# LUCID: Leveraging Unverified Claims Into Deliverables

A Neuroscience-Grounded Framework for Exploiting Large Language Model Hallucination as a Software Specification Engine

Ty Wells
ty@snapperland.com
https://github.com/gtsbahamas/hallucination-reversing-system

February 2026

**Abstract**

Large language model (LLM) hallucination is universally treated as a defect to be minimized. We argue this framing is backwards. Hallucination—the confident generation of plausible but unverified claims—is computationally identical to the brain's pattern completion mechanism that underlies both perception and imagination. We present LUCID (Leveraging Unverified Claims Into Deliverables), a development methodology that deliberately invokes LLM hallucination, extracts the resulting claims as testable requirements, verifies them against a real codebase, and iteratively converges hallucinated fiction toward verified reality. By prompting an LLM to author Terms of Service for an application that does not yet exist, we exploit the model's confabulatory tendency to produce comprehensive, precise, multi-dimensional specifications—covering functionality, security, data privacy, performance, and legal compliance—in seconds. We provide theoretical grounding through three convergent lines of evidence: (1) the mathematical equivalence between transformer attention and hippocampal pattern completion, (2) the predictive processing framework from cognitive neuroscience, and (3) the REBUS model of psychedelic hallucination. We demonstrate the framework on a real-world application, achieving convergence from 57.3% to 90.8% compliance across six iterations. We position LUCID as the software engineering analogue of protein hallucination [Anishchenko et al., 2021], where neural network "dreams" serve as blueprints validated against physical reality. Formal impossibility results proving that hallucination cannot be eliminated from LLMs [Xu et al., 2024, Banerjee et al., 2024] suggest that harnessing hallucination may be more productive than fighting it.

**Keywords:** LLM hallucination, confabulation, predictive processing, specification generation, requirements engineering, iterative convergence, neuroscience of hallucination

## 1 Introduction

The field of large language model research has, since the publication of GPT-3 [Brown et al., 2020], treated hallucination as a failure mode requiring mitigation. Retrieval-Augmented Generation [Lewis et al., 2020], Chain-of-Thought prompting [Wei et al., 2022], Chain-of-Verification [Dhuliawala et al., 2024], and Constitutional AI [Bai et al., 2022] all share a common goal: reduce the rate at which models generate false, ungrounded, or fabricated content.

This work takes the opposite position. We argue that:

1. **Hallucination is mathematically inevitable** in any model that generalizes beyond its training distribution [Xu et al., 2024, Banerjee et al., 2024].

2. **Hallucination is computationally equivalent** to the pattern completion mechanism that underlies human perception, memory, and imagination [Ramsauer et al., 2021, Clark, 2023, Friston, 2010].

3. **Hallucination, when deliberately invoked and externally verified, is a productive signal**—the richest, cheapest, and fastest method available for generating comprehensive software specifications.

We present LUCID, a six-phase iterative methodology:

**Describe → Hallucinate → Extract → Build → Converge → Regenerate**

The key innovation is Phase 2: we ask an LLM to write a Terms of Service document for an application that does not exist. The model confabulates—inventing specific capabilities, data handling procedures, performance guarantees, user rights, and limitations—in the precise, declarative language that legal documents demand. Each confabulated claim becomes a testable requirement. Verification against the actual codebase reveals which claims are real (the model guessed correctly), which are aspirational (plausible but unimplemented), and which are infeasible (should be dropped). Regeneration feeds verified reality back to the model, producing an updated specification that is progressively more grounded with each iteration.

This paper makes four contributions:

1. A **formal framework** (LUCID) for exploiting LLM hallucination as a specification engine, with a working open-source implementation.
2. **Theoretical grounding** from cognitive neuroscience, connecting LLM hallucination to predictive processing, memory reconsolidation, confabulation, and lucid dreaming.
3. **Empirical results** demonstrating convergence from 57.3% to 90.8% specification-reality alignment across six iterations on a production application.
4. A **positioning argument** that LUCID is the software engineering analogue of protein hallucination—the Nobel Prize-winning methodology where neural network "dreams" serve as blueprints for novel biological structures [Anishchenko et al., 2021].

## 2 Background and Related Work

### 2.1 The Impossibility of Eliminating Hallucination

Two independent formal results establish that hallucination is intrinsic to LLMs:

Xu et al. [2024] prove, using learning theory, that any LLM with a computable ground truth function will inevitably hallucinate when used as a general problem solver. The proof shows that LLMs cannot learn all computable functions, and therefore must sometimes generate outputs inconsistent with ground truth.

Banerjee et al. [2024] reach the same conclusion via Gödel's First Incompleteness Theorem, the Halting Problem, and the Emptiness Problem. They demonstrate that hallucination stems from the fundamental mathematical and logical structure of LLMs, and cannot be eliminated through architectural improvements, dataset enhancements, or fact-checking mechanisms.

These results motivate our core argument: if hallucination cannot be eliminated, a productive methodology must *harness* it rather than fight it.

### 2.2 Self-Correction Requires External Feedback

A critical finding from Huang et al. [2024] demonstrates that LLMs cannot reliably self-correct their reasoning without external feedback. Performance can *degrade* after self-correction attempts. The bottleneck is feedback generation: LLMs cannot produce reliable signals about their own errors.

This result has direct implications for LUCID's design. The verification phase uses the actual codebase—not the model's self-assessment—as ground truth. This aligns with the LLM-Modulo framework [Kambhampati et al., 2024], which pairs LLMs with external sound verifiers and achieves dramatic accuracy improvements (24% → 98% on blocks world planning).

### 2.3 Hallucination as Productive Signal

Several lines of work have begun to reframe hallucination as useful:

**Protein hallucination.** Anishchenko et al. [2021] used the trRosetta neural network to iteratively optimize random amino acid sequences via Monte Carlo sampling—a process the authors called "hallucination." The hallucinated protein structures, when expressed in bacteria, closely matched predictions. David Baker's subsequent work generated millions of novel proteins, leading to approximately 100 patents and 20+ biotech companies, and the 2024 Nobel Prize in Chemistry.

**Drug discovery.** Various Authors [2025a] found that hallucinations significantly improve predictive accuracy for molecule property prediction by acting as "implicit counterfactuals"—speculative interpretations that help LLMs generalize over unseen compounds.

**Confabulation value.** Sui et al. [2024] empirically demonstrated that LLM confabulations display increased narrativity and semantic coherence relative to veridical outputs, mirroring the human propensity to use narrativity as a cognitive resource for sense-making.

**Computational imagination.** Various Authors [2025b] deliberately amplified LLM hallucinations using LoRA fine-tuning for creative applications, reframing hallucinations as "computational imagination."

### 2.4 Closed-Loop Verification Systems

LUCID builds on established generate-verify-refine architectures. Chain-of-Verification [Dhuliawala et al., 2024] implements a four-step draft-verify-revise process, reducing factual hallucinations by 50–70%. CRITIC [Gou et al., 2023] introduces tool-interactive critiquing where LLMs interact with external tools to verify and correct output iteratively. Self-Refine [Madaan et al., 2023] demonstrated the generate-critique-refine loop. VENCE [Chen et al., 2023], whose title "Converge to the Truth" anticipates our convergence framing, formulates factual error correction as iterative constrained editing.

LUCID differs from all of these in two respects: (1) verification occurs against a *codebase*, not against web knowledge or reference documents, and (2) the output is not a corrected text but a *specification* that drives development.

### 2.5 Adjacent Methodologies

| Methodology | Specification Source | AI Role | Hallucination | Loop |
|---|---|---|---|---|
| Spec-Driven Dev | Human-authored spec | Implements | Prevents | No |
| Readme-Driven Dev | Human-authored README | None | N/A | No |
| Design Fiction | Human-authored fiction | Optional | Intentional (human) | Loose |
| Protein Halluc. | Neural network output | Generates | Exploits | Validate |
| Vibe Coding | Human prompt | Generates | Tolerates | No |
| **LUCID** | **AI-hallucinated ToS** | **Hallucinates, extracts, verifies** | **Exploits** | **Yes** |

Table 1: Comparison of specification methodologies. LUCID is the only approach combining AI-generated specification, deliberate hallucination exploitation, and iterative convergence verification against a codebase.

## 3 Theoretical Foundations

We ground LUCID in three convergent lines of evidence from cognitive neuroscience and mathematical machine learning.

### 3.1 Transformers as Associative Memory

Ramsauer et al. [2021] proved that transformer self-attention is mathematically equivalent to the update rule of modern Hopfield networks—associative memory systems that retrieve stored patterns from partial cues. Given a query (partial cue), the attention mechanism retrieves the most similar stored pattern (key-value pair). This is pattern completion: the same computation performed by the hippocampal CA3 autoassociative network.

Whittington et al. [2021] extended this connection, showing that transformer architectures relate to hippocampal formation computations, specifically place cells and grid cells used for spatial memory and navigation.

The implication is direct: when an LLM generates text about a nonexistent application, it is performing pattern completion from partial cues (the prompt) against distributed representations (training data). The output includes both veridical completions (patterns the model has reliably encoded) and confabulated completions (plausible extensions that overshoot the stored patterns). This is identical to what the hippocampus does when reconstructing a memory from a partial cue—some details are accurate recall, and some are gap-filling confabulation [Bartlett, 1932, Loftus, 2005].

### 3.2 Predictive Processing: Perception as Controlled Hallucination

The dominant framework in modern cognitive neuroscience—*predictive processing*—holds that the brain is fundamentally a prediction machine [Friston, 2009, 2010, Clark, 2013, Hohwy, 2013, Seth, 2021].

The brain does not passively receive sensory data. It generates top-down predictions about what it expects to perceive, compares those predictions against incoming sensory signals, and propagates only the **prediction error** upward through the cortical hierarchy. When predictions are good, experience feels normal. When predictions are unconstrained by sensory data, the result is hallucination.

As Seth [2021] states: "We're all hallucinating all the time; when we agree about our hallucinations, we call it reality." Clark [2023] formalizes this: "Perception is controlled hallucination. Hallucination is uncontrolled perception." Both use the same generative machinery; the difference is the degree of constraint from external evidence.

This framework maps precisely onto LLM behavior:

| **Predictive Processing** | **LLM Generation** |
|---|---|
| Internal generative model | Trained transformer weights |
| Top-down prediction | Next-token probability distribution |
| Sensory data constraining predictions | Context window, RAG, grounding |
| Prediction error signal | Loss during training |
| Hallucination (unconstrained) | Generation without factual grounding |
| Perception (constrained) | Generation with strong context/retrieval |

Table 2: Mapping between the predictive processing framework and LLM generation.

LUCID deliberately operates in the "unconstrained" mode during the Hallucinate phase, then progressively introduces constraint during Converge and Regenerate. Each iteration moves the system along the spectrum from hallucination toward perception.

### 3.3 REBUS: The Psychedelic Model and Temperature

Carhart-Harris and Friston [2019] proposed the REBUS model (Relaxed Beliefs Under Psychedelics), integrating predictive coding with the entropic brain hypothesis [Carhart-Harris et al., 2014]. Psychedelics reduce the **precision weighting** of high-level priors—the brain's top-down constraints. When these constraints relax, novel associations form that are normally suppressed, producing both hallucination and creative insight.

This maps directly to the **temperature parameter** in LLM sampling. High precision (brain) or low temperature (LLM) yields conservative, factual output. Low precision or high temperature yields divergent, hallucination-prone generation. The REBUS model explains why psychedelic states are associated with both hallucination *and* creativity—relaxing constraints enables novel pattern combinations. LUCID exploits this: the Hallucinate phase operates at "high temperature," and the convergence loop progressively increases "precision."

### 3.4 Confabulation: The Correct Term

Multiple authors have argued that *confabulation*—not hallucination—is the correct term for LLM fabrication. Smith et al. [2023] use neuroanatomical metaphor to argue that LLMs behave like an "unmitigated confabulating left hemisphere." Hirstein [2005] identified the key structural insight: the creative ability to construct plausible responses and the ability to *check* them are **separate processes** in the brain. Confabulation occurs when the checking process fails. LLMs have the generative process but lack the checker. LUCID reintroduces the checker via external codebase verification.

Schnider [2003] identified the neural substrate: the posterior medial orbitofrontal cortex performs **reality filtering**—suppressing activated memories that do not pertain to ongoing reality. LUCID's verification step serves as the computational analogue of orbitofrontal reality filtering.

### 3.5 Memory Reconsolidation

Loftus [2005] demonstrated that human memory is reconstructive, not reproductive. When recalling an event, the brain regenerates it from partial traces, filling gaps with plausible details.

The mechanism is hippocampal pattern completion [Yassa and Stark, 2011]: given a partial cue, the CA3 autoassociative network fires the full stored pattern. Since transformer self-attention is mathematically equivalent to this computation [Ramsauer et al., 2021], the analogy between "a trickle of memory triggering hallucination" and LLM next-token prediction is not metaphorical—it describes the same underlying computation in different substrates.

### 3.6 Lucid Dreaming: The System's Namesake

The name "LUCID" embodies the neuroscience of *lucid dreaming*—the state where a dreamer becomes metacognitively aware that they are dreaming while remaining in the dream [Baird et al., 2019, Filevich et al., 2015]. The dorsolateral prefrontal cortex (executive function, reality monitoring) reactivates, and the dreamer can *steer* the dream without stopping it.

| **Lucid Dreaming** | **LUCID System** |
|---|---|
| The dream (unconstrained generation) | Hallucinated Terms of Service |
| Becoming lucid (recognizing the dream) | Extract phase (claims become hypotheses) |
| Reality testing (checking dream vs. reality) | Verify phase (claims checked against code) |
| Dream steering (directing content) | Regenerate phase (feeding reality back) |
| Prefrontal cortex reactivation | Human judgment evaluating verdicts |
| Staying in the dream while aware | Staying in the generative loop while verifying |

Table 3: Mapping between lucid dreaming neuroscience and the LUCID system.

A lucid dreamer does not fight the dream. They participate with awareness, harvesting creative content while maintaining the ability to distinguish generated from real. LUCID does exactly this to AI hallucination.

This also maps onto dual-process theory [Kahneman, 2011]: LLMs are pure System 1 (fast, automatic, confabulation-prone). LUCID wraps a System 1 machine in a System 2 process. The convergence loop *is* the deliberate, checking function that the model lacks.

## 4 The LUCID Framework

### 4.1 Overview

LUCID is a six-phase iterative cycle that converts loose application descriptions into verified software specifications through controlled exploitation of LLM hallucination.

1. **Describe.** Provide a deliberately incomplete application description. Gaps are where confabulation does its most productive work.

2. **Hallucinate.** The LLM writes Terms of Service as if the application is live in production. The prompt instructs declarative statements ("The Service processes X" rather than "may process").

3. **Extract.** Each declarative claim is parsed into a structured requirement with category (functionality, security, data-privacy, operational, legal), severity (critical, high, medium, low), and testability flag.

4. **Build.** Implementation proceeds using any methodology. ToS-derived claims serve as acceptance criteria.

5. **Converge.** Two-step verification: (a) file selection—identify which source files contain evidence for each claim; (b) verdict assignment—PASS, PARTIAL, FAIL, or N/A with evidence.

6. **Regenerate.** Verified reality is fed back. The model writes updated ToS retaining PASS claims, revising PARTIAL claims, dropping or revising FAIL claims, and hallucinating new capabilities.

### 4.2 Why Terms of Service

ToS is the ideal hallucination vehicle because the document format demands specificity across multiple dimensions simultaneously. Service descriptions produce functional requirements; acceptable use policies produce input validation rules; data handling sections produce privacy and security requirements; limitation sections produce performance boundaries; SLA sections produce reliability requirements; termination sections produce lifecycle requirements; liability sections produce error handling requirements.

No other document format forces this level of specificity across this many dimensions. Legal language cannot be vague—"The Service may do things" is not a valid legal clause—so the format forces the model to hallucinate *precisely*. A typical hallucination produces 400–600 lines of dense legal text containing 80–150 extractable claims.

### 4.3 Compliance Score

Verification assigns verdicts, and compliance is computed as:

$$S = \frac{N_{\text{pass}} + 0.5 \cdot N_{\text{partial}}}{N_{\text{total}} - N_{\text{na}}} \times 100 \tag{1}$$

where $N_{\text{pass}}$, $N_{\text{partial}}$, $N_{\text{total}}$, and $N_{\text{na}}$ are the counts of PASS, PARTIAL, total, and N/A verdicts respectively.

### 4.4 Convergence Dynamics

Each iteration shifts the ratio of accurate-to-hallucinated claims. PASS claims are retained as verified. New hallucinations are generated in context of verified capabilities, making them more grounded. The gap between specification and reality shrinks monotonically (see Section 6).

## 5 Implementation

LUCID is implemented as an open-source CLI tool in TypeScript (Node.js 20+), using the Anthropic Claude SDK for all LLM interactions. The tool provides commands for each phase: `lucid hallucinate`, `lucid extract`, `lucid verify`, `lucid report`, `lucid remediate`, and `lucid regenerate`. All artifacts are stored in `.lucid/iterations/{N}/` directories, maintaining a complete audit trail.

Claim extraction uses streaming API calls with validation against strict type schemas. Codebase verification uses a two-step process: file selection (model identifies relevant files from the file tree) followed by verdict assignment (model reads file contents truncated to 10K characters each, 100K total context). Claims are processed in batches of 15.

Failed and partial verifications are transformed into actionable remediation tasks with title, description, action type (add/modify/remove/configure), target files, estimated effort, and code-level guidance.

## 6 Empirical Results

### 6.1 Case Study

We applied LUCID to a production Next.js application—a career development platform with AI coaching, financial planning, goal tracking, and document management—comprising approx-

imately 30,000 lines of TypeScript across 200+ files.

### 6.2 Convergence Data

| Iteration | Score | Claims | PASS | PARTIAL | FAIL | N/A |
|---|---|---|---|---|---|---|
| 1 | ∼35% (est.) | 91 | — | — | — | — |
| 3 | 57.3% | 91 | 38 | 15 | 32 | 6 |
| 4 | 69.8% | 91 | 47 | 18 | 20 | 6 |
| 5 | 83.2% | 91 | 61 | 15 | 9 | 6 |
| 6 | 90.8% | 91 | 68 | 12 | 5 | 6 |

Table 4: Convergence across iterations. Compliance score increased monotonically.

After six iterations, five claims remained as FAIL: weekly job market data refresh (hardcoded), subscription downgrade retention logic, ClamAV malware scanning, server-side rate limiting, and account lockout parameter mismatch. All five represent genuine missing functionality—not false positives.

### 6.3 Verification Layers (Iteration 6)

| Layer | Score |
|---|---|
| Code-level verification | 84.5% |
| End-to-end testing | 100% |
| UX audit | 85% |
| Click verification | 95% |
| **Composite** | **90.8%** |

Table 5: Multi-layer verification scores at iteration 6.

### 6.4 Token Economics

A complete six-iteration cycle cost approximately $17 in API tokens—producing a verified specification with 91 claims, a gap report, and a prioritized remediation plan. Individual phases ranged from $0.15 (hallucination) to $1.50 (verification per iteration).

## 7 Discussion

### 7.1 Why LUCID Works

LUCID's effectiveness can be explained through three converging mechanisms:

**Pattern completion from training data.** Given a vague application description, the model completes the pattern using representations learned from millions of ToS documents, software documentation sets, and codebases. The completions reflect genuine statistical regularities—real applications in this domain *tend* to have these features.

**Legal language as forcing function.** The ToS format constrains the model to produce specific, testable, declarative claims. A prompt asking for "features" yields vague bullets. A prompt asking for legally binding commitments yields precise specifications.

**External verification as reality filtering.** Following Schnider's (2003) model of orbitofrontal reality filtering, the verification step suppresses hallucinated claims that do not

correspond to reality. This is the System 2 checking process the model lacks [Kahneman, 2011], and that Huang et al. [2024] proved cannot be performed by the model itself.

### 7.2 Relationship to Protein Hallucination

The closest existing analogue to LUCID is Baker Lab's protein hallucination [Anishchenko et al., 2021]. The structural parallel is exact: neural network generates novel structures that do not exist; laboratory/codebase validates them; functional results become candidates; iterative sampling refines candidates. Baker's methodology produced approximately 100 patents and 20+ biotech companies. The insight—that neural network "dreams" can serve as blueprints for real-world engineering—earned the 2024 Nobel Prize in Chemistry. LUCID applies the identical insight to software engineering.

### 7.3 The Lucid Dreaming Design Principle

The neuroscience of lucid dreaming provides a design principle, not just a metaphor: (1) *Do not wake up*—do not suppress hallucination; (2) *Gain metacognitive awareness*—the Extract phase parses claims as unverified hypotheses; (3) *Reality-test within the dream*—verify against code; (4) *Steer, don't control*—provide context and let the model hallucinate freely within it.

### 7.4 Beyond Hallucination Reduction

The deeper claim is that LUCID is not a hallucination *reduction* technique. It is a specification *extraction* technique that uses hallucination as input signal. No human requirements gathering process produces 91 testable claims spanning functionality, security, data privacy, performance, operations, and legal compliance in 30 seconds. The hallucination does.

### 7.5 Toward One-Shot Development

LUCID is not one-shot; it is a convergence loop. But neither is perception—perception is a rapid inference loop (predict → compare → update) that runs so fast it feels instantaneous [Friston, 2009]. The trajectory toward one-shot is determined by initial hallucination quality (improves with better models) and verification speed (improves with better tooling). At the limit, a sufficiently good model with sufficiently fast verification converges in a single iteration.

### 7.6 Limitations

LUCID's claim quality depends on the underlying model's training data. The verification step uses an LLM (not formal verification), introducing potential for verification errors. LUCID is designed for software applications; applicability to other engineering domains is untested. Hallucinated ToS documents should not be used as legal documents without qualified review.

## 8 Future Work

**Multi-document hallucination.** API documentation, user manuals, privacy policies, and compliance certifications each force different kinds of specificity. Simultaneous multi-document hallucination with cross-document consistency verification may improve specification coverage.

**Formal verification integration.** Replacing LLM-based verification with property-based testing, model checking, or static analysis for specific claim categories would increase fidelity.

**Continuous monitoring.** Integration into CI/CD pipelines would enable real-time specification drift detection on each code push.

**Cross-domain transfer.** The core insight—hallucination as blueprint, verification as reality filter—may transfer to regulatory compliance, safety certifications, and standards conformance.

**Hallucination quality benchmarks.** The initial compliance score (before remediation) may serve as a metric for model capability—a "specification hallucination benchmark."

## 9 Conclusion

We have presented LUCID, a framework that inverts the dominant paradigm around LLM hallucination. Rather than treating hallucination as a defect, LUCID exploits it as a specification engine—the fastest, cheapest, most comprehensive method available for generating testable software requirements from minimal input.

The theoretical grounding is robust. Transformer attention is mathematically equivalent to hippocampal pattern completion [Ramsauer et al., 2021]. The predictive processing framework establishes that hallucination and perception are the same generative computation under different constraint conditions [Clark, 2023, Seth, 2021, Friston, 2010]. The REBUS model explains why relaxing constraints enables creative discovery [Carhart-Harris and Friston, 2019]. The confabulation literature provides the structural insight that generation and verification are separable processes [Hirstein, 2005, Schnider, 2003]—LLMs have the former but not the latter.

LUCID provides the latter. By verifying hallucinated claims against real codebases and feeding verified reality back into the generative process, LUCID creates a convergence loop that progressively transforms hallucinated fiction into verified specification. Empirical results demonstrate convergence from 57.3% to 90.8% compliance across six iterations at a cost of approximately $17 in API tokens.

The formal impossibility results of Xu et al. [2024] and Banerjee et al. [2024] establish that hallucination cannot be eliminated. If hallucination is inevitable, the productive response is to harness it. LUCID is one such harness—and the precedent of protein hallucination earning the 2024 Nobel Prize suggests that the principle of "neural network dreams as engineering blueprints" has transformative potential across domains.

We leave the reader with a reformulation of Seth's dictum:

> *Normal specification is hallucination constrained by reality. LUCID is the first development methodology that uses this principle: generate freely, then constrain iteratively, just as the brain does.*

## References


Ivan Anishchenko, Samuel J Pellock, Tamuka M Chidyausiku, Theresa A Ramelot, Sergey Ovchinnikov, Jingzhou Hao, Kelly A Bator, Yaira M Baez-Santos, Alex Kang, Asim K Bera, et al. De novo protein design by deep network hallucination. *Nature*, 600(7889):547–552, 2021.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Benjamin Baird, Sergio A Mota-Rolim, and Martin Dresler. The cognitive neuroscience of lucid dreaming. *Neuroscience & Biobehavioral Reviews*, 100:305–323, 2019.

Sourav Banerjee, Ayushi Sarkar, and Philippe Schwaller. LLMs will always hallucinate, and we need to live with this. *arXiv preprint arXiv:2409.05746*, 2024.

Frederic C Bartlett. *Remembering: A Study in Experimental and Social Psychology*. Cambridge University Press, 1932.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

Robin L Carhart-Harris and Karl J Friston. REBUS and the anarchic brain: Toward a unified model of the brain action of psychedelics. *Pharmacological Reviews*, 71(3):316–344, 2019.

Robin L Carhart-Harris, Robert Leech, Peter J Hellyer, Murray Shanahan, Amanda Feilding, Enzo Tagliazucchi, Dante R Chialvo, and David Nutt. The entropic brain: a theory of conscious states informed by neuroimaging research with psychedelic drugs. *Frontiers in Human Neuroscience*, 8:20, 2014.

Jiangjie Chen et al. Converge to the truth: Factual error correction via iterative constrained editing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Andy Clark. Whatever next? Predictive brains, situated agents, and the future of cognitive science. *Behavioral and Brain Sciences*, 36(3):181–204, 2013.

Andy Clark. *The Experience Machine: How Our Minds Predict and Shape Reality*. Pantheon, 2023.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-verification reduces hallucination in large language models. *Findings of the Association for Computational Linguistics: ACL 2024*, 2024.

Elisa Filevich, Martin Dresler, Timothy R Brick, and Simone Kühn. Metacognitive mechanisms underlying lucid dreaming. *Journal of Neuroscience*, 35(3):1082–1088, 2015.

Karl Friston. The free-energy principle: a rough guide to the brain? *Trends in Cognitive Sciences*, 13(7):293–301, 2009.

Karl Friston. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11(2):127–138, 2010.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*, 2023.

William Hirstein. *Brain Fiction: Self-Deception and the Riddle of Confabulation*. MIT Press, 2005.

Jakob Hohwy. *The Predictive Mind*. Oxford University Press, 2013.

Jie Huang, Shubham Dasgupta, Debashis Ghosh, John Langford, and Jason Weston. Large language models cannot self-correct reasoning yet. *International Conference on Learning Representations*, 2024.

Daniel Kahneman. *Thinking, Fast and Slow*. Farrar, Straus and Giroux, 2011.

Subbarao Kambhampati et al. LLM-modulo: An LLM-modulo framework for task planning. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

Elizabeth F Loftus. Planting misinformation in the human mind: A 30-year investigation of the malleability of memory. *Learning & Memory*, 12(4):361–366, 2005.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36, 2023.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, et al. Hopfield networks is all you need. *International Conference on Learning Representations*, 2021.

Armin Schnider. Spontaneous confabulation and the adaptation of thought to ongoing reality. *Nature Reviews Neuroscience*, 4(8):662–671, 2003.

Anil Seth. *Being You: A New Science of Consciousness*. Dutton, 2021.

Matthew Smith, Nina Greaves, and Trishan Panch. Hallucination or confabulation? Neuroanatomy as metaphor in large language models. *PLOS Digital Health*, 2(11):e0000388, 2023.

Daniel Sui, Eamon Duede, Yue Wu, and Richard So. Confabulation: The surprising value of large language model hallucinations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024.

Various Authors. Can hallucinations help? Boosting LLMs for drug discovery. *arXiv preprint arXiv:2501.13824*, 2025a.

Various Authors. Purposefully induced psychosis (PIP): Embracing hallucination as imagination in large language models. *arXiv preprint arXiv:2504.12012*, 2025b.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

James C R Whittington et al. Relating transformers to models and neural representations of the hippocampal formation. *arXiv preprint arXiv:2112.04035*, 2021.

Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*, 2024.

Michael A Yassa and Craig E L Stark. Pattern separation in the hippocampus. *Trends in Neurosciences*, 34(10):515–525, 2011.