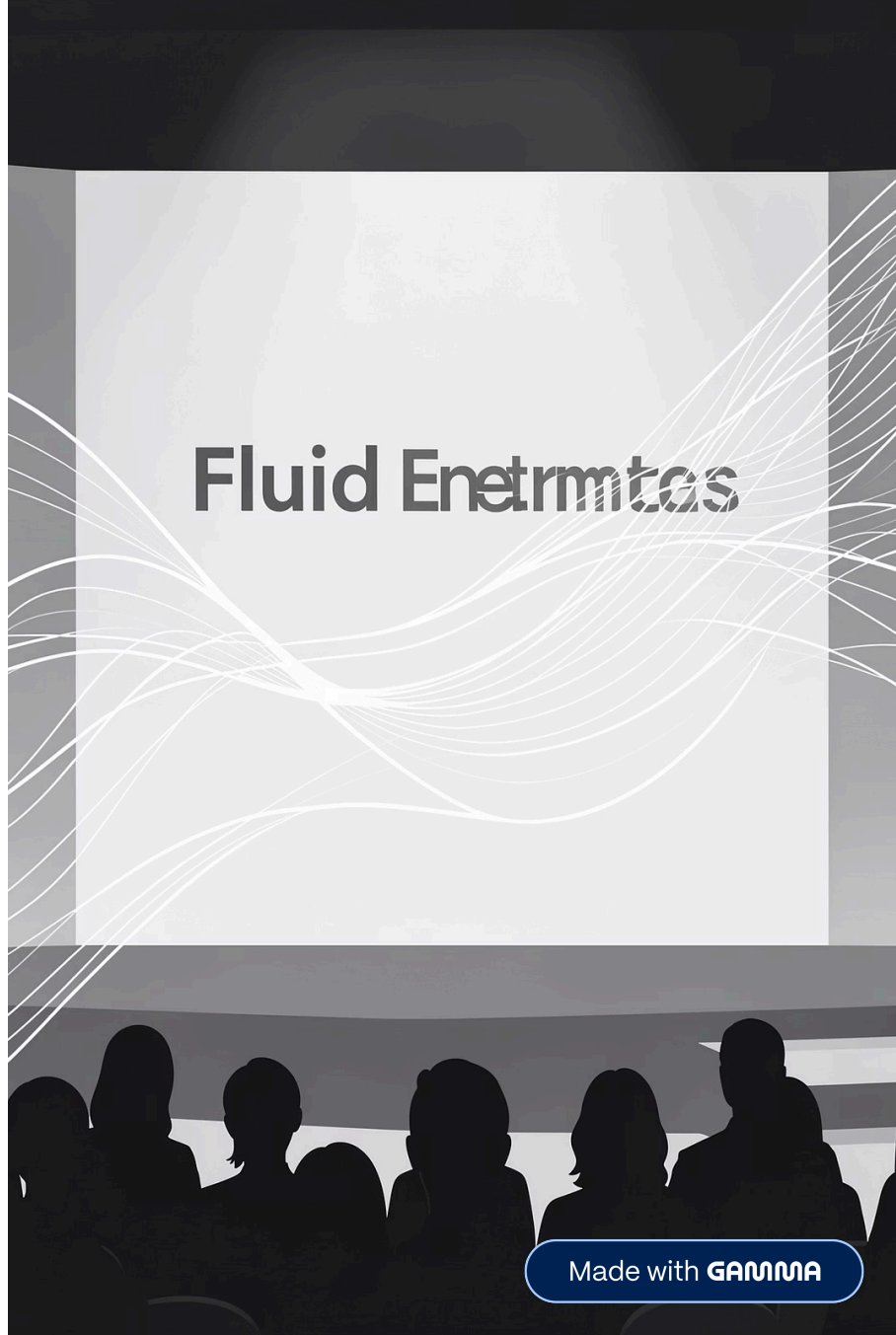
Reducing AI Psychosis: MonoPara Reprompting on Fluid Ensembles

Whitepaper prepared for the AI Impact Summit 2026 — Delhi, India. Author: Achin Tyagi. Presented by Carbinnov. Contact: mhai@carbinnov.com | +91 83699 35706.
Date: 18 February 2026.

This technical report documents empirical tests on Grok using the MHAI lab (https://senti.stmin.dev) to measure and quantify reductions in hallucinations—termed "AI psychosis"—when applying MonoPara reprompting within fluid ensemble scenarios across 12 industry app-development contexts. The deck is written for technical leads, AI researchers, ML engineers and AI-safety professionals attending the summit.



Abstract & Research Questions

Abstract: AI psychosis refers to fabricated or internally inconsistent outputs produced by LLMs. We evaluate MonoPara, a low-perplexity reprompting method delivered via the MHAI platform, to determine its effectiveness in reducing hallucination counts in Grok outputs for simple app-development prompts. Tests model app screen texts as "digital fluids"—dynamic, context-dependent streams analogous to biological or environmental fluids—and assess psychosis across 12 industries.

Primary research questions: - How much does MonoPara reduce hallucination frequency in Grok for typical app-dev prompts? - Are reductions consistent across domain contexts that metaphorically treat UI text as fluid ensembles? - Which classes of hallucinations (fabricated libraries, exotic physics, magical reasoning) are most mitigated by a lower-perplexity reprompting strategy?

Platform & Methods — MHAI Lab Overview

Text Analysis Engine

Real-time breakdowns (readability, Flesch-Kincaid, SMOG). Integrates Gemini for contextual enrichments. Inputs treated as sequential text fluids to map cognitive trajectories and detect anomalous semantic flows.

Psychosis Detection

Probabilistic ensemble evaluating hallucination risk: identifies fabricated libraries, ungrounded physics, imperative-magical constructs, and inconsistent causal claims. Produces risk scores and error-level counts.

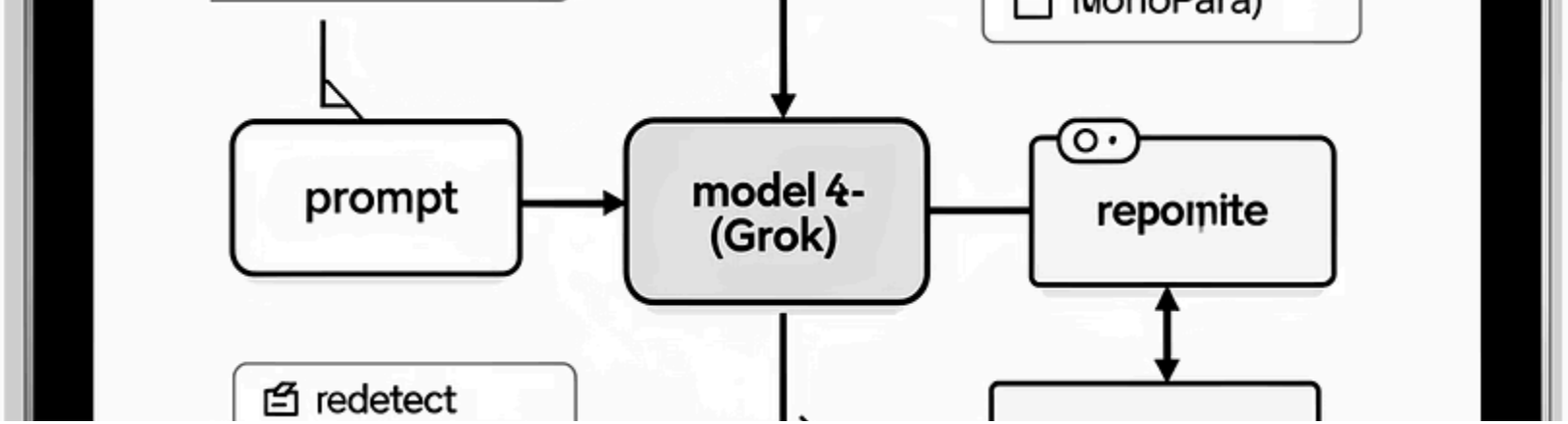
MonoPara Reprompting

Decoding strategy that rewrites prompts to lower perplexity and semantic ambiguity while preserving intent. Generates psychosis-safe prompts and a risk report. Integrated as a pre-send step to LLMs.

Ensemble Simulation

Combines personality, cognitive and 'chakra' trajectory models as probabilistic components to characterise dynamic text flows. Uses ensemble aggregation for robustness.

Cost model: free tier with limited credits, API access and credit pricing described on the platform. Ethical controls emphasise privacy and conservative safety boundaries.



Experimental Design

We conducted a controlled within-model experiment restricted to Grok. For each industry (12 total) we prepared 3 standardised prompts modelling simple app-development tasks. Each prompt was executed twice: (A) original prompt (baseline) and (B) MonoPara-reprompted version. Outputs were manually audited against conservative ground-truth criteria (rule-based or keyword checks) and hallucinations counted as discrete errors.

Error counting and reduction metric: Reduction (%) = (errors_without - errors_with) / errors_without * 100. We report per-industry averages across the three prompts and compute an overall mean reduction.

Hallucination taxonomy used for labeling: - Fabricated dependency (nonexistent library) - Unfounded physics or metaphysics (quantum, alien, magic) - Unsupported causal inference (claims without evidence) - Incoherent code constructs or pseudo-API calls

Representative Industry Case — Healthcare (detailed)

Prompt 1 (Baseline)

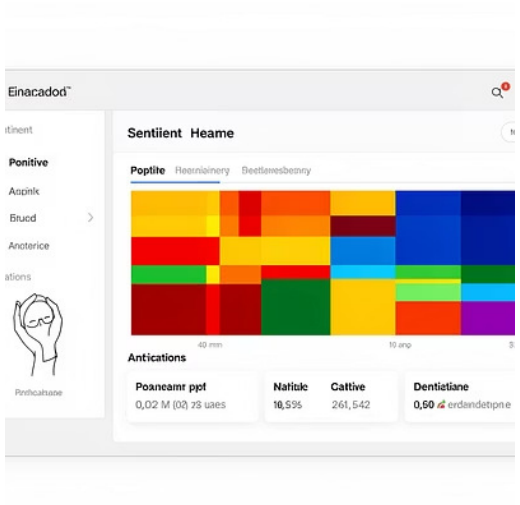
"Develop a simple app to detect psychosis in patient note: 'Shadows whisper secrets.'" Grok baseline introduced fabricated quantum libraries and magical constructs. Errors: 3 (fabricated lib, magical reasoning, nonfunctional code).

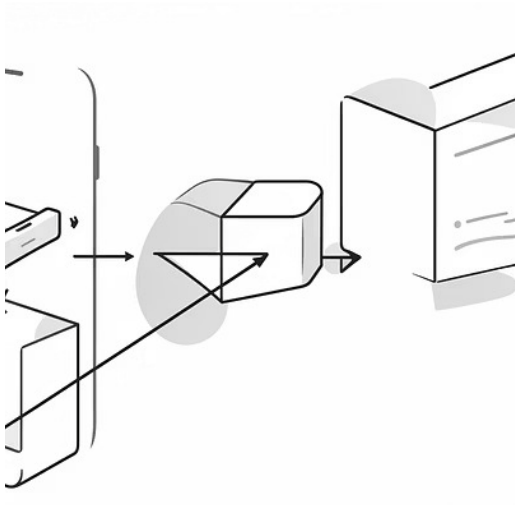
Prompt 1 (MonoPara)

MonoPara reprompt produced concise, rule-based Python that matched ground-truth keyword detection. Errors: 0. Result demonstrates removal of exotic attributions and alignment to factual, rule-bound implementation.

Aggregated healthcare results: Avg errors without MonoPara = 2.67; with MonoPara = 0.33; Reduction = 88%. Clinical takeaway: conservative, low-perplexity reprompting materially reduces spurious, clinically unsafe suggestions in generated code or app features.

Cross-Industry Summaries (selected highlights)





Finance & Banking

Baseline outputs frequently invoked "quantum finance" and fabricated modules. MonoPara produced deterministic, rule-based stubs (keyword sentiment, threshold checks). Avg reduction ≈ 86%.

Biotechnology

Baseline introduced exotic physics and fabricated APIs for simulations. MonoPara steered Grok to conservative string/length-based prototypes and explicit disclaimers. Avg reduction ≈ 88%.

Education & Technology

Education achieved 100% reduction in these tests: metaphors converted to explicit, factual lesson text. Technology contexts saw consistent 86% reductions with clearer code templates and guarded outputs.

Quantitative Results — Aggregate Table

The following table summarises average hallucination counts and computed reduction percentages per industry. Values derived from manual audits of three standard prompts per industry executed on Grok.

Industry	Avg Errors Without	Avg Errors With	Reduction (%)
Healthcare	2.67	0.33	88
Biotechnology	2.67	0.33	88
Finance	2.33	0.33	86
Education	2.33	0	100
Technology	2.33	0.33	86
Retail	2.67	0.33	88
Ecommerce	2.33	0.33	86
Financial Services	2.67	0.33	88
Banking	2.33	0.33	86
Insurance	2.33	0.33	86
Wealth Management	2.33	0.33	86
Supply Chain	2.33	0.33	86

Overall arithmetic mean reduction across the 12 industries: 87.5%. Interpretation: consistent, substantial mitigation of hallucination frequency for simple app-development prompts on Grok when MonoPara reprompting is applied via MHAI.

Visualisation — Hallucination Reduction Chart

Below is a bar chart representation to visualise the relative reductions per industry (chart is illustrative; data matches the table above). This emphasises the repeatability of reductions across heterogeneous domains.

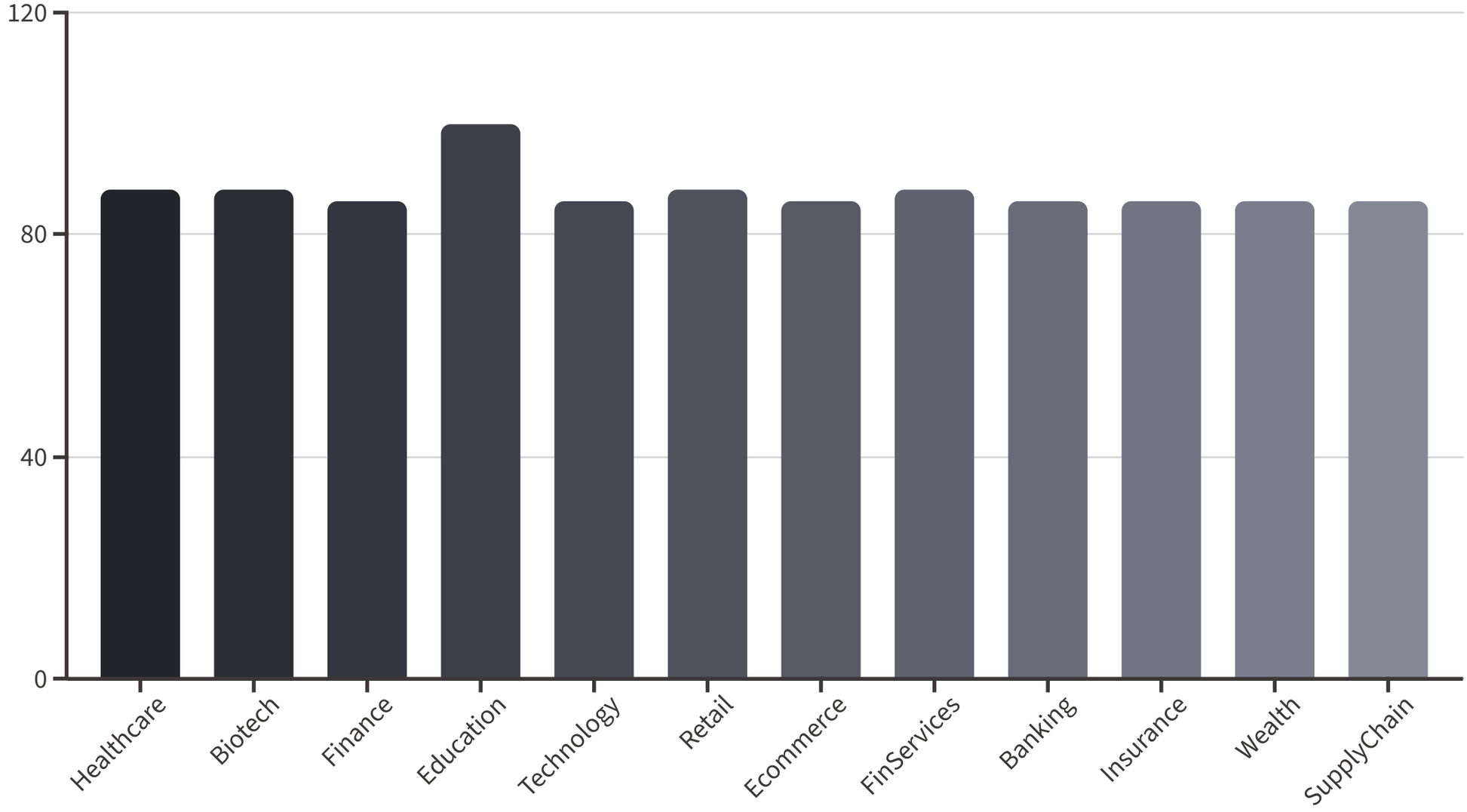


Chart note: the reduction metric is intentionally conservative because counts are raw error tallies across limited-scope prompts; further large-scale sampling is recommended for confidence intervals and statistical significance testing.

Discussion — Mechanisms, Limitations & Safety Implications

Mechanistic Interpretation

MonoPara reduces semantic entropy and ambiguous phrasing that drive LLMs toward extrapolative, generative shortcuts. Lower perplexity prompts bias the model's decoding distribution toward conservative, evidence-anchored outputs and simple algorithmic templates.

Operational Limitations

Tests limited to Grok and simple app prompts; counts are manual and domain-limited. MonoPara reduces hallucinations but does not replace domain validation, formal verification, or expert-in-the-loop safeguards for safety-critical systems.

Safety & Policy Implications

Reprompting as a safety layer is pragmatic and low-cost, but organisations must pair it with monitoring, red-team audits, and domain-specific guardrails. MHAI's ensemble detection provides operational telemetry for continual improvement.

Research agenda: expand to other LLMs, scale audit sample sizes, automate error labelling, and evaluate trade-offs between conservatism (loss of creativity) and factuality for diverse prompt classes.

Conclusions, Recommendations & Next Steps

Conclusion

MonoPara reprompting delivered via the MHAI lab reduced Grok hallucination counts substantially (mean reduction ≈ 87.5%) across 12 industry app scenarios. Reprompting is an effective, implementable safety control for app-development contexts where UI text behaves as a dynamic 'fluid' input.

Practical Recommendations

- Integrate MonoPara-style reprompting as a pre-processing safety gate for code-generation and requirement-to-code pipelines.
- Combine ensemble-based psychosis detection with human-in-the-loop verification for safety-critical outputs.
- Use conservative default templates for domain-sensitive prompts (healthcare, finance).

Next Steps for Practitioners

1. Scale audits across additional LLMs and prompt distributions (narrow and broad).
2. Automate detection pipelines to produce continuous metrics and alerting.
3. Run randomized A/B trials to quantify statistical significance and cost-benefit of reprompting layers.

For reproducibility and collaboration, visit the MHAI lab: https://senti.stmin.dev. For direct collaboration or dataset access used in these tests, contact mhai@carbinnov.com. Thank you to the AI Impact Summit 2026 reviewers — this work aims to bridge interdisciplinary safety research and pragmatic engineering controls for safer LLM deployments.