeatable process that produces **evidence-backed artifacts** suitable for internal review and external audit.

Key Objectives:

- Implement a multi-layered verification protocol to detect factual inaccuracies, biases, and security vulnerabilities in model outputs.
- Integrate automated and human-in-the-loop review cycles into AI-assisted workflows.
- Establish clear accountability chains and documentation standards for AI-generated content.

Developed by Russell Nida

Contents

Part I Proactive Auditing – Building Verifiable Reasoning

For high-stakes policy, compliance, or logic audits, this section details a repeatable reasoning protocol.
Goal: not to make the model *prove* a claim but to force it to **test and justify every step**, producing an evidence-backed artifact.

1 Problem Definition

Clarify ambiguous claims before analysis.
Goal: Precisely define the claim and its boundaries before any reasoning begins.

Prompt Template
"Formalize the following claim. 1) Restate it precisely. 2) List explicit assumptions. 3) List hidden assumptions. 4) Define scope and domain."

---

2 Pre-Proof Audit (Falsification First)

Try to break the claim before defending it.

Prompt Template
"Evaluate whether this claim holds. 1) Restate formally. 2) List assumptions. 3) Generate three counterexamples. 4) Test each. 5) If any works, revise the claim for precision."

---

3 Structured Construction

Build reasoning you can audit, not persuasive prose.

Prompt Template
"Develop a structured argument with auditable components: PREMISES (P1, P2,…), LEMMAS showing dependencies, INFERENCE TRACKING naming each logical rule, and EDGE CASES noting exceptions."

---

4 Adversarial Review (Three Roles in One Pass)

Prompt
"Conduct an adversarial audit using three roles:

1. **Verifier** – validate each inference; cite rule.

2. **Gap Analyst** – find unsupported jumps; propose missing premises.
3. **Stress Tester** – attack with boundary cases or domain shifts.
   Output: Top 3 weaknesses + Rigor Score (0-10) + Fix List.”

Rigor Score Rubric

Range	Description
0–3	Informal narrative; missing premises; no counterexamples
4–6	Structured argument; some inference labels; counterexamples attempted
7–8	Complete premises; labeled inferences; edge cases handled
9–10	Fully auditable chain; adversarial challenges resolved; residual risk quantified

---

5 Cross-Representation Check

Expose ambiguity by forcing a restatement.
Prompt: “Restate the same reasoning in a different structure (list → table → narrative). If meaning changes, flag underspecified parts.”

---

6 Audit Acceptance & Deliverable Checklist

Audit Acceptance Criteria

- Claim formalized with explicit + hidden assumptions
- Premises traceable to evidence or agreed axioms
- Inference steps labeled and justified
- Counterexamples attempted; scope adjusted
- Edge cases documented
- Adversarial review completed + Rigor Score assigned
- Open risks disclosed

Deliverable Package

- Refined claim + scope

- Assumption register
- Premises / lemmas / labeled inferences
- Edge cases + boundary conditions
- Adversarial review output + Rigor Score
- Residual risks and recommended controls

Part II Reactive Diagnostics – Troubleshooting Model Failures

When outputs are poor, use this systematic method to diagnose and correct errors.
Key Principle: Every failure is a labeled example—capture it to prevent recurrence.

---

1 Common Failure Modes

Category	Symptoms	Typical Cause	Fix / Example
Hallucination	Confident but false claims	Weak grounding / no retrieval	Add retrieval; require citations; enforce “Unknown if not in sources.”
Inconsistency	Different answers to same query	High temperature / prompt drift	Temp 0.1–0.3; pin system prompt; compare n = 3 runs
Refusal / Over-caution	Declines reasonable tasks	Safety filter / vague context	Add policy context; reframe as hypothetical
Context Window Bleed	Forgets early constraints	Overlong context	Restate constraints every segment
Over-Literal Compliance	Follows text, misses intent	Missing purpose framing	Add role + intent (“As a risk analyst…”).
Spec Drift	Output diverges from requirements	Iterative edits w/out spec echo	Require “Spec Echo” header each output.

Category	Symptoms	Typical Cause	Fix / Example
Retrieval Contamination	Irrelevant docs influence output	Broad queries	Restrict corpus; show source IDs per claim.

---

2 Diagnostic Decision Tree (D.I.S.C.O.)

- **D – Define the deviation:** factual / logic / style / completeness?
    - If *factual* → enable retrieval and re-run.
- **I – Is it reproducible?** Run prompt ×3. If not → set temp ≤ 0.2.
- **S – Simplify the prompt:** reduce to one verb + object.
    - If output improves → prompt flaw; rewrite with C.G.A.F.R.
- **C – Check context/data:** were relevant docs retrieved and fresh?
    - If not → fix corpus or filters.
- **O – Operational parameters:** model version, decoding, post-processing.
    - If parser fails → inspect downstream scripts.

Stop / Go Gates

- **Stop** when root cause identified and fix validated.
- **Escalate** when ≥ 2 categories persist or impact ≥ High.

---

3 Categorizing Root Causes

Layer	Inspect	Example Diagnostic Question
Prompt	Instruction clarity / structure	Does the task contain one clear verb?
Context	Retrieval accuracy / freshness	Were the right docs retrieved?
Model	Version / temperature / decoding	Is sampling stochastic?
Post-Processing	Output parser / regex filters	Did a script discard valid text?

Tip: Work top-down (Prompt → Context → Model → Post-Processing).
Jumping to fine-tuning before these checks is wasted effort.

---

4 Corrective Techniques

A Prompt Refinement

- Apply C.G.A.F.R.; add "Reasoning → Answer" separator.
- Use **Spec Echo:** "You asked for X with Y constraints; plan → …"

B Model Parameter Adjustment

- Temp 0.1–0.3 for facts. Top-p 0.7–0.9 to reduce tangents.
- Fix max tokens to avoid truncation.

C Verification & Testing

- Side-by-side diffs vs baseline; score Accuracy / Completeness / Compliance / Traceability (0-5 each).
- Log model version, params, corpus, reviewer, date.

D Quirks & Workarounds (Model-Agnostic)

Issue	Workaround
Short-context bias	Repeat constraints every 600–800 tokens
Over-politeness / Refusals	Add lawful basis + role authority
Vision OCR drift	Require "Echo values used" before conclusion

---

Part III The Unified Quality System

1 Audit & Diagnostic Log (Single Source of Truth)

Field	Description	Example
Timestamp	UTC time of event	2025-11-01 T14:23 Z

Field	Description	Example
Session Type	Proactive Audit / Reactive Diagnostic	Proactive Audit
Prompt / Claim ID	Stable ID or hash	PRMPT-1243 A
Issue Type	Factual / Logic / Style / Completeness	Factual
Root Cause	Prompt / Data / Model / System	Prompt
Evidence Links	Source IDs / retrieval logs	SRC-A12, A13
Rigor Score (Audit)	0–10 per rubric	7
Business Impact	Low / Med / High / Critical	High
Corrective Action	What was changed	Rewrote prompt; temp 0.2
Reviewer	Human initials	A.S.
Retention Class	Policy code for logs	QA-12-RET-24 M

Compliance Note: Logs may contain sensitive data. Classify and retain per organizational policy and applicable privacy law (see TS-6 Security).

---

2 Integrated Escalation Path (with Triggers & SLAs)

Level	Responsibility	Trigger / SLA
1 – Team	Prompt/parameter fixes	SLA = 2 business days
2 – Tech Ops	RAG audit / A-B tests	≥ 3 High impact in 14 days
3 – AI Governance Board	Policy exceptions / performance drift	≥ 3 Critical in 30 days or systemic root cause
4 – Vendor	API performance / safety filters	Confirmed cross-model degradation

---

3 Quick Reference: Problem → First Diagnostic → Audit / Mitigation

Problem	First Diagnostic	Audit / Mitigation
Factual errors	Enable retrieval / tools / rerun	Require citations + “Unknown if not in sources.”
Inconsistent outputs	Temp 0.2; 3 runs; compare	Add Spec Echo; clarify claim
Refusals	Add lawful basis + role	Reframe as hypothetical; escalate policy
Flawed logic	Run D.I.S.C.O. (S + C)	Apply Proactive Audit Protocol
Retrieval noise	Inspect top-k docs	Constrain corpus; show source IDs

---

Key Takeaway

Quality LMM output results from both **rigorous construction** and **systematic debugging**. Building and repairing are two phases of the same discipline: **injecting human judgment** into AI workflows to produce reliable, auditable results.

Cross-References

- These audit prompts follow **C.G.A.F.R.** (Core Guide § 2.1).
- Verification discipline aligns with **Core Guide § 4.3**.
- Risk tiers and escalation align with **Core Guide § 1.3**.
- For orchestration and workflow chaining, see **Technical Supplement 4**.
- For security controls and retention policy, see **Technical Supplement 6**.
